Springer Tracts in Advanced Robotics 105

Luis Mejias
Peter Corke
Jonathan Roberts *Editors*

Field and Service Robotics

Results of the 9th International Conference

Springer

Springer Tracts in Advanced Robotics 105

For further volumes:
http://www.springer.com/series/5208

Editorial Advisory Board

STAR (Springer Tracts in Advanced Robotics) has been promoted under the auspices of EURON (European Robotics Research Network)

Luis Mejias · Peter Corke
Jonathan Roberts
Editors

Field and Service Robotics

Results of the 9th International Conference

Springer

Editors

Luis Mejias
Science and Engineering Faculty
Queensland University of Technology
Brisbane, Queensland
Australia
e-mail: luis.mejias@qut.edu.au

Peter Corke
Science and Engineering Faculty
Queensland University of Technology
Brisbane, Queensland
Australia
e-mail: peter.corke@qut.edu.au

Jonathan Roberts
Autonomous Systems
CSIRO Computational Informatics
Pullenvale, Queensland
Australia

and

Science and Engineering Faculty
Queensland University of Technology
Brisbane, Queensland
Australia
e-mail: jonathan.roberts@qut.edu.au

ISSN 1610-7438 ISSN 1610-742X (electronic)
ISBN 978-3-319-07487-0 ISBN 978-3-319-07488-7 (eBook)
DOI 10.1007/978-3-319-07488-7
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014941083

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Foreword

Robotics is undergoing a major transformation in scope and dimension. From a largely dominant industrial focus, robotics is rapidly expanding into human environments and vigorously engaged in its new challenges. Interacting with, assisting, serving, and exploring with humans, the emerging robots will increasingly touch people and their lives.

Beyond its impact on physical robots, the body of knowledge robotics has produced is revealing a much wider range of applications reaching across diverse research areas and scientific disciplines, such as: biomechanics, haptics, neurosciences, virtual simulation, animation, surgery, and sensor networks among others. In return, the challenges of the new emerging areas are proving an abundant source of stimulation and insights for the field of robotics. It is indeed at the intersection of disciplines that the most striking advances happen.

The *Springer Tracts in Advanced Robotics (STAR)* is devoted to bringing to the research community the latest advances in the robotics field on the basis of their significance and quality. Through a wide and timely dissemination of critical research developments in robotics, our objective with this series is to promote more exchanges and collaborations among the researchers in the community and contribute to further advancements in this rapidly growing field.

The Ninth edition of *Field and Service Robotics* edited by Luis Mejias, Peter Corke and Jonathan Roberts offers in its ten-part volume a collection of a broad range of topics ranging from fundamental concepts such as control, vision, mapping and recognition to advanced applications such as autonomous underwater vehicles, unmanned aerial vehicles, space robots, outdoor driving, search and rescue robots, humanoids and agriculture robots. The contents of the thirty-six contributions represent a cross-section of the current state of robotics research from one particular aspect: field and service applications, and how they reflect on the theoretical basis of subsequent developments. Pursuing technologies aimed at non-factory robots, typically mobile, that must operate in complex and dynamic environments is the big challenge running throughout this focused collection.

Rich by topics and authoritative contributors, FSR culminates with this unique reference on the current developments and new directions in field and service robotics. A fine addition to the series!

Naples, Italy
March 2014

Bruno Siciliano
STAR Editor

Preface

The Field and Service Robotics (FSR) conference is a single track conference with a specific focus on field and service applications of robotics technology. The goal of FSR is to report and encourage the development of field and service robotics. These are non-factory robots, typically mobile, that must operate in complex and dynamic environments. Typical field robotics applications include mining, agriculture, building and construction, forestry, cargo handling and so on. Field robots may operate on the ground (of Earth or planets), under the ground, underwater, in the air or in space. Service robots are those that work closely with humans, importantly the elderly and sick, to help them with their lives.

The first FSR conference was held in Canberra, Australia, in 1997. Since then the meeting has been held every 2 years in Asia, America, Europe and Australia. It has been held in Canberra, Australia (1997), Pittsburgh, USA (1999), Helsinki, Finland (2001), Mount Fuji, Japan (2003), Port Douglas, Australia (2005), Chamonix, France (2007), Cambridge, USA (2009), Sendai, Japan (2012) and most recently in Brisbane, Australia (2013).

This year we had 54 submissions of which 36 were selected for oral presentation. The organisers would like to thank the international committee for their invaluable contribution in the review process ensuring the overall quality of contributions. The organising committee would also like to thank Ben Upcroft, Felipe Gonzalez and Aaron McFadyen for helping with the organisation and proceedings.

The conference was sponsored by the Australian Robotics and Automation Association (ARAA), CSIRO, Queensland University of Technology (QUT), Defence Science and Technology Organisation Australia (DSTO) and the Rio Tinto Centre for Mine Automation, University of Sydney.

January 14

Luis Mejias (QUT)
Peter Corke (QUT)
Jonathan Roberts (CSIRO)

Organization

International Program Committee

Tim Barfoot	U Toronto, Canada
Peter Corke	QUT, Australia
Henrik Christensen	Georgia Tech, USA
Aarne Halme	Helsinki U of Tech, Finland
Karl Iagnemma	MIT, USA
Alonzo Kelly	CMU, USA
Simon Lacroix	LAAS, France
Christian Laugier	INRIA, France
John Leonard	MIT, USA
David P. Miller	U Oklahoma, USA
Keiji Nagatani	Tohoku U, Japan
Cedric Pradalier	Georgia Tech, France
Jonathan Roberts	CSIRO, Australia
Daniela Rus	MIT, USA
Miguel Angel Salichs	U Carlos III, Spain
Sanjiv Singh	CMU, USA
Gaurav Sukhatme	USC, USA
Satoshi Tadokoro	Tohoku U, Japan
Takashi Tsubouchi	U Tsukuba, Japan
David Wettergreen	CMU, USA
Kazuya Yoshida	Tohoku U, Japan
Arto Visala	Helsinki U of Tech, Finland
Alex Zelinsky	DSTO, Australia
Uwe Zimmer	ANU, Australia

Contents

Part I: Autonomous Underwater Vehicles

Part II: Outdoor Driving

Part III: Unmanned Aerial Vehicles

Part IV: Control

Part V: Humanoid and Space

Part VI: Mapping and Recognition

Part VII: Vision

Part VIII: Domestic Robots

Part IX: Agriculture Robots

Part X: Search, Rescue Robots

Part I
Autonomous Underwater Vehicles

Hierarchical Classification in AUV Imagery

M.S. Bewley, N. Nourani-Vatani, D. Rao, B. Douillard,
O. Pizarro, and S.B. Williams

Abstract. In recent years, Autonomous Underwater Vehicles (AUVs) have been used extensively to gather imagery and other environmental data for ocean monitoring. Processing of this vast amount of collected imagery to label content is difficult, expensive and time consuming. Because of this, typically only a small subset of images are labelled, and only at a small number of points. In order to make full use of the raw data returned from the AUV, this labelling process needs to be automated. In this work the single species classification problem of [1] is extended to a multi-species classification problem following a taxonomical hierarchy. We demonstrate the application of techniques used in areas such as computer vision, text classification and medical diagnosis to the supervised hierarchical classification of benthic images. After making a comparison to flat multi-class classification, we also discuss critical aspects such as training topology and various prediction and scoring methodologies. An interesting aspect of the presented work is that the ground truth labels are sparse and incomplete, i.e. not all labels go to the leaf node, which brings with it other interesting challenges. We find that the best classification results are obtained using Local Binary Patterns (LBP), training a network of binary classifiers with probabilistic output, and applying "one-vs-rest" classification at each level of the hierarchy for prediction. This work presents a working solution that allows AUV images to be automatically labelled with the most appropriate node in a hierarchy of 19 biological groupings and morphologies. The result is that the output of the AUV system can include a semantic map using the taxonomy prescribed by marine scientists. This has the potential to not only reduce the manual labelling workload, but also to

M.S. Bewley · N. Nourani-Vatani · D. Rao · B. Douillard ·
O. Pizarro · S.B. Williams
Australian Centre for Field Robotics The University of Sydney,
NSW 2050, Australia
e-mail: m.bewley@acfr.usyd.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_1, © Springer International Publishing Switzerland 2015

reduce the current dependence that marine scientists have on extrapolating information from a relatively small number of sparsely labelled points.

1 Introduction

Autonomous Underwater Vehicles (AUVs) have made a significant impact on areas of marine science that require an understanding of the sea floor [2, 3, 4]. Common practice is shifting from using human divers with hand-held cameras, to sending AUVs equipped with stereo cameras and other sensors to capture benthic images. These AUVs are capable of capturing far more data, both in the type (various sensor modalities) and volume (a several hour mission can collect tens of thousands of spatially registered stereo image pairs). In addition, the imagery is geo-referenced far more precisely, and data can be gathered from beyond diver depths. While using an AUV improves the raw data in many ways, the sheer volume of it introduces a new problem for interpretation.

For marine scientists studying the location, distribution and coverage of benthic organisms and morphology, the state of the art is to take a small subset of images from an AUV survey, and manually label the content. For the data set used in this paper [5], biological species and physical formations under 50 randomly selected pixels from every 100^{th} image were labelled. The scientists then extrapolate from this subset, to make inferences about the ecosystems and populations in the geographic area. Figure 5 shows an example image from our data set (described in Section 3) where 50 randomly selected pixels have been labelled using the CPCe labelling software [6].

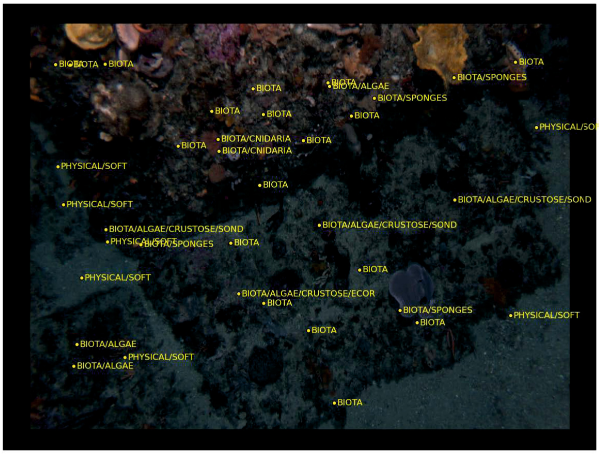


Fig. 1 Example AUV image, with CPC point labels

The ultimate goal of research in this field is to automate the labelling process, such that the vehicle returns a semantic map of the environment, rather than simply providing the raw visual and sensory data. From a robotics perspective, this requires the robotic system to take on the translation and communication effort with which the human users are currently burdened. Recent work [1] describes a solution to a simplified problem of automatically classifying a single species (*Ecklonia Radiata* commonly known as “Kelp”), based on supervised learning algorithms applied to a large AUV dataset. In this paper, we extend the problem to multiple species and physical morphologies, using the same data set.

2 Hierarchical Classification

In order to perform classification on a large set of benthic classes, a different approach is required. Supervised machine learning is typically performed on binary problems (such as presence/absence detection) [7, 3], or multi-class problems where there are a set of strict alternatives [8, 9]. In this study, we examine solutions to the more complex problem of predicting a hierarchy of classes, which is the semantic output that is desired by the marine science community that uses this data set (see Section 3 for details). The contribution of this paper is to demonstrate and evaluate the techniques used in other areas (computer vision, text classification and medical diagnosis) applied to hierarchical classification in benthic images.

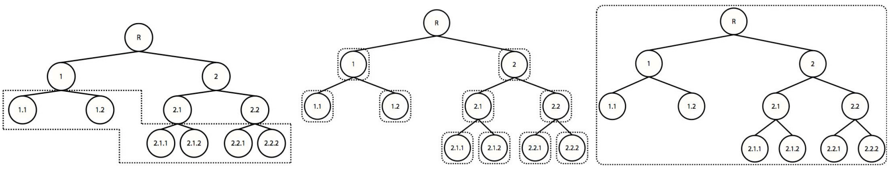
2.1 Classification

In a recent survey, Silla Jr and Freitas [10] extensively reviewed the problem of supervised hierarchical classification and defined a range of methods of dealing with the hierarchy of the data. They defined three main approaches:

1. Ignore the hierarchy and perform flat classification (Figure 2a)
2. Use a network of several local binary classifiers for various regions of the hierarchy (Figure 2b)
3. Use a single classifier but encode the hierarchical structure somehow in the data (Figure 2c).

From the list of methodologies presented, we employ the *Local Classifier per Node* approach of Figure 2b. In this approach, which is by far the most used in the literature [10], each node in the classification tree has a binary classifier that is trained to distinguish that class from others. This approach is relatively straightforward to implement, but has several advantages:

- It is possible to either output the single node in the hierarchy which represents the best prediction, or to query the probability of a given image point being a member of any given node.



(a) Flat classifier (b) Local classifiers (c) Global classifier

Fig. 2 Classification approaches for a hierarchical data set. The dashed lines show the extent of each classifier. In the flat (2a) and global (2c) approaches a single classifier is employed whereas in the local approach (2b) several classifiers are used; the shown tree is the *Local Classifier per Node* approach. Figures taken from [10].

- It allows use of the full data set, including the points which have not been labelled to a leaf node (e.g. as simply “biological”, or “algae” rather than specific species).
- It allows different features, classification algorithms and even training sets to be used for each classification sub-problem. Intuitively, it makes sense that the features which distinguish between various species of algae would differ from those that distinguish sand from rock.

2.2 Training Policies

In training a network of binary classifiers, it is necessary to decide which examples to use as the training set for each node, and how feature extraction is performed at each node. We compare the two policies described in [10] that most naturally fit our problem: the *inclusive* and the *sibling* policies.

2.2.1 The Inclusive Policy

This policy includes as positive examples the entire subtree of the training node. The nodes in the rest of the tree are used as negative examples, with the exception of direct ancestors of the training node. The direct ancestors cannot be used as either negative or positive examples as they contain instances of both; e.g. if training a classifier for node *2.1* in Figure 2b, then nodes *2.1.1* and *2.1.2* are used as positive examples and all other nodes with the exception of the ancestor nodes *R* and *2* are used as negative example (nodes *1*, *1.1*, *1.2*, *2.2*, *2.2.1* and *2.2.2*).

2.2.2 The Sibling Policy

This alternative policy uses the same positive training examples, however, the negative examples are restricted to siblings of the training node (and not siblings of the ancestor nodes). Using the example from before, the negative training samples will come from nodes *2.2*, *2.2.1* and *2.2.2* only.

The expected performance difference between these two policies is not obvious, with no consistent winner found in [11]. On one hand, the *inclusive policy* ensures that each node is as informed as possible, and should be able to deal better with classifying instances that belong elsewhere in the tree (as it has seen examples of all the data). On the other hand, the *siblings policy* solves a much more specific problem (distinguishing a node's class from its siblings), and may give better discriminative performance between these classes. Such nodes will, however, be less informed about instances that belong elsewhere in the tree. An inherent advantage of the siblings approach is that far less training data is required for nodes deeper in the tree, which becomes significant when the tree is large.

2.3 Prediction

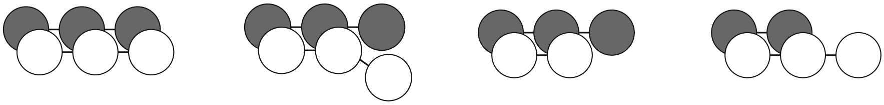
After training the network of classifiers, the decision remains of how to predict the class of a given image point. As we require complete consistency in the hierarchical labels (such that there is a single, unbroken chain of classifiers predicting positive results from the root to the deepest node), the simplest choice described in [10] is chosen, which we refer to as maximum probability switching (MPS). An instance starts at the root node, and flows to the leaf node with the highest prediction probability (akin to performing one-vs-rest classification at each node). This technique forces prediction down to the leaf node level. We can remove this constraint by stopping a prediction from moving further down the tree when the maximum predicted probability falls below some threshold (say 0.5 for the typical cut-off of a binary classifier).

We also test an alternative approach: the use of a simple probabilistic graphical model (PGM). Here the class tree also represents the independence relations in the PGM, and we assume the conditional probability of node membership is given by the probabilistic predictions of the classifiers[1]. This allows exact Bayesian inference to be performed trivially, by multiplying probabilities of a node's ancestors to obtain the net probability of membership.

2.4 Performance Measure

Lastly, a robust performance metric is needed; ideally a single number to evaluate the performance on an entire tree. The closest performance measure commonly used in the literature is the hierarchical f1-score [12]. Each instance has multiple counts of true/false positives/negatives, as each node in the chain

[1] This assumption is at least reasonable for the *siblings* training policy, as the training sets only include those instances which we know belong to the parent node. The output of the classifiers therefore represents e.g. "the probability that this instance is *Algae*, given we know it is an instance of *Biota*".



(a) True prediction: (TP, TP, TP)

(b) False prediction: (TP, TP, FP, FN)

(c) False prediction due to thresholding: (TP, TP, FN)

(d) Modified prediction: (TP, TP)

Fig. 3 The hierarchical f1-score performance metric. The connected circles represent classifier nodes, with the root on the left, and leaf on the right. Grey circles are ground truth, white circles are prediction. Figure 3a shows the case of the classifier network correctly predicting a node of depth 3 in the hierarchy. Figure 3b represents the first two levels being predicted correctly, whereas the classifier network incorrectly chooses a sibling of the true class at depth 3.

of true class nodes is compared to the chain of predicted nodes. Figures 3a and 3b illustrate two scenarios. In the former case, the prediction down the entire tree is accurate. In the latter case the last prediction is incorrect. This results in a False Negative (FN) detection as the correct node was missed as well as a False Positive (FP) detection for an incorrect node being detected.

When employing thresholded MPS it is possible that the prediction stops earlier up in the tree than the ground truth. This occurs when the child classifier prediction probability is below the set threshold. In such an instance, the missed nodes are calculated as FNs. This is illustrated in Figure 3c.

Finally, there can be cases where the predicted class is more specific (lower down the tree) than the ground truth class. This can occur when the ground truth label is not at the leaf level. In such an instance the hierarchical f1-score commonly used in the literature would penalise the prediction with a FP. However, there are instances where this is not fair; e.g. the classifier predicts “Labrador” while the label says “Dog”. “Labrador ” is of course a breed of “Dog”, and may be a more accurate description. This is true for our labels of the underwater species, where the labeller does not always label the instance to the leaf-node level. This can occur due to poor image quality, inability of the labeller to recognise the species or cost and effort constraints if more detailed labels are not required for the given research project.

We therefore modify the metric such that if the predicted class is more specific than the ground truth (Figure 3d), we do not reward or penalise any results deeper than the deepest known class.

2.5 Descriptors

A subtlety of the *Local Classifier per Node* approach is in the selection of image features. In flat multi-class classification (Figure 2a) and global classification

(Figure 2c), the same image features are typically used for the nodes. With the local classifiers (Figure 2b), we can select (either manually, or by feature learning techniques) features that are optimised per classifier node for the *siblings* or *inclusive* training data sets. In this paper we employ various features that both optimise for the local node or are constant across the nodes.

We analyse the performance of Principle Component Analysis (PCA), Local Binary Patterns (LBP) texture descriptors, and feature learning (FL). In all instances, the features are extracted from localised image patches around the ground truth label.

As discussed in [1], the AUV travels at an approximately constant height of 2m above the sea floor. We therefore use the same assumption of pixels in the image representing a fixed scale. In the 1360 × 1024 RGB images, 100 pixels corresponds to approximately 10 centimetres.

2.5.1 Principal Component Analysis

The PCA descriptors are calculated on each of 7, 15, 31, 63 and 95 pixel square RGB image patches. The patches are whitened and empirical study showed that keeping 60 components is sufficient.

2.5.2 Local Binary Patterns

We investigate two varieties of the popular LBP descriptors, namely the rotation-invariant uniform descriptor [13] and the histogram Fourier transform descriptor [14]. The descriptors are calculated from a 31×31 pixel patch and normalised using various image normalisation methods.

2.5.3 Feature Learning

Feature learning uses K-means clustering to learn a dictionary of 1000 patches for both 7 and 15 pixel square images. The CIFAR-10 Images Dataset [15] was used to generate the features, instead of the Tasmanian AUV image data. Compared to the AUV images, the CIFAR-10 data set has more variation in the content, and is therefore able to produce a more diverse dictionary of learned features. The features are encoded using the L2 distance to each patch in the dictionary [16]. The centroids, or learned features, tend to resemble Gabor-like edge or texture filters, as shown in Figure 4.

3 Data Set and Hierarchy

The Tasmania Coral Point Count data set is comprised of 14 separate dive missions conducted by the AUV *Sirius* off the South-East coast of Tasmania, Australia, in October 2008. From the data set containing over 100,000 stereo pairs of images, marine scientists at the University of Tasmania [5] selected every 100$^{\text{th}}$ colour image and used the CPCe software package [6] to label

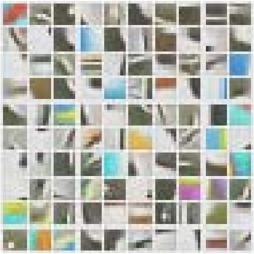


Fig. 4 A subset of the 1000 features learned from 7×7 pixel patches in the CIFAR-10 Dataset

50 random points on each [17]. This has resulted in 62,900 labels from 1258 images. Figure 5 illustrates some images from the data set[2], with original and predicted labels.

A wide range of class labels were used, indicating biological species (including types of sponge, coral, algae etc.), abiotic elements (types of sand, gravel, rock, shells etc.), and types of unknown data (ambiguous species, poor image quality, etc.). Precise details of the labelling methodology can be found in [17].

Recently, a standardised tree structure for biological and physical classes for underwater species was defined as part of the Catami project [18]. By mapping the data set to this classification hierarchy, a complex tree of 19 classes was obtained (See Figure 7). Note that we have merged some of the labels under their ancestor nodes. These labels corresponded to species with very few instances, because they belonged to an unknown species or a mixture class. Despite this consolidation, the class instances remain highly unbalanced; e.g. the “SOFT” class has 6139 instances while the “BROWN” erect-branching algae has only 10 instances.

4 Results and Discussion

4.1 Hierarchical Classification Approaches

We present results using logistic regression (LR) classifiers, with features derived using PCA, LBP texture features, and feature learning (FL), and compare the *sibling* and *inclusive* policies.

Performance is measured in terms of the modified hierarchical f1-score with the same training and validation sets described in [1]. In brief, 2/3 of the data is used for training and 1/3 used for testing. Three-fold cross validation within the training set is used for estimation of the Logistic Regression cost parameter, C, in the interval [0.125-128].

[2] The data set is available from `http://marine.acfr.usyd.edu.au/datasets`

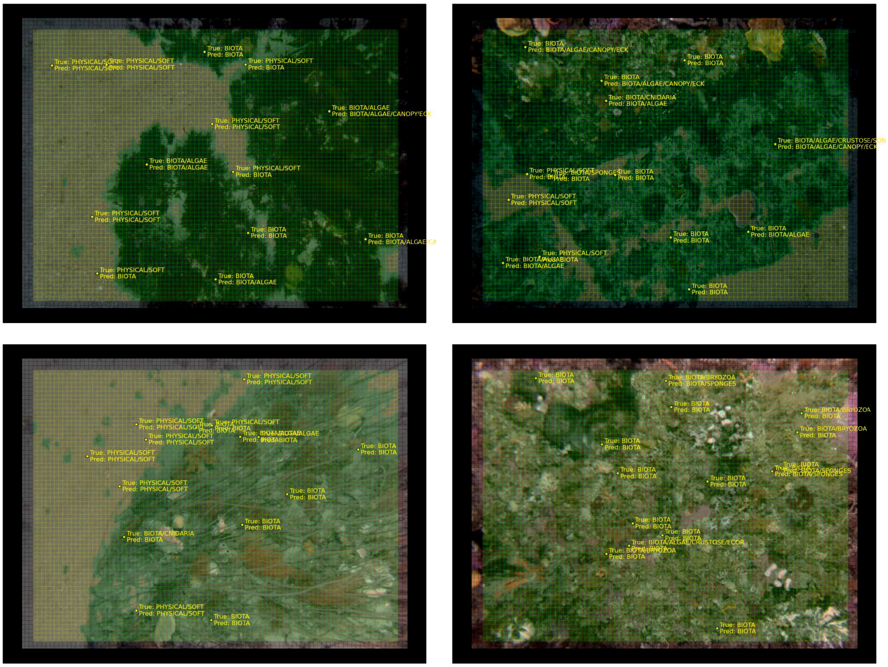

True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: BIOTA Pred: BIOTA
True: PHYSICAL/SOFT Pred: BIOTA
True: BIOTA/ALGAE Pred: BIOTA/ALGAE/CANOPY/ECK
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: BIOTA/ALGAE Pred: BIOTA/ALGAE
True: PHYSICAL/SOFT Pred: BIOTA
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: BIOTA Pred: BIOTA
True: BIOTA Pred: BIOTA/ALGAE
True: PHYSICAL/SOFT Pred: BIOTA
True: BIOTA Pred: BIOTA/ALGAE
True: BIOTA Pred: BIOTA/ALGAE/CANOPY/ECK
True: BIOTA Pred: BIOTA
True: BIOTA Pred: BIOTA/ALGAE/CANOPY/ECK
True: BIOTA/CNIDARIA Pred: BIOTA/ALGAE
True: BIOTA/ALGAE/CRUSTOSE/SON Pred: BIOTA/ALGAE/CANOPY/ECK
True: PHYSICAL/SOFT Pred: BIOTA
True: BIOTA/SPONGES Pred: BIOTA
True: BIOTA Pred: BIOTA
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: BIOTA Pred: BIOTA
True: BIOTA Pred: BIOTA/ALGAE
True: PHYSICAL/SOFT Pred: BIOTA
True: BIOTA/ALGAE Pred: BIOTA/ALGAE
True: BIOTA Pred: BIOTA
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: BIOTA Pred: BIOTA
True: PHYSICAL/SOFT Pred: BIOTA
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: BIOTA/ALGAE Pred: BIOTA
True: BIOTA Pred: BIOTA
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: BIOTA Pred: BIOTA
True: BIOTA Pred: BIOTA
True: BIOTA/CNIDARIA Pred: BIOTA
True: PHYSICAL/SOFT Pred: PHYSICAL/SOFT
True: BIOTA Pred: BIOTA
True: BIOTA Pred: BIOTA
True: BIOTA/BRYOZOA Pred: BIOTA/SPONGES
True: BIOTA Pred: BIOTA
True: BIOTA/BRYOZOA Pred: BIOTA
True: BIOTA/BRYOZOA Pred: BIOTA
True: BIOTA Pred: BIOTA
True: BIOTA Pred: BIOTA
True: BIOTA Pred: BIOTA/SPONGES
True: BIOTA Pred: BIOTA/SPONGES
True: BIOTA Pred: BIOTA
True: BIOTA Pred: BIOTA
True: BIOTA Pred: BIOTA
True: BIOTA/ALGAE/CRUSTOSE/ECOR Pred: BIOTA
True: BIOTA/BRYOZOA Pred: BIOTA
True: BIOTA Pred: BIOTA

Fig. 5 Example AUV images with both true and predicted CPC labels from the validation set. In addition, the top level hierarchy choice has been evaluated at a large number of points, to show where the algorithm predicts BIOTA (green) and PHYSICAL (yellow). The predictions are based on the classifier network using PCA on 63×63 image patches, with the inclusive training policy, and MPS prediction with a threshold of 0.5.

In Figure 6, performance using the modified hierarchical f1-score is compared for the two policies across a range of image descriptors and prediction approaches.

As can be observed, there is a clear trend that for mandatory leaf node prediction, the PGM is generally superior to MPS. This is promising for future work, as the PGM is the more principled approach, and more sophisticated models can be used. Also, the modification to permit the network to predict only higher level classes when less confident (using thresholding) was highly successful at improving the performance.

In terms of *inclusive* and *sibling* policies, we obtain the same finding on underwater images as that found using text classification [11]—no clear winner. Given the *sibling* policy has a significant advantage in reducing training time, it is preferred in situations where the results are comparable.

The best result was accomplished using the LBP descriptor, the *inclusive* training methodology and thresholded MPS prediction with a threshold of

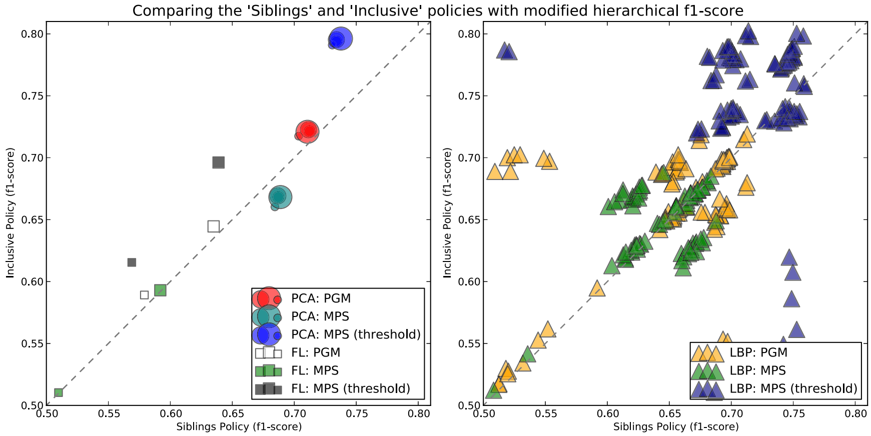


Fig. 6 Comparison of *Siblings* and *Inclusive* policies using the modified hierarchical f1-score. Each marker represents both *sibling* and *inclusive* performance on a given feature and prediction setup. Marker size is scaled by the image patch size used.

0.5. This resulted in a tree f1-score of 80.2%. The detailed performances are shown in Figure 7. We notice that the performance at the highest level of the hierarchy (differentiating between biological and physical) is in the mid 80's. The result is also satisfactory, compared to [1]. However, for nodes without a significant number of training instances, performance was very poor.

4.2 Hierarchical vs Flat Multi-class Classification

In order to compare the approach of the hierarchical binary network and traditional flat multi-class classification, it was necessary to restructure the problem. For the hierarchical approach, MPS was used to force predictions to leaf node level. The flat classifier was comprised of the 13 leaf nodes in the tree (selected as per Figure 2a). The test set was reduced to contain only data that had been labelled to leaf node (as higher level labels have no defined assignment in a leaf node flat classifier). Rather than the tree f1-score (which also makes little sense for a flat classifier), results were evaluated using the mean f1 score across the leaf node classifiers. As the mean f1-score weights nodes equally, the overall results were significantly lower than the tree f1-score. On this metric, the hierarchical classifier's performance was marginally higher. This was largely due to superior performance on the most poorly performing nodes (e.g. the f1-score on Cnidaria reduced from 0.09 to 0.00) with the flat multi-class case). Note that this comparison was performed using the *siblings* policy, and the best performing descriptors (Fourier LBP).

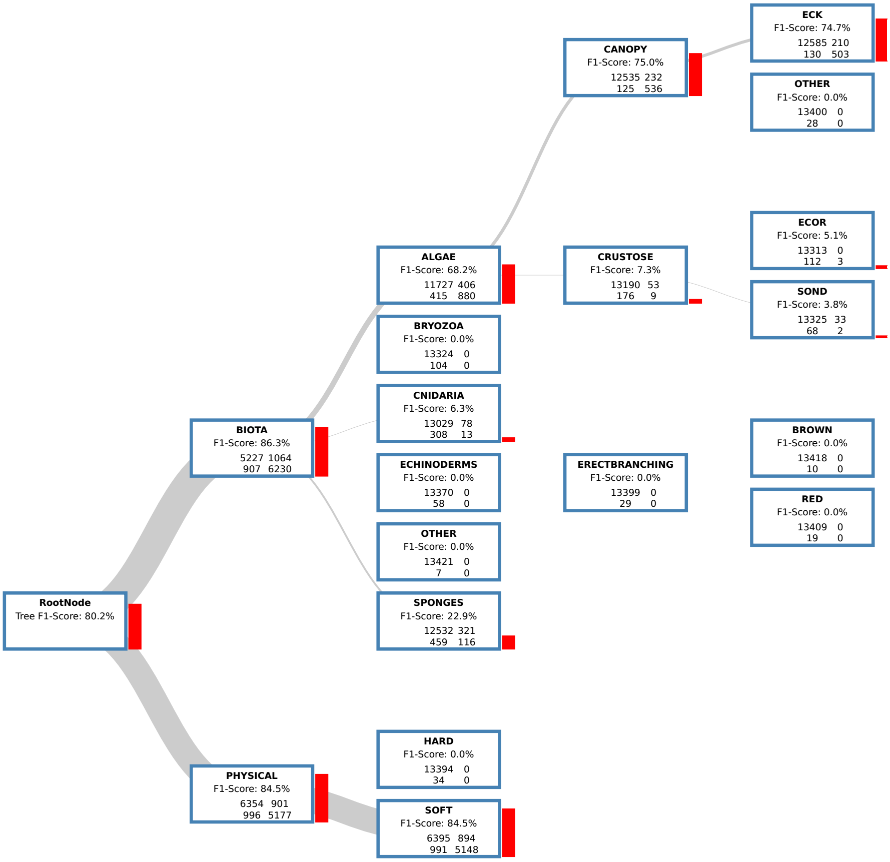


Fig. 7 Performance results on best classifier (HLNP with thresholding, LBP features, *inclusive* training set). The grid of 4 numbers is the confusion matrix for instances in the test set, which was used to compute both the local f1-scores for each node (red bars), and the tree f1-score (given in the RootNode box, and calculated by computing the f1-score on the sum over the local node confusion matrices). The thickness of the grey edges between nodes is proportional to the number of instances from the test set that belong to a given node.

Table 1 Flat multi-class and hierarchical classification comparison, tested on the reduced leaf-node only data set

	Mean f1-score
Hierarchical (trained on leaf-node training data)	0.197
Hierarchical (trained on all training data)	0.182
Flat multi-class (trained on leaf-node training data)	0.178

5 Conclusion

We have investigated various aspects of performing supervised hierarchical classification on sparsely labelled benthic imagery for the purpose of species recognition. The aim was to apply techniques used in the literature from other fields, to construct an initial solution for the automated interpretation of AUV images.

Results have shown over a range of feature descriptors and patch sizes, that with the PGM prediction, better results were obtained than using simple MPS. This is promising for future directions as it is a more principled approach. However, the best results were achieved employing thresholded MPS.

We also compared two different classifier training approaches, namely the local *sibling* policy against the global *inclusive* policy. It was demonstrated that comparatively, the *sibling* and *inclusive* training policies exhibit similar performance, with the *sibling* option holding the advantage due to reduced training time.

In addition, the comparison with a flat multi-class approach on the reduced data set confirms that basic performance at leaf-node level is at least as good as the traditional approach, using our hierarchical classification scheme.

Future work will cover a number of areas. In terms of the hierarchical classification, we will investigate potential improvements, such as the use of PGMs with variable depth prediction. Because the classification scheme permits different features to be used at different nodes, another challenging area of research will be to find ways of incorporating other sensor modalities from the AUV (such as dense stereo information) to enhance the ability of the classifier to distinguish between various species and objects. Although the automated semantic labelling described in this paper has been applied as a post-processing step, eventual incorporation as a real time algorithm onboard the robot would have further benefits. Communication with the AUV through water is typically performed using an acoustic modem, which has far lower bandwidth than what is necessary to transmit raw image and sensor data. If the robot can "understand" what it sees by assigning automated labels, it could use that information to either adapt its behaviour, or relay it to the human operators for monitoring and intervention.

Acknowledgements. The data set used in this paper was collected under the IMOS program [19], by the Marine Robotics group at the Australian Centre for Field Robotics (ACFR). Performing the supervised learning would not have been possible without the extensive manual labelling effort undertaken by marine scientists at the University of Tasmania[3]. Data was sourced from the NERP Marine Biodiversity Hub - the Marine Biodiversity Hub is supported through funding from the Australian Government's National Environmental Research Program (NERP),

[3] Barret, N.S. and Hill, N.A., Institute for Marine and Antarctic Studies, University of Tasmania

administered by the Department of Sustainability, Environment, Water, Population and Communities (DSEWPaC). The work of Bertrand Douillard was supported by Australian Postdoctoral Discovery grant DP110101986.

References

1. Bewley, M., Douillard, B., Nourani-Vatani, N., Friedman, A., Pizarro, O., Williams, S.: Automated species detection: An experimental approach to kelp detection from sea-floor AUV images (December 2012)
2. Williams, S., Pizarro, O., Jakuba, M., Johnson, C., Barrett, N., Babcock, R., Kendrick, G., Steinberg, P., Heyward, A., Doherty, P., Mahon, I., Johnson-Roberson, M., Steinberg, D., Friedman, A.: Monitoring of benthic reference sites: Using an autonomous underwater vehicle. IEEE Robotics and Automation Magazine 19(1), 73–84 (2012)
3. Smith, D., Dunbabin, M.: Automated counting of the northern pacific sea star in the derwent using shape recognition. In: The Australasian Conference on Robotics & Automation (ACRA), pp. 500–507 (December 2007)
4. Steinberg, D., Friedman, A., Pizarro, O., Williams, S.: A bayesian nonparametric approach to clustering data from underwater robotic surveys. In: International Symposium on Robotics Research (ISRR) (August 2011)
5. National environmental research program (NERP) marine biodiversity hub, scored autonomous underwater vehicle imagery, east tasmania 2008-2010 (2010), `http://marine.acfr.usyd.edu.au/datasets`
6. Kohler, K.E., Gill, S.M.: Coral point count with excel extensions (CPCe): a visual basic program for the determination of coral and substrate coverage using random point count methodology. Computers & Geosciences 32(9), 1259–1269 (2006), `http://www.sciencedirect.com/science/article/pii/S0098300405002633`
7. Denuelle, A., Dunbabin, M.: Kelp detection in highly dynamic environments using texture recognition. In: The Australasian Conference on Robotics & Automation (ACRA) (December 2010)
8. Beijbom, O., Edmunds, P., Kline, D., Mitchell, B., Kriegman, D.: Automated annotation of coral reef survey images. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), p. 1170 (2012)
9. Mehta, A., Ribeiro, E., Gilner, J., van Woesik, R.: Coral reef texture classification using support vector machines. In: VISAPP (2), 302–310 (2007)
10. Silla Jr., C.N., Freitas, A.A.: A survey of hierarchical classification across different application domains. Data Mining and Knowledge Discovery 22(1-2), 31–72 (2011)
11. Fagni, T., Sebastiani, F.: On the selection of negative examples for hierarchical text categorization. In: Proceedings of the 3rd Language & Technology Conference (LTC 2007), pp. 24–28 (2007)
12. Kiritchenko, S., Matwin, S., Nock, R., Famili, A.F.: Learning and evaluation in the presence of class hierarchies: application to text categorization. In: Lamontagne, L., Marchand, M. (eds.) Canadian AI 2006. LNCS (LNAI), vol. 4013, pp. 395–406. Springer, Heidelberg (2006)
13. Ojala, T., Pietikaeinen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(7), 971–987 (2002)

14. Ahonen, T., Matas, J., He, C., Pietikäinen, M.: Rotation invariant image description with local binary pattern histogram fourier features. In: Salberg, A.-B., Hardeberg, J.Y., Jenssen, R. (eds.) SCIA 2009. LNCS, vol. 5575, pp. 61–70. Springer, Heidelberg (2009)
15. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto (2009)
16. Coates, A., Lee, H., Ng, A.Y.: An analysis of single-layer networks in unsupervised feature learning. In: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS), vol. 15 (2011)
17. Barrett, N., Meyer, L., Hill, N., Walsh, P.: Methods for the processing and scoring of AUV digital imagery from south eastern tasmania. University of Tasmania, Technical Report 11560 (August 2011), `http://eprints.utas.edu.au/11560/`
18. Edwards, L.: Release of CATAMI classification scheme (February 2013), `http://catami-australia.blogspot.com.au/`
19. IMOS: integrated marine observing system (September 2013), `http://www.imos.org.au`

Mapping 3D Underwater Environments with Smoothed Submaps

Mark VanMiddlesworth, Michael Kaess, Franz Hover, and John J. Leonard

Abstract. This paper presents a technique for improved mapping of complex underwater environments. Autonomous underwater vehicles (AUVs) are becoming valuable tools for inspection of underwater infrastructure, and can create 3D maps of their environment using high-frequency profiling sonar. However, the quality of these maps is limited by the drift in the vehicle's navigation system. We have developed a technique for simultaneous localization and mapping (SLAM) by aligning point clouds gathered over a short time scale using the iterative closest point (ICP) algorithm. To improve alignment, we have developed a system for smoothing these "submaps" and removing outliers. We integrate the constraints from submap alignment into a 6-DOF pose graph, which is optimized to estimate the full vehicle trajectory over the duration of the inspection task. We present real-world results using the Bluefin Hovering AUV, as well as analysis of a synthetic data set.

1 Introduction

Inspection of underwater infrastructure is currently a costly, time-consuming, and dangerous task performed manually by human divers. As autonomous underwater vehicles become more sophisticated, it will be increasingly feasible and desirable to automate these inspection tasks. However, there are many technical challenges that

Mark VanMiddlesworth · John J. Leonard
Computer Science and Artificial Intelligence Laboratory (CSAIL),
Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, USA
e-mail: {mvanmidd,jleonard}@mit.edu

Michael Kaess
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
e-mail: kaess@cmu.edu

Franz Hover
Department of Mechanical Engineering, MIT , Cambridge, MA 02139, USA
e-mail: hover@mit.edu

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_2, © Springer International Publishing Switzerland 2015

must still be overcome, particularly in the domain of localization and navigation in complex 3D environments.

Underwater localization is particularly difficult because globally-referenced satellite navigation such as GPS is rapidly attenuated in water. Underwater vehicles must therefore either rely on inertial measurements, observations of local features, or acoustic communication with a globally-referenced transponder for localization. Current AUVs, such as those used for seafloor photographic and bathymetric surveys, generally use a combination of these techniques.

Inspection of harbors, platforms, and other underwater infrastructure poses unique navigational challenges beyond those of a seafloor survey. Range-based acoustic localization such as LBL is subject to multi-path interference, which is exacerbated by the shallow depths and hard, flat walls of harbor environments. Additionally, large metallic objects such as ship hulls render magnetic compasses largely ineffective, requiring that heading be estimated with drift-prone inertial sensors. Finally, while seafloor surveys can often be performed using inertial navigation and corrected in post-processing, the collision hazards posed by underwater infrastructure require accurate navigation in real time.

2 Related Work

This paper builds upon a large body of prior research in underwater simultaneous localization and mapping (SLAM), 3D mapping, and dense point cloud alignment. The goal of SLAM is to correct for drift in the vehicle's dead reckoning by using repeated observations of static landmarks in the environment. There are two broad families of approaches: *filtering* and *smoothing*. Both approaches generally assume Gaussian process and measurement error models.

Filtering approaches track the robot's current pose by incrementally adding dead reckoning and loop closure constraints. Because constraints are added incrementally, this approach is naturally suited to real-time operation. Barkby et al. [2] used a particle filter along with a bathymetric sonar to produce a 2.5D map of the seafloor in real time. The extended Kalman filter (EKF) has been applied to imaging sonar data [16], and forward-looking sonar [11] collected by AUVs. The extended information filter (EIF) [21], in which the normal distribution is parameterized in terms of its information vector and information matrix rather than its mean and covariance, has a sparse structure which enables efficient computation. Walter et al. [22] used a filtering approach to survey an underwater structure using features manually extracted from an imaging sonar.

A disadvantage of filtering approaches is that they estimate only the current vehicle pose. Because information from loop closure constraints is not back-propagated to correct previous pose estimates, these approaches do not provide an accurate estimate of the entire vehicle trajectory. This is particularly problematic when adding constraints from large loop closures, which produces discontinuities in the estimated vehicle path.

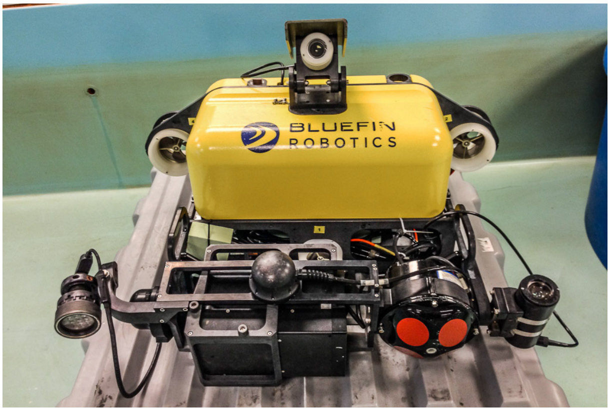


Fig. 1 Bluefin Hovering Autonomous Underwater Vehicle (HAUV) with Soundmetrics DIDSON sonar

Smoothing approaches also include all past poses into the optimization. Exploiting the fact that the information matrix is exactly sparse in view-based SLAM, Eustice et al. [9] applied the information filtering approach to camera data from the RMS Titanic, producing a 6-DOF trajectory estimate. Dellaert and Kaess [7] formulate the SLAM problem as a bipartite factor graph, and provide an efficient solution by smoothing and mapping (SAM). Incremental smoothing and mapping (iSAM) [13] incrementalizes the matrix factorization to efficiently integrate new constraints without re-factoring the information matrix.

In the underwater domain, pose graphs have been shown to produce more consistent maps due to their ability to correct prior navigation error and re-linearize around the corrected trajectory. Beall et al. [3] used an offline pose-graph based smoothing approach to estimate a full 6-DOF trajectory in a large-scale underwater photo survey. Kunz and Singh [14] applied offline pose graph optimization to visual and sonar data. Pose graphs have been used for real-time mapping of a locally planar complex structures such as ship hulls [12].

We chose to use a pose graph formulation for the improved handling of nonlinear error models and large loop closures, as we expect relatively large navigation drift in the absence of a magnetic compass. Additionally, the ability to reconstruct the full vehicle trajectory is particularly important for inspection tasks, to verify that the target has been fully covered by the vehicle's sensors.

In bathymetric and photomosaicing applications, a 2.5-dimensional representation of the environment (depth map) is sufficient, but complex environments require a full 3D representation. Fairfield et al. [10] use evidence grids inside a particle filter to perform real-time 3D mapping of a sinkhole with an imaging sonar.

Submap alignment requires generation of loop closure constraints, which, in visual SLAM, are commonly derived using viewpoint-invariant visual features such as SIFT. However, bathymetric and profiling sonars generally do not produce easily identifiable viewpoint-invariant features. If the vehicle dead reckoning is accurate over short time periods, as is the case with most IMU and DVL based systems, the sonar data can be aggregated into "submaps" for improved matching. The aggregated point cloud data is then treated as a single measurement. Point cloud-based approaches such as [15] and [5], which use iterative closest point (ICP) to align submaps, have been applied to scanning laser data. However, sonar data generally exhibits much higher noise than laser scanners, complicating registration. Prior work in the underwater domain has demonstrated success at aligning bathymetric (2.5D) submaps using cross-correlation [18] and ICP [17] to provide constraints for an EKF-based SLAM system; these are perhaps the most closely related to the work presented here. However, our work differs in a few key areas. Our contributions are as follows:

- A full 3D representation of arbitrarily complex marine environments
- Reconstruction of the entire vehicle trajectory using a pose graph
- A method of smoothing submaps for improved alignment

3 Problem Statement

Our complex area inspection missions begin with a long-range survey, which is used to construct a rough mesh of the inspection target. This mesh is used to plan an inspection path that covers the entire inspection target while avoiding collision. We use the sampling-based technique described in [8] for path generation.

Data for these experiments was collected on the Bluefin Hovering Autonomous Underwater Vehicle (HAUV), a vehicle specifically designed for ship hull inspection (see Fig. 1). Full details of the platform can be found in [12]; here we will briefly summarize the relevant attributes.

3.1 Navigation Sensors

The HAUV is equipped with a Honeywell HG1700 IMU, an RDI 1200kHz DVL, and a Keller pressure sensor. The DVL can be locked downward or rotated to point at the ship hull; in our complex-area operations, we keep the DVL locked downward. The DVL and IMU are combined to provide a position estimate. The x and y position is estimated by integrating velocities from the DVL and IMU, and are therefore subject to long-term drift. Because magnetic compasses are unreliable in the presence of steel structures, we do not use a magnetic compass; therefore, heading ψ is also subject to drift. Depth z, pitch θ, and roll ϕ are all measured directly, so they are subject to measurement noise, but not to drift.

3.2 Sonar

We use the DIDSON profiling sonar, which has an aperture 22° wide and 1° tall. There are 96 beams comprising the width of the sonar, each of which provides acoustic intensity in 512 bins representing range. To extract ranges from these intensities, we perform a median filter to reduce noise, then threshold the intensity and accept the first (shortest-range) return along each beam. This somewhat reduces the effect of multipath interference, which often appears as an echo of close objects at a longer range. The range extraction at time t produces 2-D polar points $p_t^l = (\theta, r)$ in the sonar coordinate frame, which are transformed into Cartesian points $p_t = (x, y, z)$ in the vehicle coordinate frame.

We generally operate the sonar with a minimum range of 1m and maximum range of 6m for close inspection tasks, therefore each of the 512 range bins represents approximately 1cm. In practice, however, the range resolution is more coarse. Even in an ideal environment, measuring a hard, flat surface in a low-noise swimming pool, we observe that the return is generally "smeared" along several bins.

Harbor environments further degrade the quality of sonar data. Reflections from ship hulls, surface waves, and, surprisingly often, large fish lead to spurious returns even with aggressive filtering. From our experience, the error is generally on the order of 5cm, with occasional larger errors up to half a meter.

On the HAUV, the DIDSON is mounted to provide a horizontal "fan" of range returns, and can be swept 90°vertically. Due to the sonar's narrow field of view, we primarily use the fixed configuration for long-range surveys. Close range surveys consist of a series of waypoints, with the vehicle holding station while the sonar is swept vertically.

4 Pose Graph SLAM Using Submap Alignment

Fig. 3.2 shows the high-level architecture of our system. Our task is to estimate the vehicle pose $\mathbf{x}_t = [x, y, z, \psi, \theta, \phi]$ over the entire trajectory of a complex inspection task, $t = [1..n]$. As the vehicle moves along the inspection path, it aggregates sonar pings into submaps. The submaps are stored in the *submap catalog*, and are smoothed and aligned to provide loop closure constraints.

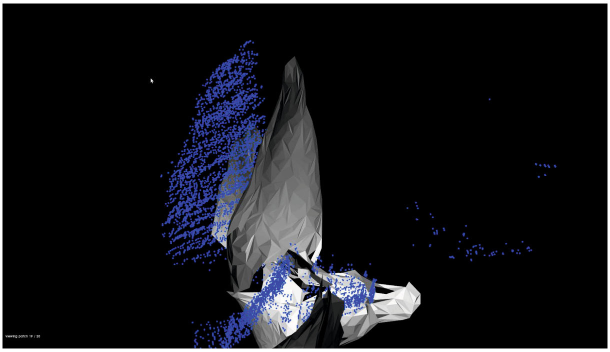


Fig. 2 An unfiltered submap projected alongside the prior mesh using dead reckoning, illustrating the approximate scale of dead reckoning error and some characteristics of unfiltered sonar data

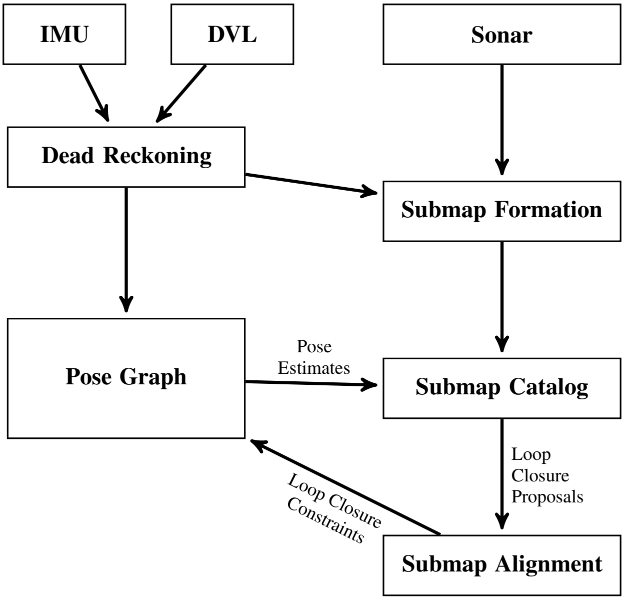


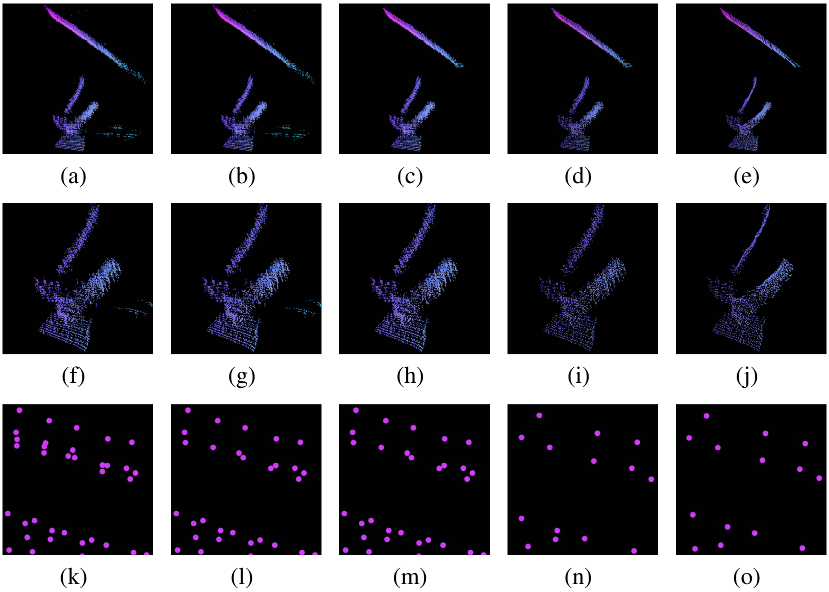
Fig. 3 Architecture of submap-based SLAM

4.1 Submap Formation

To form submaps, we assume that dead reckoning is accurate over a short time scale, and use dead reckoning to aggregate groups of consecutive sonar pings into submaps. Each submap is assigned an *anchor time*, in this case halfway between the start and end time, and is treated as a single measurement taken at the anchor time.

Choosing submap size is a balance between the submaps being large enough to align with one another, but not so large as to incorporate significant navigation drift. With the DIDSON in the sweeping configuration, we simply define a submap as a single vertical sweep. When the DIDSON is locked in the horizontal configuration, we define a minimum number of points k_{sub} and a maximum time window for the submaps t_{sub}; when either threshold is reached, a submap is created. For a list of parameter values used in our experiments, see Table 1.

When the vehicle is actively scanning a target, the points threshold k_{sub} generally triggers submap formation well within the time window t_{sub}. When sonar returns are sparse (e.g. the vehicle is transiting between inspection waypoints), the time limit t_{sub} will trigger submap formation. In our experiments, these sparse submaps often produce poor alignments and are therefore not used for loop closures.


(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

(k) (l) (m) (n) (o)

Fig. 4 A submap as it progresses through the four steps of the smoothing process. This submap consists of a vertical portion of the middle of the propeller and a segment of the hull, viewed from the starboard aft side. From left to right: the raw submap, submap after voxel filter 1, submap after outlier rejection, submap after voxel filter 2, and submap after parametric surface modeling. (a-d) wide view showing outliers. (f-j) detail shows smoothing of propeller blades. (k-o) illustrate the relative point density at each step.

4.2 Submap Smoothing

Before aligning the submaps, we perform a filtering and smoothing operation to reduce the sonar artifacts described above. This serves two purposes: to eliminate spurious returns caused by acoustic reflections, fish, etc., and to achieve a more uniform point density for better alignment. We rely primarily on the implementations found in the freely-available Point Cloud Library [19].

The first step is a voxel filter, in which the submap is divided into cubes of size v_1, and if multiple points occupy the same cube, they are removed and replaced with a single point at their centroid. We choose v_1 to be 2cm, which is approximately the spacing between DIDSON beams at medium range. Thus the first voxel filter serves to remove “redundant” data without significantly reducing the resolution of the submap.

Second, we perform k-nearest-neighbor outlier rejection, in which points are rejected if the average distance to their k_{nn} nearest neighbors is $> \sigma_{nn}$ standard deviations above the mean (for details and implementation, see [20]). This is effective at removing artifacts caused by electrical noise, minor multipath reflections, and small fish (we leave systematic multipath interference and large fish for future work).

The third step is a larger voxel filter of size v_2. The goal of this step is to produce a cloud of roughly uniform density, as the parametric surface modeling in step 4 is sensitive to variations in point density. Therefore, we choose v_2 to be roughly the distance covered between DIDSON frames (5-10 Hz) when the vehicle is moving at speed (.5-1 m/s), giving us $v_2 \approx$ 10cm. This step has the effect of combining adjacent points from within DIDSON frames, so that their spacing more closely matches the spacing between frames.

Finally, the resulting points are smoothed using a local parametric approximation as described in [1]. For each point, a polynomial surface is constructed which minimizes the mean squared error to points within a radius r. The point normal is estimated as the normal to the parametric surface, and the point is projected onto the surface. The surface normal is also stored for each point, as it will be utilized in the alignment step. [1]

After smoothing, the smoothed submap and surface normals are stored in the submap catalog and used for alignment, while the full-resolution unprocessed submap is retained for reprojection into the final map. For an illustration of submaps at each step of the smoothing process, see Fig. 4.

4.3 Submap Alignment

We align submaps using Iterative Closest Point [4], an algorithm for aligning a set of *measurement points* (also called *source points*) to a *target model* by iteratively computing rigid transformations that minimize the sum of squared distances between the points and target model. Each measurement point is represented in Cartesian coordinates as $p = [x, y, z]$, and the target can be any model that allows computation of distances to a point, such as a parametric surface, line, or another point cloud. In our case, we have normal estimates for each submap, so we use point-to-plane distances in computing the transform.

For measurement points p'_i corresponding to target points p_i with normals n_i in the target, ICP computes the rigid transform T that minimizes the sum of square errors between the measurement set and the target set:

$$\sum_i ||(p'_i - Tp_i) \cdot n_i||^2 \tag{1}$$

When a suitable alignment is found, we transform the anchor pose of the source submap, and formulate a relative constraint between the source and target anchor poses. This constraint is added as a factor in the pose graph.

[1] While the previous steps were fairly well-grounded in sonar geometry and error characteristics, it is not immediately obvious why step 4, parametric surface modeling, is appropriate. In the general case, there is no reason to expect arbitrary input data to form a smooth manifold. However, in the underwater inspection domain, we find this technique justified for two reasons (beyond its empirical effectiveness). First, we make the general observation that the seafloor is, although not strictly polynomial, almost by definition a watertight manifold. Second, many of our inspection targets are anthropogenic structures which do, in fact, have a good polynomial approximation.

ICP is known to be highly sensitive to initialization, due to the local minima which are often present in the cost function. If not initialized close to the correct solution, it will converge to the wrong local minimum. We address this issue in two ways. First, we initialize ICP with the most recent estimate of the relative position of the source and target poses. This is equivalent to assuming that the dead reckoning between the last correctly-aligned pose and the current pose is within the region of attraction of the correct alignment. Second, we assign a *fitness score* to each alignment that represents the normalized sum of squared distances between corresponding points. Smaller fitness scores represent better alignments. If the fitness score exceeds a threshold α, we reject the alignment and do not add a loop closure constraint to the pose graph. This threshold is dependent on the scope of the data set, point density, and noise levels; we found $\alpha = .1$ to be a reasonable value for our experiments.

For the purposes of this work, we assume data association is known; future work will address the issue of determining which pairs of submaps from the catalog to align.

4.4 Pose Graph Construction

We formulate the pose graph as a factor graph, in which nodes represent poses and factors represent constraints between poses, following the formulation in [13]. Specifically, a factor graph $G = (\mathscr{F}, \mathscr{Q}, \mathscr{E})$ is comprised of factor nodes $f_i \in \mathscr{F}$ and variable nodes $q_i \in \mathscr{Q}$. An edge $e_{ij} \in \mathscr{E}$ exists if factor node f_i depends on variable node q_j. Our goal is to find the maximizing variable assignment

$$\mathscr{Q}^* = \arg\max_{Q} \prod_i f_i(Q_i), \tag{2}$$

where Q_i is the set of variables adjacent to the factor f_i.

We construct a pose graph using constraints from dead reckoning and submap alignments. Dead reckoning constraints are periodically added, with a conservative covariance based on our sensor properties. When we get a match between two submaps, it is formulated as a relative constraint between the two corresponding anchor poses.

We estimate the covariance of a submap alignment constraint using a conservative heuristic based on the normalized covariance of the points in the source submap. While not exact, this provides an intuitively reasonable approximation. For example: alignment of the Y-Z plane would have higher certainty along the surface normal (X axis), and alignment of a uniform ball of points would be considered equally certain in all directions.

As constraints are added, they are incorporated incrementally, without costly variable re-ordering or redundant computation. The full batch optimization using Gauss-Newton, including variable reordering, is performed asynchronously in the background. This enables real-time operation, even for large pose graphs.

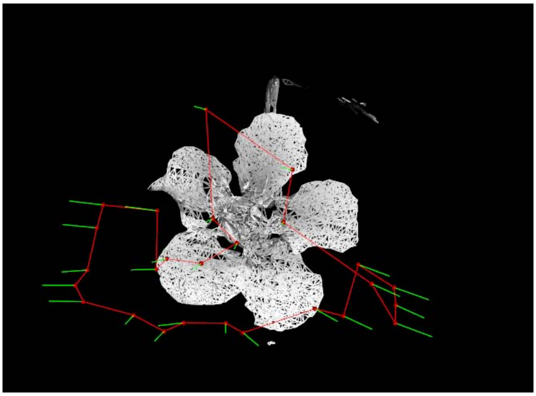


Fig. 5 Planned path for inspecting the propeller of the USS Saratoga

5 Results

We tested our system in an underwater inspection scenario on the USS Saratoga, a decommissioned aircraft carrier in Newport, RI. We performed a survey of the running gear, including the approximately 7m diameter propeller and a section of the hull above the propeller. The survey consisted of three vertical track lines from 5-6m range with the DIDSON in fixed mode, followed by 24 close-range waypoints at which the DIDSON was swept vertically. The trajectory was generated by Englot and Hover's sampling-based coverage planning [8]; the planned path is shown in Fig. 5. The entire trajectory took approximately 1200 seconds to execute. For a summary of the parameters used for our submap smoothing and alignment, see Table 1.

We do not have ground truth for this data set, but we estimate that the navigation drift was on the order of 30cm over the course of the survey. For an illustration of the navigation drift, see Fig. 2.

A mesh generated from our inspection trajectory is show in Fig. 6(b). The raw point cloud was denoised and smoothed using a variant of our submap smoothing

Table 1 Key parameters for submap formation, smoothing, and alignment

k_{sub}	500	Submap formation: min. points per submap
t_{sub}	25 sec	Submap formation: max. time per submap
v_1	2 cm	Voxel filter 1: size
k_{nn}	50	Outlier rejection: # neighbors considered
σ_{nn}	2	Outlier rejection: std. dev. limit
v_2	10 cm	Voxel filter 2: size
r	0.3 m	Polynomial surface modeling: radius
α	.1	Submap alignment: fitness threshold

algorithm described in Section 4.2. It was then meshed with a simple greedy triangulation. The raw sonar returns, reprojected from the optimized vehicle trajectory, can be seen in Fig. 6(a). Behind the propeller, the drive shaft is visible, along with a support strut. Note the high number of spurious sonar returns. Because the ship was not operational, it had a significant amount of growth on the propeller and shaft, and had become a habitat for fish, oysters, and other marine life.

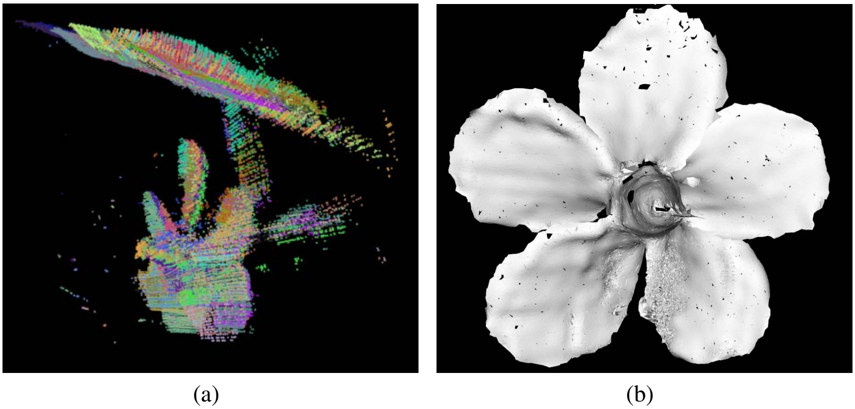

(a) (b)

Fig. 6 (a): The final point cloud from the propeller inspection trajectory. Raw submaps have been reprojected according to the SLAM-corrected trajectory, and points are colored by submap. (b): A smoothed mesh generated from the raw point cloud.

To isolate the effect of pose graph alignment, we also generated a partially synthetic data set based on the actual vehicle trajectory. We corrupted the vehicle trajectory with navigation drift, simulated by incrementally adding zero-mean Gaussian noise to dead reckoning measurements of x, y, and ψ state variables. We used a standard deviation of .01m for x and y, and .00001 radian for ψ, accumulating at 20Hz. We also added zero-mean Gaussian noise to the absolute measurements of z, θ, and ϕ to simulate increased measurement noise without drift. We also added synthetic sonar measurements created from the actual vehicle trajectory and a high-resolution sonar map of the propeller gathered in a previous experiment. To simulate sonar pings at each waypoint, we extracted points in the sonar field of view, downsampled them to the sonar resolution, and added Gaussian noise with a standard deviation of 5cm.

Fig. 7 shows the reprojected clouds from our synthetic data set, using the dead reckoning (left) and SLAM (right) trajectories. As is apparent, the dead reckoning error caused misalignment in the reprojection, which is corrected in our SLAM framework. Fig. 8 shows the average per-pose trajectory error over time. At each loop closure event, the SLAM error is reduced, while the dead reckoning error continues to accumulate. Note that, in the SLAM reprojection, the sonar noise dominates the alignment error, even though the trajectory is estimated based on the noisy sonar returns. We attribute this to our smoothing procedure detailed in Section 4.2.

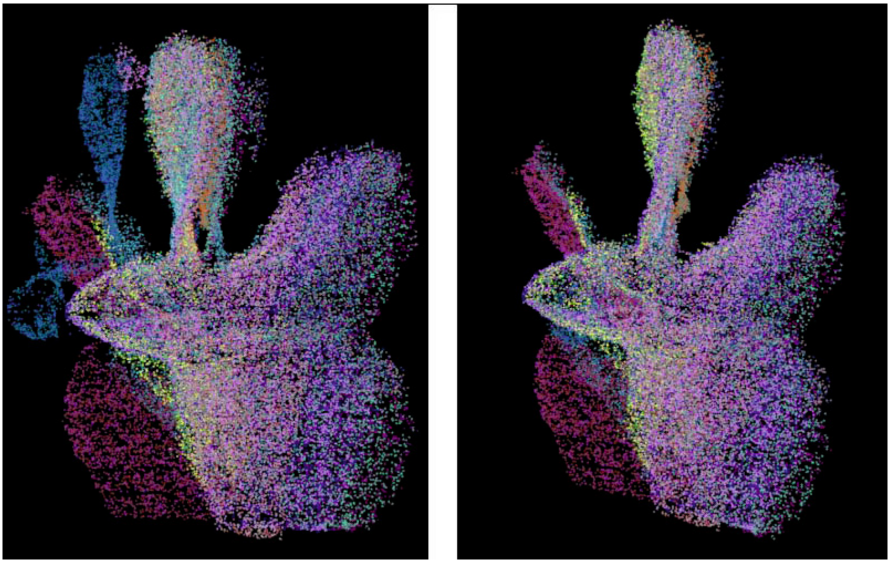


Fig. 7 Comparison of point clouds generated from dead reckoning (left) vs. SLAM (right)

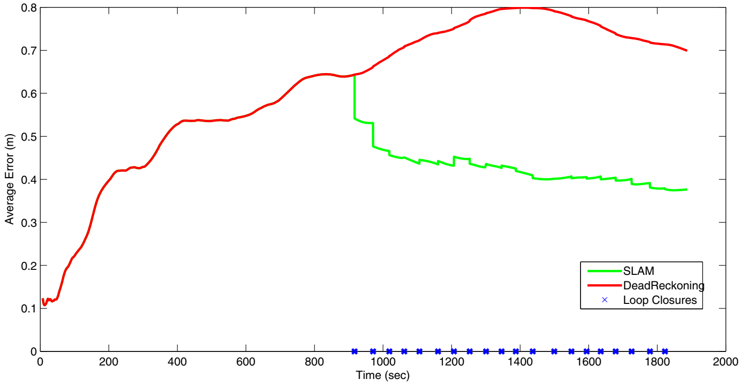


Fig. 8 Average per-pose error over time

6 Conclusion

We have presented a system for submap-based loop closure in 3D underwater mapping, and demonstrated its use on real and synthetic data sets. By smoothing the submaps before alignment, we have reduced the effect of the spurious returns frequently found in cluttered and heavily biofouled environments. By regularizing point cloud density, our system is able to combine submaps of varying resolution, while retaining the full-resolution point clouds for reprojection into the final map. Using a pose graph framework, we are able to reconstruct the entire vehicle trajectory, which is necessary to ensure full coverage of the inspection target. Our system runs in real time, and it is robust to moderate amounts of sonar noise and navigation drift.

7 Future Work

This effort suggests many areas for future research. We have a simple threshold-based system for outlier rejection, but a fully probabilistic solution could potentially improve our results. We would also like to experiment with different alignment techniques, such as multi-scale ICP or the normal distribution transform.

We could potentially further reduce error by using a volumetric, rather than point-based, reconstruction. For example, the signed distance function [6] combines multiple measurements into a single implicit surface model. This has been used to great effect with RGBD cameras, and we suspect it may help reduce the sonar noise still present in our final models. A drawback of naive volumetric techniques is that misalignments are incorporated into the final model and affect all subsequent alignments. Perhaps a hybrid system, which used our pose graph technique for initial trajectory estimates and a volumetric reconstruction for further refinement, could combine the benefits of volumetric techniques with the flexibility of pose graph SLAM.

Acknowledgements. This work was partially supported by ONR grants N00014-12-1-0093, N00014-12-10020, N00014-10-1-0936, N00014-13-1-0588, and N00014-11-1-0688; NSF Award IIS-1318392; and the NSF Graduate Research Fellowship Program. The authors would also like to thank Josh Leighton and Pedro Vaz Teixeira at MIT, Ryan Eustice, Ayoung Kim and Paul Ozog at the University of Michigan, and Jerome Vaganay at Bluefin Robotics.

References

1. Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., Silva, C.T.: Computing and rendering point set surfaces. IEEE Transactions on Visualization and Computer Graphics 9(1), 3–15 (2003)
2. Barkby, S., Williams, S., Pizarro, O., Jakuba, M.: An efficient approach to bathymetric SLAM. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS (2009)
3. Beall, C., Dellaert, F., Mahon, I., Williams, S.: Bundle adjustment in large-scale 3D reconstructions based on underwater robotic surveys. In: Proc. of the IEEE/MTS OCEANS Conf. and Exhibition (2011)
4. Besl, P.J., McKay, N.D.: A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Machine Intell. 14(2), 239–256 (1996)
5. Borrmann, D., Elseberg, J., Lingemann, K., Nuchter, A., Hertzberg, J.: Globally consistent 3D mapping with scan matching. J. of Robotics and Autonomous Systems 56(2), 130–142 (2008)

6. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: SIGGRAPH, pp. 303–312 (1996)
7. Dellaert, F., Kaess, M.: Square Root SAM: Simultaneous localization and mapping via square root information smoothing. Intl. J. of Robotics Research 25(12), 1181–1203 (2006)
8. Englot, B., Hover, F.: Sampling-based coverage path planning for inspection of complex structures. In: Proc. of Intl. Conf. on Automated Planning and Scheduling (2012)
9. Eustice, R., Singh, H., Leonard, J., Walter, M., Ballard, R.: Visually navigating the RMS Titanic with SLAM information filters. In: Robotics: Science and Systems, RSS (2005)
10. Fairfield, N., Kantor, A.G., Wettergreen, D.: Real-time SLAM with octree evidence grids for exploration in underwater tunnels. J. of Field Robotics (2007)
11. Folkesson, J., Leonard, J.: Autonomy through SLAM for an underwater robot. In: Proc. of the Intl. Symp. of Robotics Research, ISRR (2009)
12. Hover, F., Eustice, R., Kim, A., Englot, B., Johannsson, H., Kaess, M., Leonard, J.: Advanced perception, navigation and planning for autonomous in-water ship hull inspection. Intl. J. of Robotics Research 31(12), 1445–1464 (2012)
13. Kaess, M., Ranganathan, A., Dellaert, F.: iSAM: Incremental smoothing and mapping. IEEE Trans. Robotics 24(6), 1365–1378 (2008)
14. Kunz, C., Singh, H.: Map building fusing acoustic and visual information using autonomous underwater vehicles. J. of Field Robotics 30(5), 763–783 (2013)
15. Nüchter, A., Hertzberg, J.: Towards semantic maps for mobile robots. J. of Robotics and Autonomous Systems 56(11), 915–926 (2008), doi:http://dx.doi.org/10.1016/j.robot.2008.08.001
16. Ribas, D., Ridao, P., Tardós, J., Neira, J.: Underwater SLAM in man-made structured environments. Journal of Field Robotics 25(11-12), 898–921 (2008)
17. Roman, C., Singh, H.: Improved vehicle based multibeam bathymetry using sub-maps and SLAM. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), pp. 3662–3669 (2005)
18. Roman, C., Singh, H.: A self-consistent bathymetric mapping algorithm. J. of Field Robotics 24(1), 23–50 (2007)
19. Rusu, R., Cousins, S.: 3D is here: Point Cloud Library (PCL). In: IEEE Intl. Conf. on Robotics and Automation (ICRA), Shanghai, China (2011)
20. Rusu, R.B., Marton, Z.C., Blodow, N., Dolha, M., Beetz, M.: Towards 3D point cloud based object maps for household environments. J. of Robotics and Autonomous Systems 56(11), 927–941 (2008)
21. Thrun, S., Liu, Y., Koller, D., Ng, A., Ghahramani, Z., Durrant-Whyte, H.: Simultaneous localization and mapping with sparse extended information filters. Intl. J. of Robotics Research 23(7) (2004)
22. Walter, M., Hover, F., Leonard, J.: SLAM for ship hull inspection using exactly sparse extended information filters. In: IEEE Intl. Conf. on Robotics and Automation (ICRA), pp. 1463–1470 (2008)

Part II
Outdoor Driving

Towards Autonomous Mobile Robots for the Exploration of Steep Terrain

Braden Stenning, Lauren Bajin, Christine Robson, Valentin Peretroukhin,
Gordon R. Osinski, and Timothy D. Barfoot

Abstract. Steep, natural terrain offers excellent opportunities for scientific investigations into the composition and history of Mars and other planetary bodies. In this paper, we present a prototype tethered robot, *vScout* (vertical scout), capable of operating in steep, rugged terrain. The primary purpose of this vehicle is to support field geologists conducting research on cliffs, in canyons, and on crater walls. However, the long-term vision is to develop a system suitable for planetary exploration (and more diverse terrestrial applications). Unlike other systems for exploration in steep terrain, vScout has demonstrated autonomous operation on steep surfaces by making use of a network of reusable paths and visual teach & repeat. Here we describe the first vScout prototype and our experiences with it. We also outline some challenges and the directions we intend to take with this research.

1 Introduction

In this paper, we present *vScout*[1] (vertical scout), a prototype tethered mobile robot with autonomous capabilities. It can operate in terrain ranging from flat to a sheer vertical drop. Figure 1 shows a photo of the vScout prototype operating in steep, rough terrain at the Canadian Space Agency's Mars Emulation Terrain.

Long-range observations of steep, natural terrain (see Figure 2) have yielded fascinating clues to the composition, history, and the current geological processes that are active on Mars. The exposed strata are a glimpse at the subsurface without the

Braden Stenning · Lauren Bajin · Christine Robson · Valentin Peretroukhin ·
Gordon R. Osinski · Timothy D. Barfoot
University of Toronto Institute for Aerospace Studies, Toronto, Ontario, Canada
e-mail: {braden.stenning,tim.barfoot}@utoronto.ca

Gordon R. Osinski
University of Western Ontario, Depts. of Earth Science, Physics and Astronomy, Canada
e-mail: gosinski@uwo.ca

[1] For videos of vScout visit: http://youtu.be/fAQiyHssJYM

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_3, © Springer International Publishing Switzerland 2015

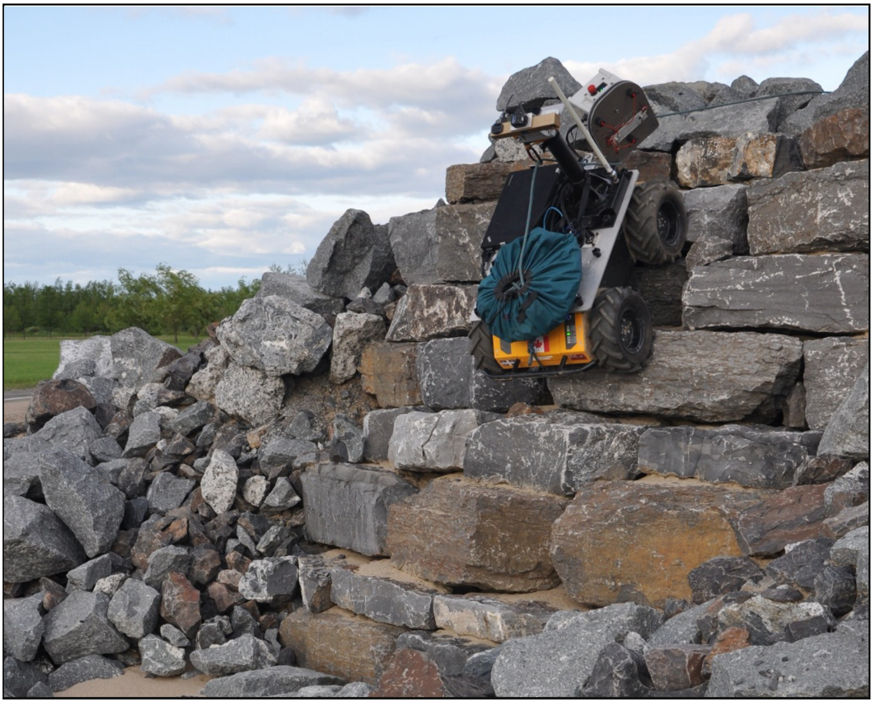


Fig. 1 The vScout prototype operating in steep, rough terrain in the Mars Emulation Terrain at the Canadian Space Agency in St. Hubert, Quebec, Canada, June 2013. Video from this test is available at: `http://youtu.be/z5ud7k9ozvQ`

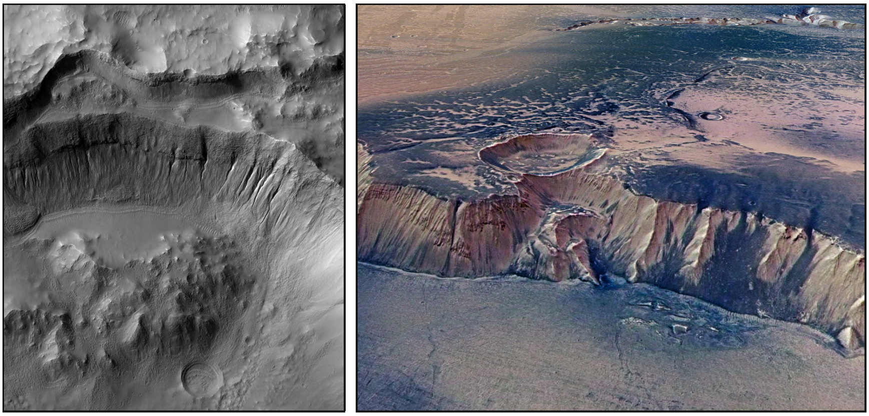


Fig. 2 Steep terrain on Mars. Left: gullies on the wall of Mariner Crater, image: NASA/JPL/University of Arizona. Right: perspective view of Echus Chasma, credits: ESA/DLR/ FU Berlin (G. Neukum)

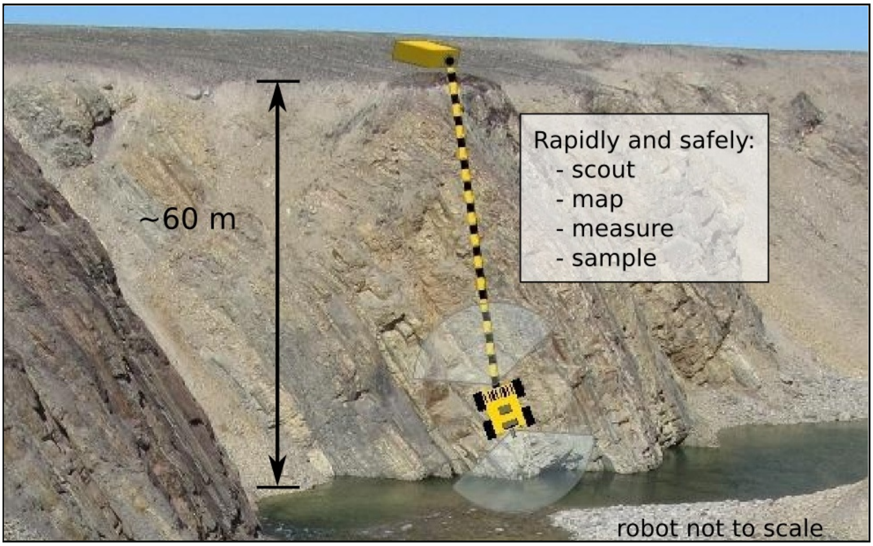


Fig. 3 Steep terrain at the newly discovered site of one of the world's largest meteorite impact craters (Victoria Island, Northwest Territories, Canada). Development of the vScout is aimed at allowing access to terrain that is scientifically important, but currently not practically accessible

need for difficult excavation or deep drilling. Yet while we have learned much from a distance, many important questions will remain unanswered while planetary scientists are without detailed local terrain models, in situ measurements, and samples returned from these steep surfaces. To obtain these we must physically access these areas. Unfortunately, steep, natural terrain is inaccessible to current planetary rovers. It is for this very reason that the Mars Science Laboratory rover Curiosity was landed *inside* Gale crater on Mars. There are only a small number of fielded research systems capable of operating in steep terrain, and fewer capable of exploring rugged, vertical cliffs [2, 6, 11]. To our knowledge, even though there has been some development of theoretical approaches to autonomy, none of these capabilities have been demonstrated in steep terrain. Autonomy is very desirable in space exploration where communications delays can make continuous, direct teleoperation far too slow or even impossible.

The Axel rover [11], in particular, is a robot for steep terrain that has been developed for space exploration. With vScout we are taking a different approach. Instead of directly targeting space exploration, we are looking to support field geologists doing work in remote areas on Earth. For an example site, see Figure 3. Even on Earth, access to steep terrain is at best difficult and time-consuming to do safely, and often it is simply not feasible because of safety, logistics, or the available time. We believe that by using existing guidance, navigation, and control (GN&C) techniques that leverage the expertise of a human operator, we can create a tool useful to scientific investigations on Earth, and that the terrestrial benefits of such a system will make it natural and reasonable to demand and expect similar capabilities in planetary exploration. This is certainly an ambitious proposal. In this paper we present the first steps toward this goal and we show preliminary results. We also outline some challenges we are working on and the directions of future research.

The remainder of this paper is as follows: Section 2 provides background information, Section 3 describes the vScout, Section 4 details the testing, the paper finishes with challenges and future work (Section 5), and conclusions (Section 6).

2 Background

The background is divided into two parts: i) an overview of robots for use in steep terrain, and ii) an overview of visual teach & repeat and a network of reusable paths, a technology that is used for the autonomous operation of vScout.

2.1 Robots for Steep, Natural Terrain

Robots have excelled at operating in some environments too dangerous for humans. This includes some types of steep areas. Although the discipline is small, the research into vertical robots has developed several promising approaches to access steep, natural terrain. Currently, none of these systems are available for regular use either because their capabilities are too limited, their cost is too high, or the deployment and operations logistics are too burdensome.

Some research has been done on developing robots that climb like humans or other more capable primates (e.g., Capuchin [16]). Similarly, other biologically inspired designs (e.g., RiSE [14]) can climb up from below with no rope (except perhaps as a safety line); however, the prototypes are still quite slow, require specific types of surfaces, or significant advances are necessary before they can reliably operate in natural environments. There are also systems that are designed for use in specialized (usually human-made) environments. For example, there are systems that have made use of magnets [5], suction [3], microspines [7], or adhesives [10]. However, the assumptions made about the surfaces and structures make general as-is use of these systems unlikely in natural terrain.

One of the most studied techniques is to have the robot descend from above using a tether. Notable examples are Dante II [2], TRESSA [6], and Axel [11]. These systems have been field tested and they can all operate on vertical slopes.

The Dante II rover [2] was an 8-legged frame walking robot used to explore Mt. Spurr, a remote Alaskan volcano, in 1994. This large robot (nearly 800 kg) was teleoperated from a remote location using onboard television cameras and laser rangefinders. The power and communications links were through the tether. Over the course of more than five days, Dante II was used to successfully complete the primary objectives of the mission. However, while it was returning to the lip of the crater, it ended up tipping over. It was eventually recovered using a helicopter and two people who hiked down to attached a sling. This experience highlights two of the many significant challenges to operating in steep terrain: i) the need for situational awareness (Dante II's laser rangefinder was inoperable at the time of tip-over), and ii) the fact that recovery in the event of mishap may be dangerous, expensive, or even impossible. If these robotic tools are to be used regularly they should, on average, be much safer and less expensive that the alternatives.

As demonstrated in Dante II and the other systems, the tether can allow operation on steep terrain, but it also makes significant lateral movement challenging. The TRESSA [6] system attempts to improve the lateral mobility by having two mobile robots at the top that are tethered to a single four-wheel Cliffbot on the steep surface. This will work well in fairly smooth steep areas but in rougher terrain it may even further complicate and limit the mobility.

The TRESSA system stores the tether at the top of the steep terrain. While this approach does reduce the weight of the vehicle on the cliff, it will also increase the wear on the tether and eventually limit the range of the tethered vehicle (once the tether drag along the ground reaches the limit of what the vehicle can pull). An alternative is the have the tether stowed on and deployed from the descent vehicle (as done by Dante II [2] and Axel [11]).

The Axel rover [11] is a two-wheeled minimalist chassis that has undergone several iterations. Unlike the other systems that have been mentioned, Axel can operate even if it is flipped over. Its two wheels are on either end of the cylindrical body and an arm extends from mid-width keeping the tether away from the main body. The tether arm can be actively controlled for improved rover mobility and to keep the tether off the ground. There has been some theoretical work done on path planning for Axel [1], but our understanding is that it has yet to move beyond simulation.

Like Dante II, Axel's power and communication links are through the tether. There are strong reasons for using the tether for more than simply mechanical support. Onboard power is a significant challenge in mobile robotics. It can be heavy and/or expensive, and the inclusion of heavy systems (such as onboard batteries or gas generators) can lead to increased structure and actuator costs. Additionally, these self-contained power systems have limited capacity and this limits the maximum mission duration (thus the vehicle cannot loiter for extended periods of time).

Similarly, a communications link integrated into the tether avoids the challenge of reliable, high-speed wireless communications. Wireless communications become difficult or impossible when there is no line-of-sight (over-the-edge). Long range antennas can be bulky and will have pointing issues. Finally, wireless communications can use a lot of power, thus exacerbating the challenges of onboard power.

2.2 Visual Teach & Repeat and a Network of Reusable Paths

We have some background experience with visual navigation and we believe it will be appropriate for use on vScout. *Visual teach & repeat* (VT&R) [4, 9] allows a robot to drive arbitrarily long distances, without the use of GPS, along previously established routes. In these systems, a chain of small maps is attached along the robot's path (estimated using visual odometry [8]) during a teaching phase; to repeat the path, the robot localizes against each small map in sequence as it drives. This

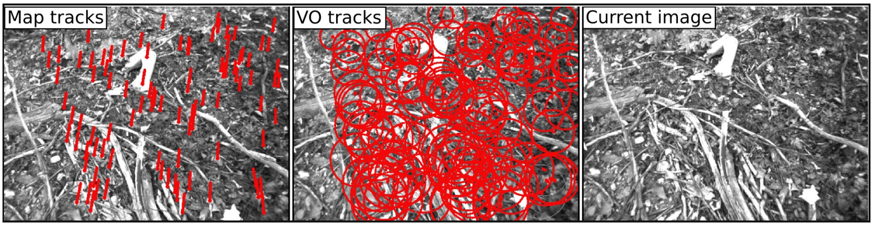


Fig. 4 Stereo-camera-based visual teach & repeat onboard the vScout. When repeating a taught route, localization against the map is interleaved with visual odometry (VO). The map tracks show the keypoints that have been matched between the taught route and the current stereo image pair. The VO tracks show the keypoint matches between the current stereo pair and the previous pair.

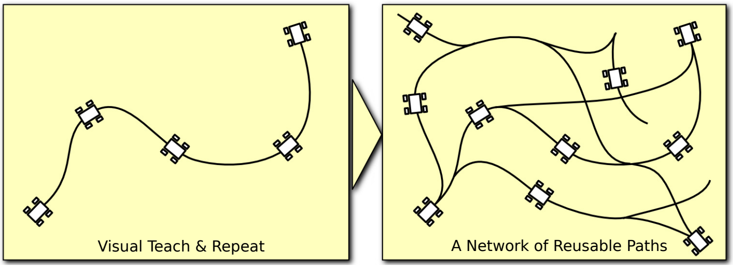


Fig. 5 A network of reusable paths (NRP) makes use of visual teach & repeat (VT&R). VT&R is a single chain of poses that the robot can repeat in either direction. NRP uses VT&R and extends it to an arbitrary network, allowing the robot to return to any previously visited pose. We use both VT&R and NRP on vScout.

local map approach works well regardless of the nature of the surface on which the robot is driving. Figure 4 shows images from stereo-camera-based VT&R onboard the vScout while repeating a path. The keypoints from the current image (right) are matched against the keypoints from the taught image (left). Visual odometry (center) is used to estimate the motion before attempting to match against the map.

The *network of reusable paths* (NRP) concept extends VT&R systems from using a simple chain of local maps, to an arbitrary network of local maps [15] (see Figure 5). The robot can return to any point on the network at any time, and by driving into new areas, the network can be extended.

On vScout, we currently use both VT&R and NRP to leverage human expertise in terrain assessment and path planning. Once a path has been taught, vScout can be entrusted to repeat it, in either direction, and the operator no longer needs to directly control the vehicle, a tedious and difficult task. Eventually we would like to have vScout autonomously teach new paths as well as repeat them. This means incorporating suitable terrain assessment and path planning capabilities. We have done this for other rovers [15] but not yet for systems in steep terrain.

3 The vScout

This section describes the current vScout proof-of-concept prototype. Our philosophy with this research has been that field experience will guide the design. Therefore, our priority has been to develop an end-to-end system rapidly, even though it is not always consistent with the long-term vision at a component level. At some points in this text we note where the long-term vision differs from the prototype and a more complete discussion of the differences is in Section 5. This section is divided into four parts: i) a description of the use scenario (Section 3.1), ii) a description of the hardware configuration (Section 3.2), iii) detail on the powered ascender mechanism (Section 3.3), and iv) an overview of the onboard GN&C (Section 3.4).

3.1 High-level Scenario

The vScout was developed with the goal of aiding field geologists. The ultimate goal is simply to have a user press a button and have the robot autonomously descend, map and document everything it can, and come back up, while carrying standard geological instruments. This preliminary system is working toward that goal. The following high-level scenario outlines how this first prototype is intended to be used.

1. Outfit the vScout with the desired measurement package.
2. Position the vScout near the top of the steep terrain and create a secure anchor.
3. As necessary:
   - Move to get a better view to remote control the vScout.
     This may be above, below, to the side, or even across from the cliff. Direct observation is used to augment data from the onboard cameras and sensors.
   - Manually drive the robot into new areas and add to the NRP.
   - Automatically generate/update a 3D model of the terrain.
   - Autonomously return to any previously visited point on the network.
4. When done at a site, command the vehicle to autonomously return to the start.

3.2 Hardware Overview

The vScout is a tethered robot designed to descend from an anchor near the top of steep terrain. For this first version we retrofitted a Husky A200 from Clearpath Robotics (see a schematic of the prototype system at the top of Figure 6). On the back of the Husky is an ascender (described in detail in Section 3.3). The ascender allows the robot to hang suspended by a 8 mm climbing rope (see Figure 7).

A Point Grey Bumblebee XB3 stereo camera is mounted on a mast facing the front of the vehicle (away from the tether). We also tried mounting the camera near the front and rear of the vehicle, as seen in photos of the vScout. There is an onboard laptop computer used for navigation and data logging. There is also a wireless connection used to connect the operator station to the vehicle. The system has a mass of approximately 70 kg, and dimensions of approximately 130 cm long and 70 cm wide. The highest point is approximately 95 cm off the ground.

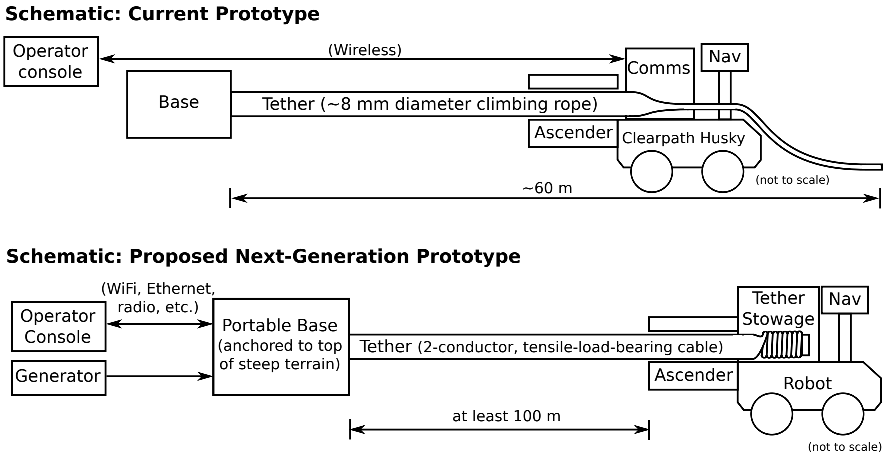

Schematic: Current Prototype
Operator console
(Wireless)
Comms
Nav
Base
Tether (~8 mm diameter climbing rope)
Ascender
Clearpath Husky
(not to scale)
~60 m
Schematic: Proposed Next-Generation Prototype
(WiFi, Ethernet, radio, etc.)
Operator Console
Generator
Portable Base (anchored to top of steep terrain)
Tether Stowage
Nav
Tether (2-conductor, tensile-load-bearing cable)
Ascender
Robot
at least 100 m
(not to scale)

Fig. 6 Schematics of the current prototype (top) and the proposed next-generation vScout (bottom). The current prototype uses VT&R/NRP with manually taught paths. The goal eventual system is to have the vScout, with the push of a button, autonomously descend, map, and return to the top of the cliff. More details on the proposed system are available in Section 5.

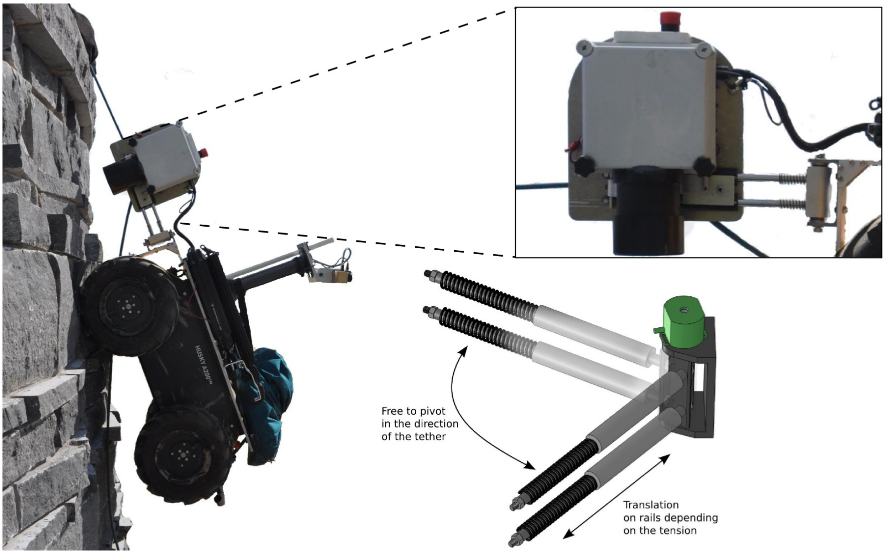

Free to pivot in the direction of the tether
Translation on rails depending on the tension

Fig. 7 The vScout on vertical terrain (left). The ascender, on spring-loaded rails that pivot, was attached to the back of a Clearpath Robotics Husky A200. The pivot angle could be measured and the tension in the tether could be estimated based on the displacement of the spring-loaded rails.

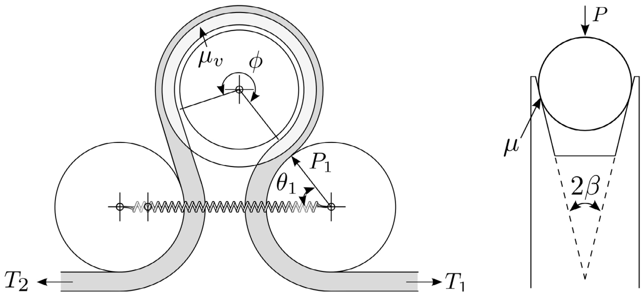


Fig. 8 A model of the ascender mechanism (left) and a profile of the grooved top pulley (right)

3.3 The Ascender Mechanism

The ascender assembly was mounted on the back of the Husky, as seen in Figure 7. It was free to passively pivot from side to side, and it could also slide on spring-loaded rails. The rails had the dual use of providing suspension to the system and, by measuring the displacement, we could estimate the tension in the tether.

The main part of the ascender consists of three wheels (see Figure 8). The middle wheel is grooved and attached to an electric motor. The other two wheels are idlers that are connected to each other through a spring-loaded mechanical linkage. The idlers slide in the *x*-direction and pinch the rope against the grooved pulley. The following analysis shows that the tether will not slip if the coefficient of friction between the aluminum and the tether is sufficiently high.

Assume that the tether cross-section is not deformable. Then the equivalent coefficient of friction for the groove, μ_v, as a function of the coefficient of friction between the materials of the tether and the pulley, μ, is

$$\mu_v = \mu / \sin\beta, \tag{1}$$

where β is the groove angle as shown in the right of Figure 8.

The *capstan equation* is a way to model the load that a tether wrapped around a smooth cylinder can hold without slipping. The high-tension end of the tether has a load of T_{high} and the low-tension end of the tether has a load of T_{low}. The angle swept by the tether is ϕ. The capstan equation is

$$T_{\text{high}} \leq T_{\text{low}} e^{\mu\phi}. \tag{2}$$

Considering the model of the ascender shown in the left of Figure 8, and using the relations from (1) and (2), we can write the no-slip holding load of the ascender as,

$$T_2 \leq (\mu_v P_1 + T_1)\, e^{\mu_v \phi}, \tag{3}$$

where T_2 and T_1 are the high-tension and low-tension loads on the tether, and P_1 is the pinching force between the grooved pulley and the right idler wheel. In this analysis we conservatively assume $T_1 = 0$. Due to the geometry of the pulleys and nature of the spring-loaded bar connecting the idlers, when T_2 is high, $P_1 = T_2/\cos\theta_1$, where θ_1 is the angle of the pinching force, P_1, as shown in Figure 8. This means the equation (3), which governs the holding force of the ascender, simplifies to

$$\mu \frac{1}{\sin\beta\cos\theta_1} e^{\mu \frac{1}{\sin\beta}\phi} \geq 1. \tag{4}$$

If the inequality in (4) is satisfied, the tether will not slip. □

The threshold coefficient of friction, μ_{thresh}, beyond which slip will occur, can be found by solving

$$\mu_{\text{thresh}} \frac{1}{\sin\beta\cos\theta_1} e^{\mu_{\text{thresh}} \frac{1}{\sin\beta}\phi} = 1. \tag{5}$$

The geometry of the mechanism is such that: $\beta = 21°$, $\theta_1 = 55°$, and $\phi = 245°$, and therefore, $\mu_{\text{thresh}} = 0.08$. We have measured μ to be between 0.3 and 0.35 using an inclined plane test. Since $\mu \geq \mu_{\text{thresh}}$, the ascender on the vScout should not allow the tether to slip. In practice, we have not had the rope slip while it is under tension.

3.4 The Onboard Guidance, Navigation & Control System

VT&R and NRP use the stereo camera in order to allow vScout to return to any previous point. The paths are taught by an operator remote controlling the robot. The path tracker is that same as in previous VT&R/NRP systems [9, 15]. It does not consider the slope of the terrain. The tether controller was designed to keep the tether tight while keeping the tether speed near the speed of the vehicle. The control law governing the speed of the tether, v_t, is

$$v_t = \begin{cases} k_1 v_r & T_t \geq T_{thresh} \text{ and } v_v \leq 0 \\ k_2 v_r & T_t \geq T_{thresh} \text{ and } v_v > 0 \\ k_1 v_r + k_3 & T_t < T_{thresh} \text{ and } v_v \leq 0 \\ k_4 & T_t < T_{thresh} \text{ and } v_v > 0 \end{cases}, \tag{6}$$

where, v_v is the speed of the vehicle in the x-direction (i.e., $v_v > 0$ is moving forward, and $v_v < 0$ is moving in reverse). The measured tether tension is T_t, and the threshold tension defines the point below which the tether is considered slack. The gains, k_1 and k_2, are set close to 1, with $k_1 \geq 1$ and $k_2 \leq 1$. The gains k_3 and k_4 are used when the tether is slack, in order to take up slack, they are set so that $k_3 \leq 0$ and $k_4 \leq 0$. This approach worked well enough in these tests, but there is much room for improvement. For instance, this approach leads to higher-than-necessary tether loads, and therefore power usage. We anticipate that the path-tracking and tether controllers (especially in the context of repeating a previous path) will be active areas of future research.

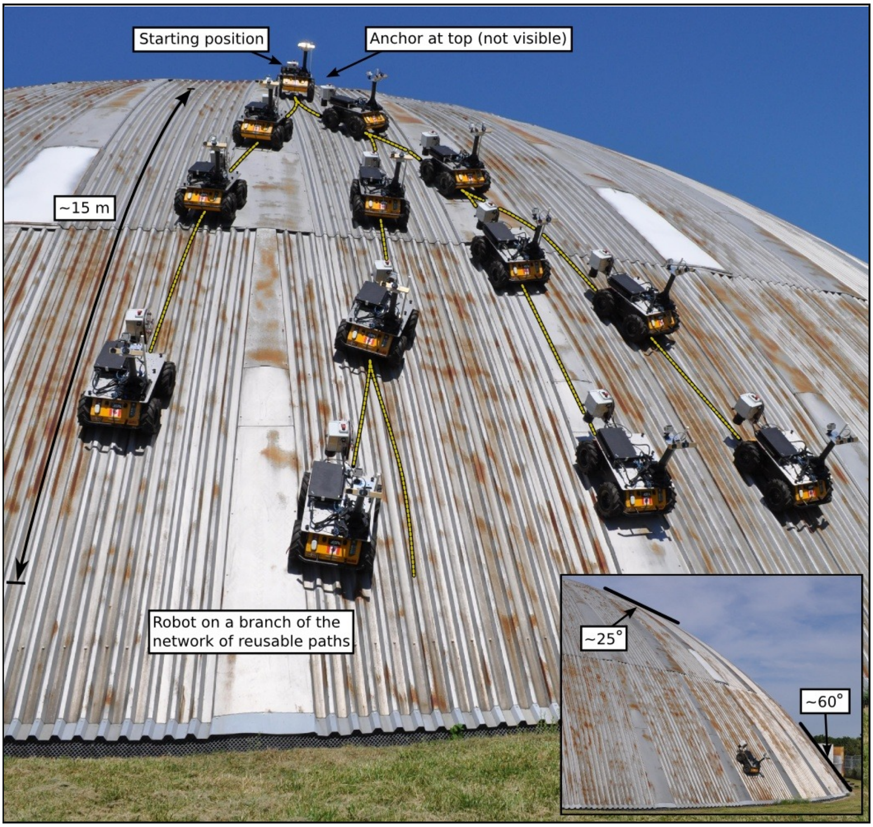

Starting position
Anchor at top (not visible)
~15 m
Robot on a branch of the network of reusable paths
~25°
~60°

Fig. 9 A time-lapse view of the vScout operating on the Dome at UTIAS. The vScout used a network of reusable paths. It was manually taught a path and then it was able to autonomously repeat that path to return to any previously visited point.

4 Testing

The first two sets of testing we did were on a steep building (the Dome) near our lab[2], and at the Mars Emulation Terrain at the Canadian Space Agency[3] (see Figure 1). Those first two sets of testing were entirely manually controlled. In later tests on the Dome and the steep walls of a ravine, vScout demonstrated autonomous behavior that made use of visual teach & repeat and a network of reusable paths. The operator manually taught the paths, using their skill and judgment to establish safe routes, and the robot would autonomously repeat the paths.

In total, we conducted five tests that made use of VT&R on the ravine wall, four tests of VT&R on the Dome, and five tests of NRP on the Dome. Figure 9 shows

[2] A video of the first ascent of the Dome at UTIAS: `http://youtu.be/o2hrGYYP9b8`

[3] A video of vScout at the CSA's Mars Emulation Terrain: `http://youtu.be/z5ud7k9ozvQ`

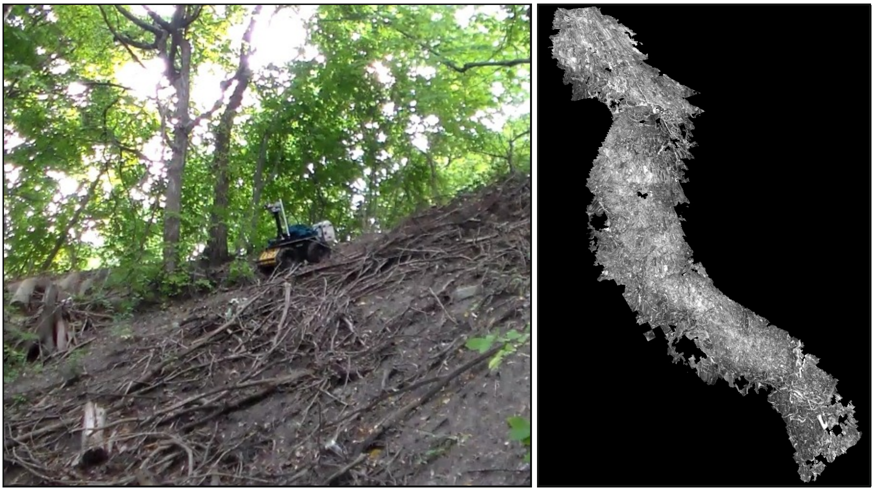


Fig. 10 The vScout in the ravine (left). A 3D model of the terrain from the first test using VT&R on the ravine wall (right). The model was made using Mobile Scene Modeler (mSM) from MDA.

the fourth test of NRP on the Dome[4]. In that test the vScout was first taught the path on the far right of Figure 9. It then autonomously reversed the path until the human operator took control and taught another branch in the network. This same process was repeated for all five branches in this network.

The vScout repeated all paths and networks on the Dome, and it was able to localize relative to the taught path. The repetitive texture of the corrugated steel was challenging for the localization system, but the discolorations created a unique, if relatively sparse, constellation of distinctive features. Additionally, the localization framerate ($\tilde{1}0\,\text{Hz}$) was fast relative to the speed of the vehicle ($\leq 0.25\,\text{m/s}$). However, the loose terrain of the ravine, combined with the orientation of the stereo camera (pointing down in front of the vScout in order to collect imagery for map building), meant that the appearance of the scene would change dramatically between teach and repeat phases (because the vScout would disturb the soil as it drove over). This made it difficult for VT&R to localize against the map, and instead the system had to use solely visual odometry to estimate its pose. A possible solution is to use a stereo camera that is pointed to the side in order to minimize the chance of driving over the visual features used for localization. However, this would mean that another camera would be needed in order to see in front of the vScout.

After each test, we used the stereo camera imagery to construct a 3D model of the terrain. This was done by MDA's Mobile Scene Modeler (mSM) [13]. Figure 10 shows the model generated during the first test of VT&R on the ravine wall. Only the outbound images (i.e., the images from teaching the path) were used to build the

[4] A video of the fourth NRP test on the Dome is at: `http://youtu.be/fAQiyHssJYM`

models in order to avoid the previously noted problem where the vehicle changes the visual appearance of the scene.

5 Challenges and Future Works

Using this first prototype has led to many valuable lessons, these are listed below.

1. The configuration of the terrain directly under the vehicle was critical knowledge for the operator when the vScout was in rough terrain.
   - *Later vScout designs will use more sensors and algorithms to give the remote operator better awareness of the terrain under the vehicle. These will also likely act as the foundation for terrain assessment and path planning capabilities necessary for one-button autonomous descent, mapping, and return.*
2. The ability to attach to or detach from the tether at mid length was particularly useful (as opposed to having to thread the tether through the ascender). This gave us the opportunity to quickly start the vScout from the bottom rather than the top.
3. The vehicle was more maneuverable in steep terrain with only two wheels on the ground (like Axel). However, the four wheels helped when overcoming obstacles.
4. The vehicle spent a great deal of the time with its underside in contact with the terrain. This reinforces the need for a good skid plate and adequate protection for any cameras underneath the vehicle.
   - *We embraced the fact that this prototype would often scrape, bang, or high-center during operation. In later prototypes we will modify the design to reduce the chance of the vehicle getting caught (e.g., streamline the ascender).*
5. The tether was not only useful in steep areas, but also quite beneficial on flat, but rough, terrain. The tether could be used to simply pull the vehicle off if the robot got stuck, and it provided a great deal of stability that made a tip-over seem much less likely when traversing hazards.

We have also experienced other challenges which we expect to address is the next vScout. For instance, path tracking will become difficult as we begin to have the vehicle move more laterally on steep terrain. However, our lab is currently developing a learning path tracking controller [12], that, when combined with improved tether control, may offer a solution.

Additionally, we are already testing the limits of our current version of VT&R and NRP. Prior to vScout, we had only used VT&R in relatively flat terrain. This has meant that the path was able to be reversed with the vehicle in the same orientation, and consequently the rigidly mounted camera was also in the same orientation and it saw the same scene regardless of whether the vehicle is traveling forward or reverse. However, when a skid-steer vehicle is operating on steep slopes, it must turn into the slope in order to track cross-slope paths. This means the vehicle orientation depends on the direction of travel, and a rigidly mounted camera is no longer feasible. Some possible solutions are to use cameras with a 360° field of view or a camera mounted on a pan-tilt unit.

Finally, in the next vScout, as in the bottom of Figure 6, there will be a tether stowage system. The tether will also be used to transmit power and communications to the vehicle. This should help avoid the risk of fouling the tether in the wheels, improve communications, and reduce battery weight.

6 Conclusions

This paper presented a prototype tethered robot, called vScout, intended to help field geologists in the exploration of steep terrain. The long-term objective is to develop technologies to enable the exploration of vertical surfaces on other planets. The system has been used at several sites where it operated under manual control and at times autonomously, and built a three-dimensional model of the terrain. The autonomous capabilities make use of VT&R and NRP to repeat previously traveled routes. Later versions of the hardware, software, and operations scenarios will build on our experiences with this first system, leading toward a vScout that will autonomously descend, map, and return, all with the push of a button.

References

1. Abad-Manterola, P., Nesnas, I., Burdick, J.: Motion Planning on Steep Terrain for the Tethered Axel Rover. In: IEEE ICRA 2011, pp. 4188–4195 (2011)
2. Bares, J., Wettergreen, D.: Dante II: Technical Description, Results and Lessons Learned. Int. J. Robot. Res. 18(7), 621–649 (1999)
3. Briones, L., Bustamante, P., Serna, M.: Wall-climbing robot for inspection in nuclear power plants. In: ICRA 1994, vol. 2, pp. 1409–1414 (1994), doi:10.1109/ROBOT.1994.351292
4. Furgale, P., Barfoot, T.: Visual Teach and Repeat for Long-Range Rover Autonomy. J. Field Robot. 27(5), 534–560 (2010)
5. Hirose, S., Tsutsumitake, H.: Disk Rover: A Wall-Climbing Robot using Permanent Magnets. In: IROS 1992, vol. 3, pp. 2074–2079 (1992), doi:10.1109/IROS.1992.601942
6. Huntsberger, T., Stroupe, A., Aghazarian, H., Garrett, M., Younse, P., Powell, M.: TRESSA: Teamed Robots for Exploration and Science on Steep Areas. J. Field Robot. 24(11-12), 1015–1031 (2007), doi:10.1002/rob.20219
7. Kim, S., Asbeck, A., Cutkosky, M., Provancher, W.: Spinybotii: climbing hard walls with compliant microspines. In: ICAR 2005, pp. 601–606 (2005), doi:10.1109/ICAR.2005.1507470
8. Matthies, L., Maimone, M.W., Johnson, A.E., Cheng, Y., Willson, R.G., Villalpando, C., Goldberg, S.B., Huertas, A., Stein, A.N., Angelova, A.: Computer Vision on Mars. Int. J. Comput. Vision 75(1), 67–92 (2007)
9. McManus, C., Furgale, P., Stenning, B., Barfoot, T.D.: Lighting-invariant visual teach and repeat using appearance-based lidar. J. Field Robot. 30(2), 254–287 (2013)
10. Murphy, M.P., Kute, C., Mengüç, Y., Sitti, M.: Waalbot II: Adhesion Recovery and Improved Performance of a Climbing Robot using Fibrillar Adhesives. Int. J. Robot. Res. 30(1), 118–133 (2011), doi:10.1177/0278364910382862

11. Nesnas, I.A., Matthews, J.B., Abad-Manterola, P., Burdick, J.W., Edlund, J.A., Morrison, J.C., Peters, R.D., Tanner, M.M., Miyake, R.N., Solish, B.S., Anderson, R.C.: Axel and DuAxel rovers for the sustainable exploration of extreme terrains. J. Field Robot. 29(4), 663–685 (2012), doi:10.1002/rob.21407
12. Ostafew, C.J., Schoellig, A.P., Barfoot, T.D.: Iterative learning control to reduce path-tracking error for a mobile robot. In: IROS 2013, Tokyo, Japan (2013)
13. Se, S., Jasiobedzki, P.: Photo-realistic 3D model reconstruction. In: ICRA 2006 (2006)
14. Spenko, M.J., Haynes, G.C., Saunders, J.A., Cutkosky, M.R., Rizzi, A.A., Full, R.J., Koditschek, D.E.: Biologically inspired climbing with a hexapedal robot. J. Field Robot. 25(4-5), 223–242 (2008), doi:10.1002/rob.20238
15. Stenning, B.E., McManus, C., Barfoot, T.: Planning using a network of reusable paths: A physical embodiment of a rapidly exploring random tree. J. Field Robot. (2013)
16. Zhang, R., Latombe, J.C.: Capuchin: A free-climbing robot. Int. J. Adv. Robot. Syst. 10 (2013)

Drivable Road Detection with 3D Point Clouds Based on the MRF for Intelligent Vehicle

Jaemin Byun, Ki-in Na, Beom-su Seo, and Myungchan Roh

Abstract. In this paper, a reliable road/obstacle detection with 3D point cloud for intelligent vehicle on a variety of challenging environments (undulated road and/or uphill/ downhill) is handled. For robust detection of road we propose the followings: 1) correction of 3D point cloud distorted by the motion of vehicle (high speed and heading up and down) incorporating vehicle posture information; 2) guideline for the best selection of the proper features such as gradient value, height average of neighboring node; 3) transformation of the road detection problem into a classification problem of different features; and 4) inference algorithm based on MRF with the loopy belief propagation for the area that the LIDAR does not cover. In experiments, we use a publicly available dataset as well as numerous scans acquired by the HDL-64E sensor mounted on experimental vehicle in inner city traffic scenes. The results show that the proposed method is more robust and reliable than the conventional approach based on the height value on the variety of challenging environment.

1 Introduction

The accurate perception of the environment is a very important step to drive autonomously for an intelligent vehicle, namely the detection of road area and obstacle. The road and obstacle detection are being nearly performed by using a various kind of sensors. Many teams participating in the DARPA Urban Challenge have nearly performed based on the data acquired by 2D range sensors for road region and obstacle detection [17, 10, 1, 2] . However, these sensors scan the environment along a plane within a limited viewing angle, thus the objects above or below this plane cannot be detected. A number of approaches focused on the use of vision exclusively have been studied for decades [18],[12]. The fusion of the range and

Jaemin Byun · Ki-in Na · Beom-su Seo · MyungChan Roh
Robot and Cognitive System Research Department(RCSRD) in Electronics
and Telecommunications Research Institute (ETRI), Daejeon, South Korea
e-mail: {jaemin.byu,kina,bsseo,mcroh}@etri.re.kr

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_4, © Springer International Publishing Switzerland 2015

vision data that allows a richer description of the world have been developed [5],[14]. Recently, a three-dimensional range-scanner which provides 3D point cloud instead of 2D slice of the environment have been commercially introduced. Although there has been an overwhelming amount of work on perception in 2D and 2.5D, but the problem of perception in 3D has been addressed by comparatively fewer researchers yet. One of its main reasons is the enormous amount of data provided by 3D sensors. The amount of data in a single scan of 3D sensor is usually several times larger than that of a 2D scan. Therefore, how to build consistent and efficient 2D representations out of 3D range data is important for the sensor data processing as well as road/obstacle detection. Improving on earlier these work, the main contribution of this paper is a method for efficient road detection based on MRF with LBP(Loopy Belief Propagation). We also employ a cylindrical 2D grid map with the different size of cell corresponding to the distance from vehicle. Besides, our objective is to present a method that can detect accurately drivable road and obstacle regions in a variety of challenging environment such as undulated road, uphill/downhill, rolling /pitching of the host vehicle as shown in Fig. 1.

In our work, the 3D range data is acquired by a Velodyne HDL-64E sensor as shown in the Fig.1-(b), which is mounted on the top of the vehicle, it is covering a total vertical range of approximately 25 degrees. To obtain data from the whole environment, the laser scanner rotates at a speed of 10 Hz. A data packet from the LIDAR consists of the rotational angle of the scanner itself, the range and the intensity measurement of each laser. From this data, a complete scan of the environment can be computed. By the way, the motion of vehicle itself would affect these 3D data information as the sensor is mounted moving on the vehicle. To remove the distortion that is caused by the movement of the vehicle during one revolution, the paper presents the strategy that involve in correction process through estimating the posture of vehicle. The paper is organized as follows. In the next section we give an outline of relevant works, followed by the detailed description of our approach. Experimental results are given in section 5. Section 6 concludes this paper and provides a perspective for future research in this area.

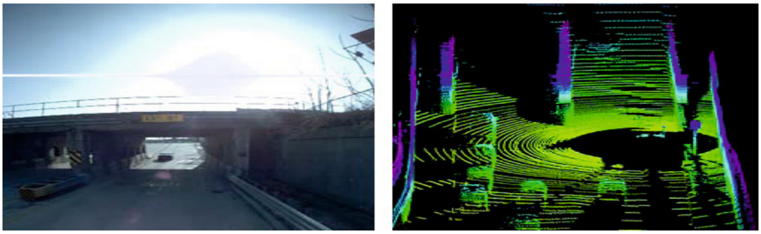


Fig. 1 (a)Uphill road.(b) 3D point clouds acquired by Velodyne LIDAR.

2 Previous Works

With range scanning devices becoming standard equipment in mobile robotics, the task of 3D scan segmentation and classification is one of increasing practical relevance. Typical algorithms for road and obstacle detection with 3D LIDAR are as follows: One of the most widely used method is the projection of 3D point clouds on the assumed or estimated ground plane and finds similar x-y coordinates whose height exceeds a given threshold value. This is represented by a grid in which each cell contains only one height by selection of the average, max, min height of the sensor data located in each grid cell [16],[15]. One of the advantages is that the several sensors can be fused easily and that mapping is straightforward. Many teams participating in the DARPA Urban Challenge successfully applied this method. However, the difficulty of road detection has still in sloped terrains or the situations with big rolling/pitch angle of the host vehicle. Both Leonard et al.[8] and Himmelsbach et al.[7] describe a method that identifies points in the point clouds that are likely to be on the ground, and then fit a ground model through those ground points. And other points above the ground model are deal with as obstacle point. Douillard et al.[3] proposes a strategy that utilizes ground models of non-constant resolution either providing a continuous probabilistic surface or a terrain mesh built from the structure of a range image. Moosmann et al.[11] proposes graph-based approach to segment ground and objects from 3D LIDAR scans using a novel unified, generic criterion based on local convexity measures. Guo et al.[6] use a graph-based approach for 2D road representation of 3D point clouds with respect to the road topography. The method describes also the gradient cues of the road geometry to construct a MRF and implements a belief propagation (BP) algorithm to classify the road environment into four categories, i.e. the reachable region, the drivable region, the obstacle region and the unknown region. However their method uses only gradient value for labeling so that it cant sometimes be distinguished the ground and the roof of vehicle. Li and Li[9] proposes a method of Four Directions Scan Line Gradient Criterion (4DSG) that is calculated the gradient with neighboring points. These features can not only reflect the flatness of pavement, but also reflect the distinguishing feature of point cloud on curbs in four directions. Bohren et al.[1] addresses a method that road points can also be detected based on the reflectivity of the ground in the Velodyne scans. However, such approach can only work well under good conditions so that their road/obstacle detection has to be supplemented by other sensor. The outline of our work can be show in the Fig.2. The proposed approach differs from previous related work. Main contributions that we propose are as follow

- Unlike most of the previous works, we focus on the correction of distorted 3D point cloud occurred by motion of vehicle (high speed/ move up and down) in practical road situation.
- We employ the approach that 3D point clouds are projected on the grid map in the cylindrical coordination. We have considered the fact that the point cloud by Velodyne sensor will be gradually sparse from near to far and the dramatic change happens between adjacent beams can reveal the vertical change of the environment along the circular direction as shown in Fig.2-(b).

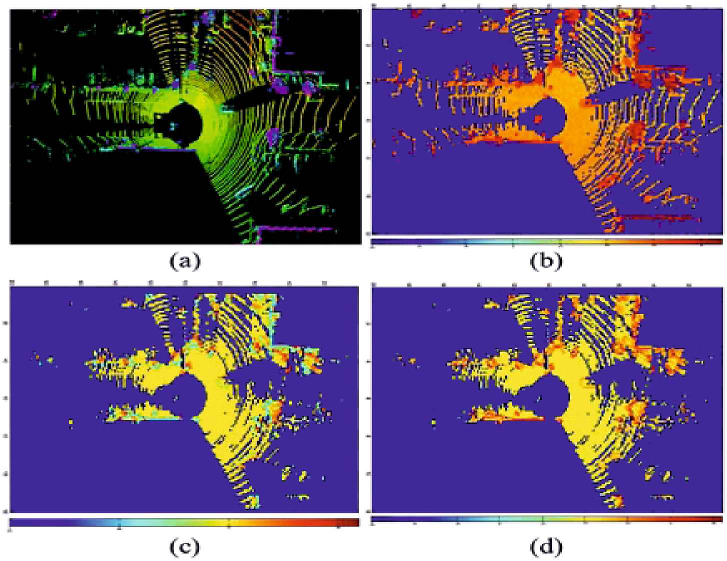


Fig. 2 Illustration of our method.(a) 3D point clouds aquired by Velodyne LIDAR. (b) cylindrical grid map. (c) extraction of feature values in each cell. (d) multi-labeling and classification with LBP.

- We propose a robust method of road detection with 3D data in undulated road such as down/uphill by the best selection of the proper features such as gradient value, height average of neighboring node as shown in Fig.2-(c).
- We formulate the road detection problem based on MRF with the loopy belief propagation to find the different regions with different classes as shown in Fig.2-(d).

3 3D Points Representation and Grid Map Building

3.1 Correction of Distorted 3D Point Cloud by Considering Vehicle Motion

As shown in Fig.3, the Velodyne LIDAR that is mounted on the top of the vehicle uses 64 lasers, which cover in different vertical angle, and it can also provide 360 degrees field of view for surrounding environment with more than 1.3 million points per second. The LIDAR returns deliver spherical point coordinates so it needs to transformation that data into Cartesian space. To do the transformation, we have to consider calibration parameters such as distance correction factor $\triangle r$, vertical/horizontal correction angle $\triangle\emptyset_v$,$\triangle\theta$,rotation angle φ,measured distance r and vertical/ horizontal offset r_v,r_h, The 3D point cloud computation in the cartesian coordinates is as below

$$p\,(x,y,z) = \begin{pmatrix} (r+\Delta r)\cdot\cos\left(\varphi+\Delta\theta\right)\cdot cos(\Delta\phi) + r_h\cdot cos(\varphi) \\ (r+\Delta r)\cdot\sin\left(\varphi+\Delta\theta\right)\cdot cos(\Delta\phi) + r_h\cdot sin(\varphi) \\ d\cdot sin\left(\varphi+\Delta\theta\right) - \ r_v \end{pmatrix}^T \tag{1}$$

Here, we have to consider that the scanner takes a no negligible amount of time to complete one rotation, so the observed 3D point clouds with LIDAR are distorted by the motion of the vehicle. For instance, if the speed of vehicle runs as about 100km/h (27.8 m/s) in the highway, it would be unfortunately given the distorted information past 2.7 meter on the every scan due to the fact that the scanner is rotating with a frequency of 10 Hz(0.1s). Furthermore, we should consider the situation that the vehicle passes over speed bumper with big rolling/pitch angle and then the front road is classified as obstacle due to the downward pitching of the vehicle. We are able to solve these problems by using information about the ego-motion of the car. In other word, the resulting frame is an approximation of how the environment would have looked like if the car had not moved.

To correct the distorted laser measurement by the vehicle's movement, we utilize a GPS/INS unit that provides highly the accurate motion information of the vehicle. Each laser measurement i during one revolution is referenced with respect to vehicle position and orientation O_{t+i} from the start of a sensor revolution, and afterwards transformed such that the coordinates are referenced with respect to $O_{t+\Delta t}$ at the end of the revolution. For transformation with rotation $\tilde{R}$ and translation $\overrightarrow{T}$ for each data, the undistorted coordinates $p^i_{O_{t+\Delta t}}$ of a point $p^i_{O_t}$ referenced with respect to $O_{t+\Delta t}$ as shown in Fig.3 can be calculated as follow as

$$p^i_{O_{t+\Delta t}} = \tilde{R}(p^i_{O_t} - \overrightarrow{T}) \tag{2}$$

3.2 Grid Map Building

The 3D point clouds which obtained by Velodyne LIDAR need to expensive costs to deal a large amount of data for real time processing, in our work we try to reduce it with using a 2.5D ego-centered cylindrical grid. Some relative works use the rectangular grid map projected by 3D point cloud points. Others approaches use

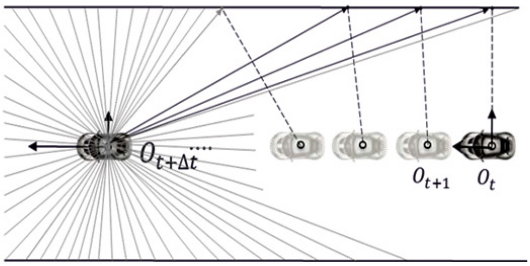


Fig. 3 Correction of distorted 3D point cloud by estimation the motion of vehicle

a mesh-grid map which directly is decomposed of a neighborhood graph from a scanner. Here, we focus on considering the manner a LIDAR scan, we know that it can give a different density of point depend on distance from vehicle despite the Velodyne sensor can take 3D scans of environment and provide millions of points per second. The points cloud will be gradually sparse from the near to the far. So we set two direction as direction and circular direction in cylindrical coordination. We can know that the gradient value is dramatically changed at some place for object/structure along the circular scan direction. Therefore, it is can be separated the coverage area with more high resolution along the circular direction than radial direction as shown in the Fig.4

4 Feature Extraction and Road Classification

4.1 Features(Gradient Value and Height) Extraction

We assume that the road surface is continuous and there is high correlation between neighborhood data. Therefore, given the world coordinates of the 3D point clouds, the gradient value at each node can be computed by using known neighboring nodes. To get the gradient value at each node as shown in Fig. 4, we need to height of neighboring nodes along the radial direction and circular direction. This gradient value $G_m(p)$ can reflect the geometrical character of roads. To obtain neighborhood points, it searches for closest nodes that have height values in four directions along the radial axis and circular axis respectively. We denote them as z_m^{c1} ,z_m^{c2} ,z_m^{r1} , z_m^{r2} ,where c means circular direction and r is radial direction.

The gradient of radial direction can be computed as

$$G_m^r(p) = \frac{z_m^{r1} - z_m^{r2}}{\|P^{r1} - P^{r2}\|} \tag{3}$$

Next, the gradient of circular direction can also be computed as

$$G_m^c(p) = \frac{z_m^{c1} - z_m^{c2}}{\|P^{c1} - P^{c2}\|} \tag{4}$$

where p is referred in the cylindrical coordinate. The gradient value is described by

$$G_m(p) = \sqrt{G_m^r(p)^2 + G_m^c(p)^2} \tag{5}$$

The height average of neighborhood nodes can be described as follows

$$H(p) = \frac{1}{n}\left(\sum_n Z_n\right) \tag{6}$$

where Z_n is an average of height value on the neighborhood nodes surrounding current node p. Finally, we describe a feature function as follow

$$g(p) = \alpha G_m^*(p) \cdot H(p)^* \tag{7}$$

The α is weight constant ane $G_m^*(p)$ and $H(p)^*$ are normalized with $G_m(p)$ and $H(p)$.

4.2 Classification Based on MRF

The goal of this step is to find the different regions with different classes and inference the area that the LIDAR does not cover. We take a graph-based approach for classification. Let $G = (V, E)$ be an undirected graph with nodes $v_i \in V$, the set of elements to be segmented, and edges $(v_i, v_j) \in E$ in corresponds to pairs of neighboring nodes. Each edge has a weight $w(v_i, v_j)$ which is a non-negative measure of the dissimilarity between neighboring elements v_i and v_j. We present the classification problem as LBP approach for performing inference on MRF as formed by the standard 4-connected neighborhood system since it models the spatial interactions present in the scene so that the labels of the points are determined jointly.

Let P be a set of node in the cylindrical grid map and L be as set of labels. The labels correspond to quantities that we want to estimate at the each node, such as gradient value and height average of neighborhood. A labeling f assigns a label $f(v_i) \in L$ to each node $v_i \in V$. We assume that the labels should vary slowly almost everywhere but may change dramatically at some places such grids along object boundaries. The quality of a labeling is given by an energy function,

$$E(f) = \sum_{(v_i, v_j) \in E} V\left(f(v_i), f(v_j)\right) + \sum_{(v_i) \in V} D(f(v_i)) \tag{8}$$

Where E are the edges in the four-connected grid graph. $V\left(f(v_i), f(v_j)\right)$ is the cost of assigning labels $f(v_i)$ and $f(v_j)$ to two neighboring nodes, and it is referred to as the discontinuity cost. $D(f(v_i))$ is the cost of assigning label $f(v_i)$ to node v_i, which is referred to as the data cost.

Finding a labeling that minimizes this energy corresponds to the maximum a posteriori (MAP) estimation for MRF in the form of Eq.8. Normally this algorithm is defined in terms of probability distributions, but an equivalent computation can be

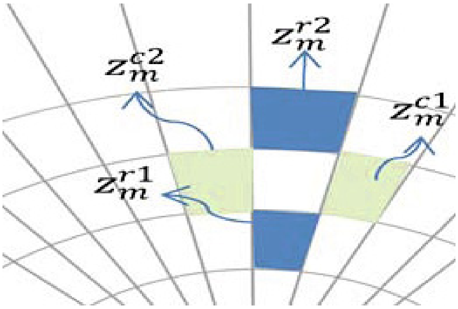


Fig. 4 The computation of gradient with neighboring nodes

performed with negative log probabilities, where the max-product becomes a min-sum. We use this formulation because it is less sensitive to numerical artifacts, and it uses the energy function definition more directly.

The max-product BP Algorithm works by passing message around the graph defined by the four-connected grid. Each message is a vector of dimension given by the number of possible levels. Let $m^t_{v_iv_j}$ be the message that node v_i sends to a neighboring node v_j at time t. When using negative log probabilities all entries in $m^0_{v_iv_j}$ are initialized to zero, and at each iteration new messages are computed in the following way,

$$m^t_{v_iv_j}\left(f(v_j)\right)=min_{f(v_i)}\left(V\left(f(v_i),f(v_j)\right)+D\left(f(v_i)\right)+\sum_{N(v_i)v_j} m^{t-1}_{v_iv_j}\left(f(v_j)\right)\right) \tag{9}$$

Where $N(v_i)\backslash v_j$ denotes the neighbors of v_i other than v_j. After T iterations a belief vector is computed for each node,

$$b_{v_j}\left(f(v_j)\right)=D_{v_j}\left(f(v_i)\right)+\sum_{p\in N\left(v_j\right)} m^T_{v_iv_j}\left(f(v_j)\right) \tag{10}$$

Finally, the $f^*_{v_j}$ label that minimizes $b_{v_j}\left(f(v_j)\right)$ individually at each node is selected.

In the work, the labels correspond to different gradient value and height average that should be assigned to grids in the map. Thus the data costs can be defined as

$$D\left(f(v_i)\right) = \min(\left\||g(v_i)|-f(v_i)\right\|,\tau) \tag{11}$$

We use a truncated step function for the data cost, τ is a truncation value, $g(v_i)$ is the feature of node v_i.The truncation makes the data cost robust to abnormally large feature values.

Another class of cost functions is based on the degree of difference between labels. The cost of assigning a pair of labels to neighboring node is generally based on the amount of difference between these quantities. In order to allow for discontinuities, as the values are not smoothly changing everywhere, the cost function should be robust, becoming constant as the difference become large. So it can be used the truncated linear model, where the cost increases linearly based on the distance between the labels $f(v_i)$ and $f(v_j)$ up to some level,

$$V\left(f(v_i),f(v_j)\right) = \min\left(s\left\|f(v_i)-f(v_j)\right\|,d\right) \tag{12}$$

Where s is the rate of increase in the cost, and d controls when the cost stops increasing.

Fig. 5 An experimental vehicle with 3D Velodyne sensor

5 Experimental Result

We have evaluated the proposed algorithm using both a publicly available dataset[4], [13] as well as numerous scans acquired by the HDL-64E sensor mounted on an experimental vehicle(Hyun-dai Sorrento) in inner city traffic scenes as shown Fig.6. As no ground truth information is available, a qualitative performance evaluation is conducted.

For cylindrical grid map, we use $\Delta\theta = 0.5$ and$\Delta r = 0.2m\,(range : 0 \sim 30m)$ and $\Delta r = 0.5m\,(range : 30 \sim 60m)$ and $\Delta r = 1m\,(outof 60m)$throughout all experiments. For classification based on MRF, we set 10 as number of labels and the truncation value were respectively fixed to $\tau = 5$ and $d = 3$.

Since the classification of the 3D point clouds in normal road environment is well demonstrated. So we focus on a variety of challenging environment. Fig.7 shows example result of a slope road, an uphill road and a downhill road, which substantiated that the proposed method is more robust and reliable than the conventional approach based on the height.

As visible in the first column, though the road is able to show normally flat in Fig7 (a), we can see that laser returns corresponding to the area are irregular as Fig.7 (e). As shown in Fig.7 (i), the conventional approach based on the height can give the wrong result that drivable space is as obstacle region, as indicated by the red circle in the figure. Whereas the proposed approach successfully classify the slope area as the drivable with feature values such as gradient and average height of road space as shown in Fig.7 (m). Furthermore, the spatial interactions based on the smoothness term in the MRF can also ensure the local consistency in such scenarios so that all of the rough region will be classified into to same category, even when some of the gradient are abnormal due to noise. Besides, we can see in the second column, when our vehicle drives on the slope road which is more high right than the left side, we can see that the result of the convention approach misrecognizes partially some road and some vehicles as obstacles indicated by the red circle in the Fig.7 (j). But our proposed method gives the robust result of detection according to this height variance of road because of shown in the Fig.7 (n). Also we can see that some area that the LIDAR does not cover is interpolated by inference algorithm based on MRF with the loopy belief propagation through the comparison of area indicated by the red circle in the Fig.7 (k) and in the Fig.7 (o). As shown in the fourth column, it is caused misunderstanding that there is a big obstacle in front of vehicle on the road by the conventional approach based on the height in the Fig.7

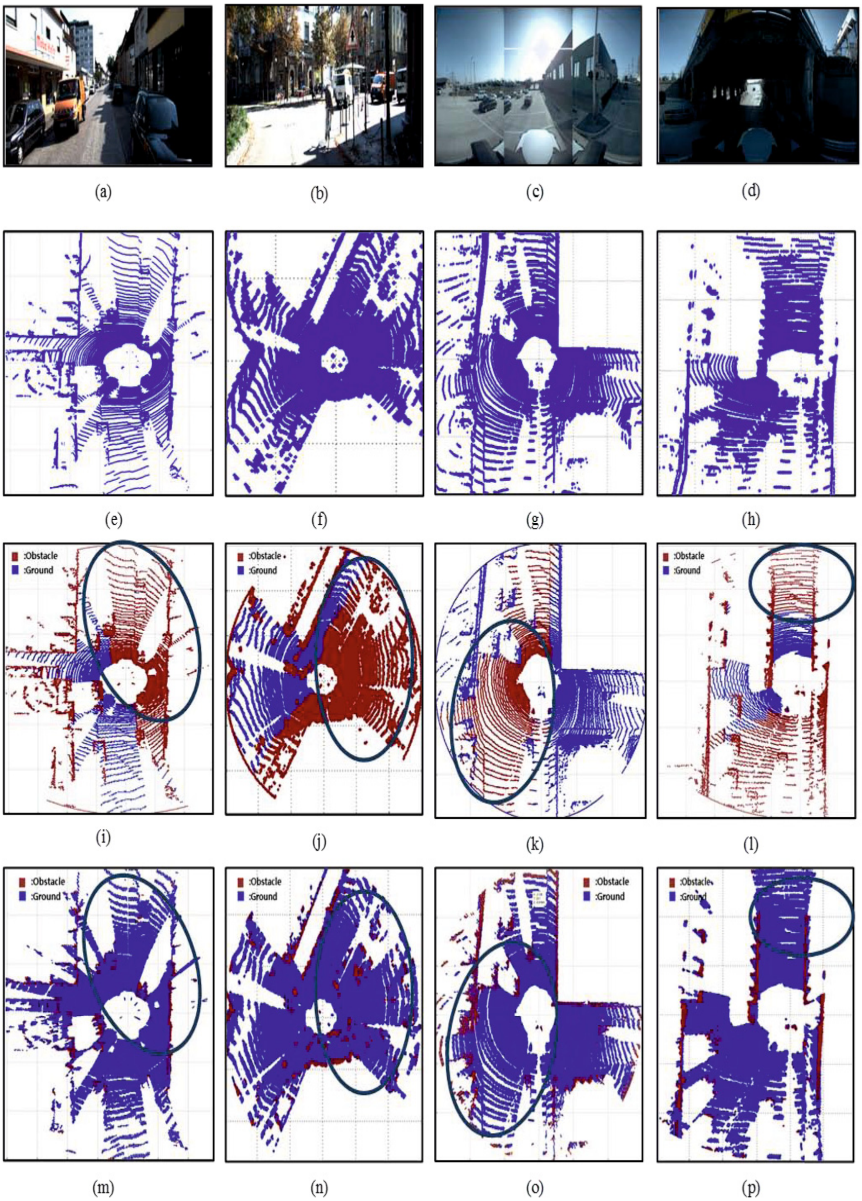


Fig. 6 Classification result of a variety of environment(flat road, slope road, down hill ,up-hill,), the pictures of environment(first row), the 3D point clouds by LIDAR , the result of conventional approach (third row), the result of our proposed work(fourth row)

(l). However the proposed approach successfully classified the road as the drivable space since our feature values are in the drivable space for vehicle in the Fig.7 (p).

6 Conclusion

In this paper, we have presented a robust method of road detection with 3D point clouds on the challenging road environments such as down/uphill, sloped road. Our first contribution is correction of 3D point cloud distorted by the motion of vehicle incorporating vehicle posture information. Our second contribution is guideline for the best selection of the proper features such as gradient value, height average of neighboring node. Our third contribution is transformation of the road detection problem into a classification problem of different features. Our fourth contribution is inference algorithm based on MRF with the loopy belief propagation for the area that the LIDAR does not cover. In experiments, we use a publicly available dataset as well as numerous scans acquired by the HDL-64E sensor mounted on experimental vehicle in inner city traffic scenes. The results proved that the proposed method is more robust and reliable than the conventional approach based on the height on the variety of challenging environment. Our future work will focus on the detection of dynamic road environment with the supervised/unsupervised learning approach and the fusion of the LIDAR and vision data.

Acknowledgements. This work was supported by the IT R/&D program of MSIP/KEIT. [KI0041417, Development of Decision Making/ Control Technology of Vehicle/Driver Cooperative Autonomous Driving System (Co- Pilot) Based on ICT].

References

[1] Bohren, J., Foote, T., Keller, J., Kushleyev, A., Lee, D., Stewart, A., Vernaza, P., Derenick, J., Spletzer, J., Satterfield, B.: Little ben: The ben franklin racing team's entry in the 2007 darpa urban challenge. Journal of Field Robotics 25(9), 598–614 (2008)

[2] Buehler, M., Iagnemma, K., Singh, S. (eds.): The DARPA urban challenge: Autonomous vehicles in city traffic. STAR, vol. 56. Springer, Heidelberg (2009)

[3] Douillard, B., Underwood, J., Kuntz, N., Vlaskine, V., Quadros, A., Morton, P., Frenkel, A.: On the segmentation of 3d lidar point clouds. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 2798–2805. IEEE (2011)

[4] Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. International Journal of Robotics Research, IJRR (2013)

[5] Guo, C., Mita, S., McAllester, D.: Hierarchical road understanding for intelligent vehicles based on sensor fusion. In: 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 1672–1679. IEEE (2011)

[6] Guo, C., Sato, W., Han, L., Mita, S., McAllester, D.: Graph-based 2d road representation of 3d point clouds for intelligent vehicles. In: 2011 IEEE Intelligent Vehicles Symposium (IV), pp. 715–721. IEEE (2011)

[7] Himmelsbach, M., Luettel, T., Wuensche, H.-J.: Real-time object classification in 3d point clouds using point feature histograms. In: IEEE/RSJ International Conference on

Intelligent Robots and Systems, IROS 2009, pp. 994–1000. IEEE (2009)

[8] Leonard, J., How, J., Teller, S., Berger, M., Campbell, S., Fiore, G., Fletcher, L., Frazzoli, E., Huang, A., Karaman, S., et al.: A perception-driven autonomous urban vehicle. Journal of Field Robotics 25(10), 727–774 (2008)

[9] Li, M., Li, Q.: Real-time road detection in 3d point clouds using four directions scan line gradient criterion. Future, 5 (2009)

[10] Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Hoffmann, G., Huhnke, B., et al.: Junior: The stanford entry in the urban challenge. Journal of Field Robotics 25(9), 569–597 (2008)

[11] Moosmann, F., Pink, O., Stiller, C.: Segmentation of 3d lidar data in non-flat urban environments using a local convexity criterion. In: 2009 IEEE Intelligent Vehicles Symposium, pp. 215–220. IEEE (2009)

[12] Nguyen, T.-N., Michaelis, B., Al-Hamadi, A., Tornow, M., Meinecke, M.: Stereo-camera-based urban environment perception using occupancy grid and object tracking. IEEE Transactions on Intelligent Transportation Systems 13(1), 154–165 (2012)

[13] Pandey, G., McBride, J.R., Eustice, R.M.: Ford campus vision and lidar data set. International Journal of Robotics Research 30(13), 1543–1552 (2011)

[14] Strom, J., Richardson, A., Olson, E.: Graph-based segmentation for colored 3d laser point clouds. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2131–2136. IEEE (2010)

[15] Tay, M.K., Mekhnacha, K., Chen, C., Yguel, M.: An efficient formulation of the bayesian occupation filter for target tracking in dynamic environments. International Journal of Vehicle Autonomous Systems 6(1), 155–171 (2008)

[16] Thrun, S.: Learning occupancy grid maps with forward sensor models. Autonomous Robots 15(2), 111–127 (2003)

[17] Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M.N., Dolan, J., Duggins, D., Galatali, T., Geyer, C., et al.: Autonomous driving in urban environments: Boss and the urban challenge. Journal of Field Robotics 25(8), 425–466 (2008)

[18] Wedel, A., Badino, H., Rabe, C., Loose, H., Franke, U., Cremers, D.: B-spline modeling of road surfaces with an application to free-space estimation. IEEE Transactions on Intelligent Transportation Systems 10(4), 572–583 (2009)

Predicting Terrain Traversability from Thermal Diffusivity

Chris Cunningham, Uland Wong,
Kevin M. Peterson, and William L. "Red" Whittaker

Abstract. This paper presents a method to predict soil traversability by estimating the thermal diffusivity of terrain using a moving, continuous-wave laser. This method differentiates between different densities of the same material, which vision-based methods alone cannot achieve. The bulk density of a granular material has a significant effect on both its mechanical behavior and its thermal properties. This approach fits the thermal response as effected by a laser to an analytical model that is dependent on thermal diffusivity. Experimental soil strength measurements validate that thermal diffusivity is a predictor of traversability for a given material.

1 Introduction

This paper presents a technique for determining terrain traversability from measurements of thermal diffusivity. Classical perception approaches detect material shape and appearance, but cannot measure the underlying properties that determine traversability. The inability to characterize these non-geometric properties is a primary cause of robotic failure on Mars, the Moon, and Earth. Spirit ended its mission mired in soft soil; Lunokhod was entrapped by loose soil while entering a crater [1]. Means to predict these conditions would transform how planetary rovers operate, increasing both safety and efficiency.

This research seeks to predict the ability of a ground vehicle to traverse a granular soil by sensing its thermal properties. The mechanical response of a granular soil to a wheel is primarily governed by particle size distribution, particle shape, bulk

Chris Cunningham · Uland Wong · Kevin M. Peterson · William L. "Red" Whittaker
Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213
e-mail: {ccunningham,red}@cmu.edu,
uyw@andrew.cmu.edu,kp@cs.cmu.edu

Kevin M. Peterson
Astrobotic Technology Inc., 2515 Liberty Ave., Pittsburgh, PA 15222
e-mail: kevin.peterson@astrobotic.com

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_5, © Springer International Publishing Switzerland 2015

density, and relative density [2]. Shape and density distribution are principal parameters determining the range of soil strength while bulk and relative density modulate strength [3] [4]. Compact materials with higher bulk density are much stronger than loose materials with low bulk density and are therefore more traversable by a wheeled vehicle.

Bulk density and porosity of a soil also have a strong influence on the specific heat, volumetric heat capacity, and thermal diffusivity of granular media [5]. Compact soils with higher bulk density and lower void ratio conduct heat more easily between particles . Likewise, loose soils with lower bulk density and higher void ratio have lower thermal conductivity and diffusivity [6] [7]. Therefore, because material density influences both the mechanical and thermal properties of granular materials, thermal diffusivity is correlated with the mechanical behavior and traversability of a soil.

This paper presents a method for detecting the difference between traversable, compact soil and loose, hazardous soil by remotely estimating thermal diffusivity of terrain. A continuous-wave laser and a thermal camera are co-located, pointed at a granular material, and translated with respect to that material. The laser introduces a thermal transient as it moves across the terrain. The thermal camera observes the resulting temperatures.

Section 2 discusses related work in non-geometric hazard detection and thermal diffusivity estimation. Section 3 presents an analytical model to estimate thermal diffusivity based on the transient temperature response of a granular material to heat flux from a moving laser. Specifics of the experiments and their results are presented in Section 4 and analyzed in Section 5. In Section 6, theoretical correlations of diffusivity to bulk density and traversability are validated using soil strength measurements. Section 7 discusses conclusions and directions for future research.

2 Related Work

Prior work on non-contact identification of non-geometric terrain hazards has primarily focused on vision-based methods. These methods have shown promise but are limited to sensing surface appearance, which is not necessarily correlated with the bulk characteristics of a material. This research enables differentiation between different preparations of the same soil, which can have very different interactions with wheels though they may appear identical on the surface. It builds upon prior research in photothermal radiometery to develop a method for diffusivity estimation that is viable for integration into an autonomous vehicle.

Visual techniques have been demonstrated for traversability prediction in planetary-like environments. Helimick, Angelova, and Matthies use color, texture, and depth from stereo imagery to classify terrain and predict wheel slip. The classes are determined by an encoding of domain knowledge [8]. Brooks and Iagnemma also use visual cues to classify terrain but terrain classes are determined using self-supervised learning [9]. These vision techniques operate on the principle that terrain with similar appearance has a similar response. Terrain response (either by driving or soil testing) is measured and visual appearance in the area of those measurements is recorded. Based on this information, learning techniques are used to map upcoming appearance to likely terrain response.

Although vision-based predictive models progress towards alerting robotic systems to variation in terrain, they are fundamentally limited. The surface appearance of a patch of terrain is not necessarily directly correlated with its bulk physical characteristics. While appearance can be used to associate similar terrain, it cannot measure important characteristics such as compaction, which is a critical factor in determining shear strength. As a result, terrain patches with similar appearance, but dissimilar compositions, and therefore traversability, have an ambiguous classification when based on visual data alone.

In the applied physics community, the problem of estimating the thermal diffusivity of a material is well researched. Photothermal radiometry is a widely used technique for non-contact estimation of the thermal properties of thin films using an infrared detector and a single laser flash [10] [11]. This method is most effective with a thin film but has been applied to layered materials as well as powders where it can detect a difference between loose and consolidated powders [12]. Though these methods are effective in a laboratory environment, they require precisely-calibrated, sensitive instruments that are not feasible to implement on mobile robots.

Multi-spectral imaging in visible and IR wavelengths has been used for terrain classification with demonstrated success in identification of vegetation [13]. In addition, thermal imaging from a Mars orbiter has been used to estimate the thermal inertia in order to estimate mechanical properties of soil in potential Mars Exploration Rover landing sites [14]. Unfortunately, the resolution of thermal images from orbiting satellites is too low for reliable application to rover mobility.

This research is distinct from the methods above in two important ways. First, this approach probes deeper than vision-based methods alone that are limited to prediction solely from observation of surface appearance. Second, this method for thermal diffusivity measurement is viable for mobile robots. This method does not require the highly calibrated experimental setups used in photothermal radiometry for thin films. In addition, it provides high resolution at the scale of a robot that satellite imagery cannot.

3 Thermal Diffusivity Estimation

The approach for thermal diffusivity estimation is macroscopic measurements of the transient temperature response caused by a low-power, continuous-wave, semiconductor laser. The laser is pointed at the soil while a thermal camera measures the temperature response of the terrain to the laser excitation. The camera and the laser are translated linearly, parallel to the ground at a constant velocity. The thermal diffusivity of the soil is estimated by fitting parameters of a known model to the transient thermal response of the material to the laser.

3.1 Analytical Model

The mathematical model of the thermal response is derived from the three-dimensional heat diffusion equation, which governs heat flow.

$$\frac{\partial\theta}{\partial t} = k\left(\frac{\partial^2\theta}{\partial x^2} + \frac{\partial^2\theta}{\partial y^2} + \frac{\partial^2\theta}{\partial z^2}\right) \tag{1}$$

θ is the temperature at a point (x,y,z) in a Cartesian coordinate system, and k is the thermal diffusivity of the material.The material under test is modeled as a semi-infinite plane extending from $z = 0$ in the negative z direction. The diffusion equation is subject to a Neumann boundary condition at $z = 0$.

$$\frac{\partial\theta}{\partial z} = 0, z = 0 \tag{2}$$

This boundary condition makes the assumption that there is no heat lost at the sample surface[11]. The incident heat from the laser is modeled as a Gaussian instead of a uniform distribution, which is more physically accurate and allows for some helpful mathematical simplifications [15].

$$Q(x',y',z',t') = \frac{P}{2\pi r^2} exp\left(-\frac{(x'-vt')^2 + y'^2}{2r^2}\right)\delta(z') \tag{3}$$

Q represents a Gaussian laser in the plane $z = 0$ moving along the x-axis at $y = 0$. P is the total power from the laser absorbed by the material, v is the velocity of the laser in the x direction, and r is the radius of the laser spot. Because this problem is addressed macroscopically and the absorption depth of the laser is very shallow, it is assumed that all of the power from the laser is absorbed at the surface of the material [16]. The Green's function for a point source in three-dimensions subject to the boundary condition given above is used to find the equation for the temperature at a point (x,y,z). This Green's function is given by Carslaw and Jaeger [17] and represents the temperature of a point (x,y,z,t) in reponse to a unit point source at (x',y',z',t').

$$G(x,y,z,t,x',y',z',t') = \frac{1}{4(\pi k(t-t')^{\frac{3}{2}}} exp\left(-\frac{(x-x')^2 + (y-y')^2 + (z-z')^2}{4k(t-t')}\right) \tag{4}$$

The equation for the heat flow from the laser is used in conjunction with the Green's function to find the temperature at any (x,y,z,t) due to the laser excitation. An offset θ_0 is added to represent the initial temperature before laser excitation.

$$\theta(x,y,z,t) = \int_{-\infty}^{t}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} Q(x',y',z',t')G(x,y,z,t,x',y',z',t')\,dz'dy'dx'dt' + \theta_0 \tag{5}$$

This solution is simplified using the recipricocity and translation properties of Green's functions and the fact that the integrals over $x\prime$ and $y\prime$ can be simplified to gaussian forms, which result in known solutions. The final result is a formula for the temperature at $(x, y, z, 0)$ [18].

$$\theta(x,y,z,0) = P\int_0^\infty \frac{exp\left(-\frac{(x+vt')^2+y^2}{2r^2+4kt'} - \frac{z^2}{4kt'}\right)}{\sqrt{\pi^3 kt'}(2r^2+4kt')} dt' + \theta_0 \qquad (6)$$

Thus, the temperature at $(x, y, z, 0)$ is only a function of a few variables. θ_0 can be easily measured from a thermal camera before the laser excitation. In this paper, v is experimentally controlled. r is estimated by calibrating the laser. The two unknown variables are the thermal diffusivity of the material under test, k, and the amount of power absorbed by the material, P, which is a function of both the laser and the material.

3.2 Diffusivity Estimation

To estimate the thermal diffusivity constant of a material, data is extracted from a thermal image to directly correspond to θ_x, which is the model (6) evaluated at $y = 0$, $z = 0$, and $t = 0$.

$$\theta_x(x; x_{off}, P, k) = P\int_0^\infty \frac{exp\left(-\frac{(x+x_{off}+vt')^2}{2r^2+4kt'}\right)}{\sqrt{\pi^3 kt'}(2r^2+4kt')} dt' + \theta_0 \qquad (7)$$

An example thermal image is shown in Figure 1. The maximum temperature in the y direction occurs along the x-axis at $y = 0$, so it is straightforward to extract the maximum temperature in the y direction for every pixel column. These are $(x, \theta_x(x))$ data points, where translational alignment of the x values with the model is still ambiguous. To align the data points with the model, all x values are translated so that $x = 0$ corresponds to the maximum θ_x value. Pixel distances are scaled to the equivalent linear distance on the material.

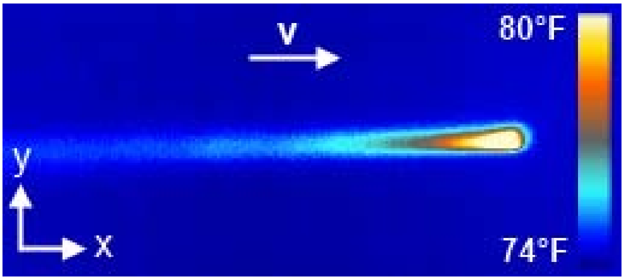


Fig. 1 Thermal transient produced by a 100mW 532nm laser being translated at a constant velocity on a loose preparation of JSC-1A. **v** shows the direction of the motion of the laser.

Unfortunately, assuming that the maximum occurs at $x = 0$ is only an approximation. With low velocity, high diffusivity, and a small laser spot radius, the temperature response is nearly symmetric about the y-axis with a maximum at $x = 0$. However, as the quantity vr/k increases, the response becomes less symmetric, and the maximum of the curve shifts towards more negative values of x. Thus the maximum value in the x-direction is not easily predicted as it is dependent on the velocity and radius of the laser as well as the diffusivity of the material under test [19] [15] [20]. For the velocities, diffusivities, and radii considered in this paper, the maximum occurs close enough to make the approximation that it is at $x = 0$. In (7), x_{off} is an offset value used to compensate for errors in x-alignment. First is the error that occurs through the assumption that the position of the maximum temperature corresponds to $x = 0$, when in fact it does not. Second is error that arises when the resolution of the camera is not high enough to precisely capture the x value of the maximum temperature of the thermal response.

The resulting (position,temperature) data points are used to fit (7). The three parameters are x_{off}, P, and k. θ_0 is estimated from the mean temperature of the surrounding material. r is estimated before the experiment through analysis of a laser pulse [21]. v is known as it is controlled by the experimental setup. An optimization algorithm (e.g. Nelder Mead) is used to minimize the root mean square error (RMSE) between the experimental data and (7) evaluated at estimates of x_{off}, P, and k.

$$(x_{off}, P, k) = \underset{x_off,P,k}{\operatorname{argmin}}(RMSE(\theta_x(x; x_{off}, P, k))) \tag{8}$$

Thermal diffusivity, k, can then be used to detect when the material type or density of the terrain has changed. For a given material, a lower diffusivity corresponds to a lower bulk density and therefore less traction. Likewise, a higher diffusivity corresponds to a higher bulk density and therefore more traction. Thus, by measuring the thermal response of terrain to a laser excitation, thermal diffusivity can be estimated and used as a predictor of traversability.

4 Experimental Diffusivity Measurements

Several experiments were conducted to estimate thermal diffusivity using the method presented in this paper. A thermal camera is used to capture the transient temperature response caused by a moving laser. Recorded thermal images are then analyzed to produce estimates for thermal diffusivity on both loose and compact granular materials.

Three lunar regolith simulants were used for these experiments, JSC-1A, BP-1, and GRC-1. JSC-1A is the gold standard for lunar regolith simulants and is widely available for research purposes [22]. BP-1 is a lunar regolith simulant that is very similar in its major elements to JSC-1A. However, its minor elemental composition precludes it from closely simulating lunar regolith chemical composition, which has a large impact on thermal properties. It does, however, closely simulate the geotechnical properties including particle size and shape distribution [23]. GRC-1 is a lunar regolith simulant specifically developed to be an inexpensive simulant of the mobility properties of lunar terrain and does not replicate all of the mechanical properties nor the chemical composition. It does not have a wide range of compaction and has a low thermal diffusivity compared to the other two simulants [24].

All three simulants were used in both loose and compact preparations. Loose preparation consisted of pouring the material into the sample container, hand agitation with a shovel, and gentle leveling to create a flat surface. Compact preparation utilized a hydraulic press on a flat steel plate top until maximum pressure was achieved as shown in Figure 2.

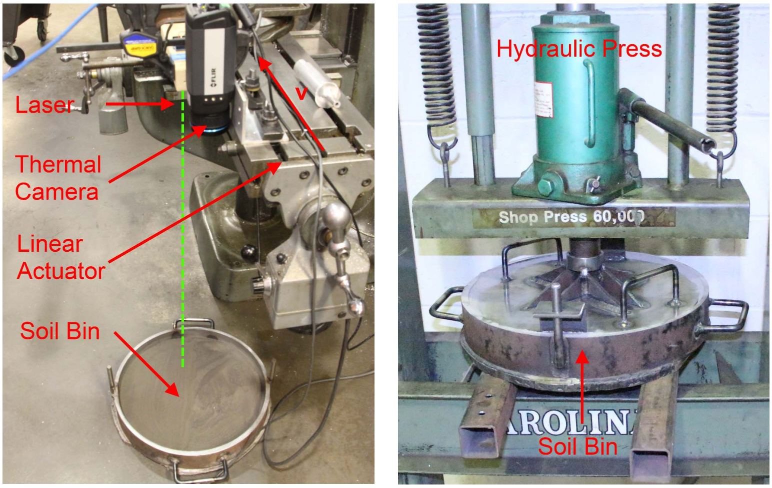


Fig. 2 Experimental setup for thermal diffusivity experiments (left). A 100mW 532nm continuous-wave laser and a thermal camera are mounted to a linear actuator and translated a constant velocity parallel to a soil sample below. A hyraulic press (left) and compressable soil bin with lid are used to repeatably prepare compressed soil samples.

Data to validate thermal diffusivity estimation was collected using a controlled setup shown in Figure 2. A thermal camera and a continuous-wave 100mW 532nm laser were mounted to a linear mill and pointed down at a soil bin containing either a loose or compact simulant. The thermal camera recorded images at 7hz. The mill was driven at 2.5 mm/s, 3.8 mm/s, 5.1 mm/s, and 6.4 mm/s. The laser point was translated .25 m, from one end of the soil bin to the other.

The experimental data was fit to the model (7) using the parameters k, P, and x_{off}. Example curves are shown in Figure 3. The blue points are the measured temperatures and the red line is the analytical model.

Four runs at different velocities were averaged together provide estimated values for k and P for loose and compact preparations of all three simulants. These values are presented in Table 1 along with the average RMSE between the model and the

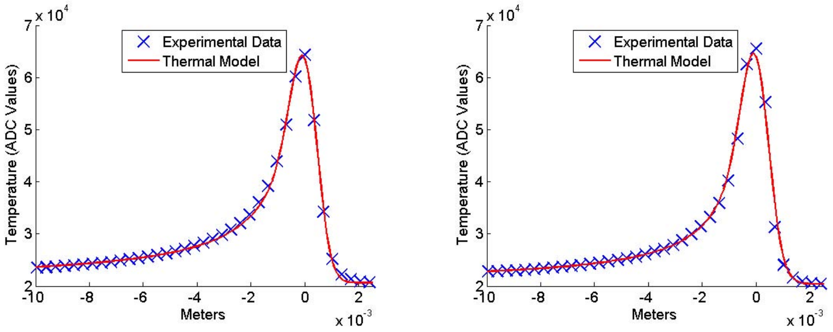


Fig. 3 Comparison of experimentally measured temperatures and temperatures estimated from the theoretical model for loose (left) and compact (right) preparations of JSC-1A at a laser speed of 2.5mm/s.

Table 1 Estimated thermal diffusivity (k), absorbed power (P), and RMSE between model and experimental data averaged over four laser velocities

Material	k (m^2/s)	P (W)	RSME
JSC-1A Loose	2.77×10^{-7}	5.78×10^{-5}	243
JSC-1A Compact	6.32×10^{-7}	9.60×10^{-5}	418
BP-1 Loose	5.56×10^{-7}	8.01×10^{-5}	180
BP-1 Compact	7.83×10^{-7}	8.40×10^{-5}	523
GRC-1 Loose	4.49×10^{-8}	3.70×10^{-6}	86
GRC-1 Compact	6.91×10^{-8}	$3.90\ \times10^{-6}$	73

experimental data. A bar graph showing a comparison between estimated diffusivity values for loose and compact soil is shown in Figure 4 for each simulant. The x-offset (x_{off}) averaged around 0.15 mm with a maximum of 0.4 mm. These low values are to be expected given that a pixel is 0.34 mm wide, and x_{off} compensates for the combination of error from pixel resolution and error caused by the theoretical maximum x value not being exactly 0.

5 Analysis

In all three simulants, this method produces a clear, quantifiable difference between loose and compact preparations of the same material. There are 56%, 29%, and 35% measured differences in diffusivity from the compact preparation to the loose preparation for JSC-1A, BP-1, and GRC-1, respectively. This data confirms the results expected from the theoretical model. At the speeds considered in this paper, translational velocity had no significant effect on estimated diffusivity. For example, for loose JSC-1A, (shown in Figure 5) there is only a 9.3×10^{-9} (2.1%) difference between the maximum and minimum estimated values. This error is within an expected range due to errors in measurement of velocity.

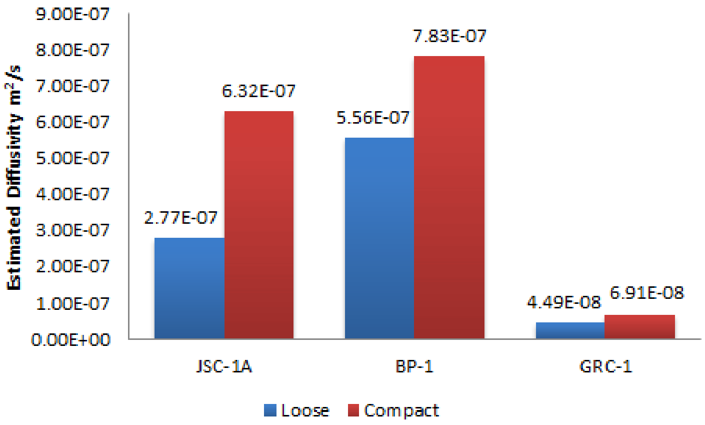


Fig. 4 Estimated thermal diffusivities (k) of JSC-1A, BP-1, and GRC-1 averaged over four laser velocities. Demonstrates a measurable difference between compact and loose granular media.

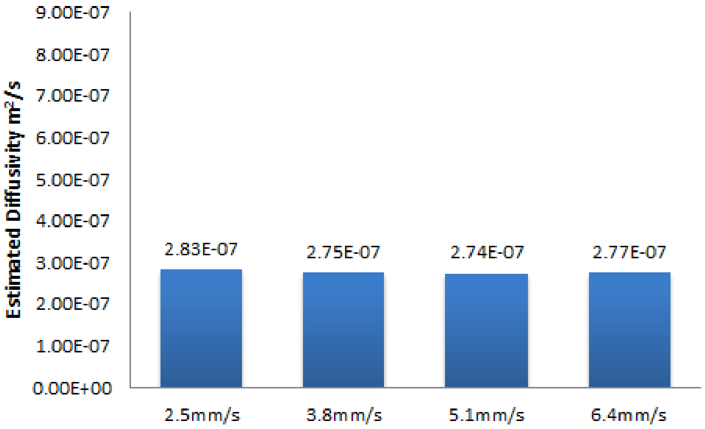


Fig. 5 Comparison of thermal diffusivity estimates for a loose preparation of JSC-1A at four different laser speeds (2.5mm/s, 3.8mm/s, 5.1mm/s, and 6.4mm/s)

As is evident in Table 1, there was some variation in P with the density of the material. This is likely due to the preparation of the soil and surface roughness, which can affect the amount of power absorbed. These results suggest that the absorbed power must be a parameter in the transient thermal model since it is dependent on properties of the material under test and difficult to predict with only a priori knowledge. The cause of this effect requires further investigation.

While the thermal diffusivity estimation method achieved similar results in each of the four trials for JSC-1A, there was slightly more variation in BP-1 and GRC-1.

The variation in BP-1 was low and can be attributed to the fact that it was a non-homogeneous material with small rocks littered throughout the soil, which causes variation in its thermal properties. The variation in GRC-1 was higher. For two of the four trials, the regression algorithm found a local minimum with an incorrect value for the absorbed heat, P. When the algorithm was adjusted to limit the possible values of P for GRC-1 to a smaller interval, the method performed consistently. This error was likely caused by a lower signal to noise ratio in the signal since the temperature change in GRC-1 was significantly lower than in the other two materials. In addition, the effective spot size of the laser on GRC-1 was significantly higher than on JSC-1A and BP-1, where the spot size was close to the a priori estimated size. This is likely caused by more scattering of the light on the surface of GRC-1 than on the other two materials.

6 Validation with Soil Strength Measurements

Correlation between diffusivity and traversability is validated with empirical testing using a bevameter instrument to measure soil strength (shown in Figure 6), which emulates a mobility archetype. Bevameter measurements are a well-known technique for predicting the response of terrain to a ground vehicle. A bevameter performs two primary functions with high repeatability. Firstly, a sinkage test presses a flat circular plate into a soil sample while recording ground pressure exerted and linear displacement. Secondly, a shear test presses and rotates a toothed annulus while recording pressure, torque, and displacement. The intent of these functions is to mimic how a robot might sink or slip while negotiating a material [25]. As such, the end effectors are sized to reproduce the ground contact area and traction of a specific wheel (or track) design. The recording of force-displacement data produces a curve that spans robot weights and predicts sink or slip given the wheel design.

This work focused on the pressure-sinkage aspect of bevameter testing. Sinkage provides primary resistance against forward locomotion and was the primary mobility entrapment of the Spirit rover [26]. Testing here emulated the mobility system of "Red Rover" a four-wheeled, solar-powered, lunar rover prototype (shown in Figure 6). The wheels of Red Rover are sized to 300mm in diameter and are 140mm wide. Rigid aluminum construction means that the wheels do not deform to terrain under normal loads. Rule of thumb estimates for the ground contact patch on loose soil give an area of 1832mm^2, which is equivalent to a 5 degree arc of the wheel. This corresponds approximately to a circular bevameter plate of 50 mm in diameter. The design mass of the vehicle is 100 kg, which results in a terrestrial ground pressure of 134 kPa or a force of 245N on each wheel.

Four pressure-sinkage trials were conducted on each of three test materials (BP-1, JSC-1A, GRC-1) under loose and compact preparation. In all tests, a sample approximately 18cm deep was utilized; due to edge effects, results for very high ground pressure vehicles ($>$ 400kPa) may be skewed for some materials. Curves for each of the parameters $\{$soil type, compaction level$\}$ were fit from independent trials using 2nd order polynomial regression.

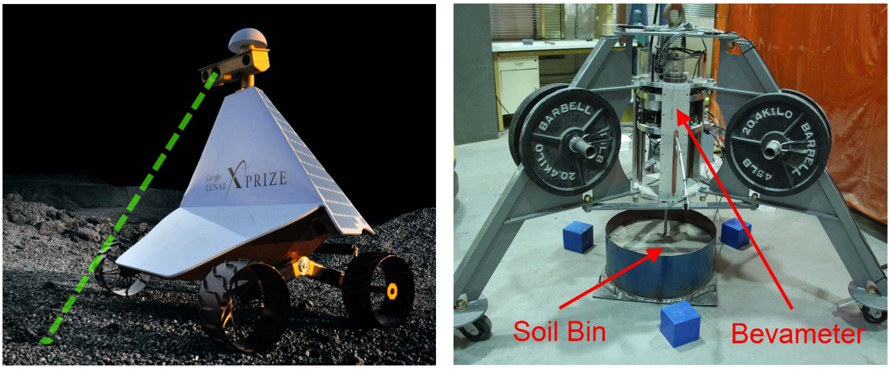


Fig. 6 A bevameter was used for measuring soil strength (right). The bevameter plate was selected to emulate Red Rover, a differential-drive lunar rover prototype (right)

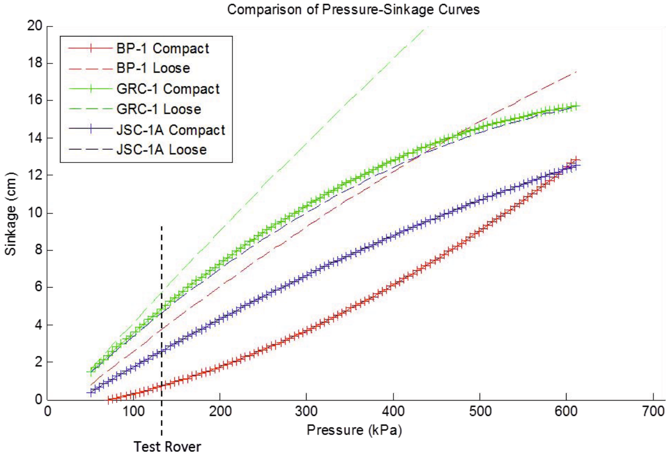


Fig. 7 Comparison of Pressure-Sinkage Curves. Curves generated for each material type and compaction level are shown. Data for the GRC-1 material is only valid for pressures under 400kPa; values beyond this are extrapolated for illustrative purposes. Performance for the rover archetype used in analysis is indicated by the vertical line at 135kPa.

Experimental data shows that the estimate of thermal diffusivity is a good predictor of material resistance to rover sinkage, which implies that it is also a good predictor of a material's traversability. Figure 7 shows the pressure-sinkage curves for all materials and compaction levels tested. A lower curve indicates a stronger material and easier mobility. Results generally have high certainty for pressures under 400kPa. In this range, compact BP-1 is empirically the strongest material, while both forms of GRC-1 were the weakest - resulting in greatest sinkage. The strength-order of material combinations here correspond directly to the estimated diffusivity. Results specific to the archetype rover examined in this paper are denoted with a vertical line in the graph above. It is noted that in these experiments, the sample size for materials is small and the differences in compaction level extreme. The authors do not make the leap to general applicability or existence of a linear diffusivity-strength relationship. However, these promising results warrant further investigation of the technique.

One point where bevameter data disagrees with diffusivity results is in the magnitude of phenomena resulting from compaction. It is useful to calculate the percent change in sinkage as a result of compaction, averaged over the entire range. This analysis shows that a robot would sink 235% more in loose BP-1 than in compact, 71% in JSC-1A and 26% in GRC-1. The percent changes in sinkage for JSC-1A and GRC-1 are similar to the percent changes in diffusivity. The percent change in sinkage for BP-1, however, was significantly higher than the percent change in diffusivity.

For both the measurements of the pressure-sinkage relationship and the estimates of thermal diffusivity, there was significantly more variation between trials in the loose preparations than the compact preparations. This is likely in part due to the more repeatable preparation of the compact materials in comparison to the loose materials. However, it may also be because of more inherent variability in the behavior of loose granular media [27].

7 Conclusions and Future Work

This research developed an approach to predict the traversability of terrain through non-contact, photothermal radiometry. The method enables differentiation between safe, compact and hazardous, loose preparations of the same soil, which vision-based methods alone cannot reliably achieve. It transits a low-power, continuous-wave laser and thermal camera across a terrain to effect a thermal transient on terrain, measure that transient, and fit the results to an analytical model to solve for an estimate of thermal diffusivity. For each of the three simulants tested (JSC-1A, BP-1, and GRC-1), a higher measured thermal diffusivity correlated to a higher density and a more traversable granular material as validated by measuring the pressure-sinkage relationship with a bevameter. Preliminary results measured a 56%, 29%, and 35% difference in diffusivity from a compact preparation to a loose preparation for JSC-1A, BP-1, and GRC-1, respectively. These correspond to sinkage increases from compact to loose material of 71% for JSC-1A , 235% for BP-1, and 26% for GRC-1. Thus, the diffusivity estimate produced by this method is a predictor of traversability that probes beyond the visual appearance of terrain.

Future work will analyze the efficacy of this method on mobile robots. Relative to lab instruments and conditions, robots present new challenges including the fact that the robot's velocity must be estimated in order to be able to use this method to predict traversability. A robot's motion will also not likely be precisely linear as it was in the controlled setup used in this paper. Further work is required to adapt this technique to account for variable velocities and nonlinear trajectories. Since both the laser spot size and the temperature change induced by the laser are suspected to be influenced by terrain properties including reflectance and scattering, more research must be conducted into aiding in the estimating of those two quantities. Possibilities include using a camera to visually estimate the spot size and the magnitude of the reflected light, which is related to the amount of power absorbed by the surface. Finally, the effective depth and accuracy of this technique must be investigated in order to determine how what amount of material is actually sensed and the accuracy of the thermal diffusivity measurement.

Acknowledgements. This work is partially supported by the NASA National Robotics Initiative program under contract NNX12AM05G. The authors would also like to thank Colin Creager and NASA Glenn Research Center for their support and for the use of the bevameter.

References

1. Zacny, K., Bualat, M., Lee, P., Alvarez, L., Fong, T., Deans, M., VanGundy, L., Lees, D.: Using percussive, dynamic, and static soil penetrometers to assess geotechnical properties and the depth to ground ice of the mars and lunar analog terrains on the devon island, canadian arctic. In: Earth and Space 2012@ struction, and Operations in Challenging Environments, pp. 284–294. ASCE (2012)
2. Carrier, D.: The four things you need to know about the geotechnical properties of lunar soil. Lunar Geotechnical Institute (2005)
3. Cho, G.-C., Dodds, J., Santamarina, J.C.: Particle shape effects on packing density, stiffness, and strength: natural and crushed sands. Journal of Geotechnical and Geoenvironmental Engineering 132(5), 591–602 (2006)
4. Li, W., Huang, Y., Cui, Y., Dong, S., Wang, J.: Trafficability analysis of lunar mare terrain by means of the discrete element method for wheeled rover locomotion. Journal of Terramechanics 47(3), 161–172 (2010)
5. Abu-Hamdeh, N.H.: Thermal properties of soils as affected by density and water content. Biosystems Engineering 86(1), 97–102 (2003)
6. Becker, B.R., Misra, A., Fricke, B.A.: Development of correlations for soil thermal conductivity. International Communications in Heat and Mass Transfer 19(1), 59–68 (1992)
7. Smits, K.M., Sakaki, T., Limsuwat, A., Illangasekare, T.H.: Determination of the thermal conductivity of sands under varying moisture, drainage/wetting, and porosity conditions-applications in near-surface soil moisture distribution analysis. In: AGU Hydrology Days (2009)
8. Helmick, D., Angelova, A., Matthies, L.: Terrain adaptive navigation for planetary rovers. Journal of Field Robotics 26(4), 391–410 (2009)
9. Brooks, C.A., Iagnemma, K.: Self-supervised terrain classification for planetary surface exploration rovers. Journal of Field Robotics 29(3), 445–468 (2012)
10. Parker, W.J., Jenkins, R.J., Butler, C.P., Abbott, G.L.: Flash method of determining thermal diffusivity, heat capacity, and thermal conductivity. Journal of Applied Physics 32(9), 1679–1684 (1961)
11. Leung, W.P., Tam, A.C.: Techniques of flash radiometry. Journal of Applied Physics 56(1), 153–161 (1984)
12. Tam, A.C., Sullivan, B.: Remote sensing applications of pulsed photothermal radiometry. Applied Physics Letters 43(4), 333–335 (1983)

13. Kelly, A., Stentz, A., Amidi, O., Bode, M., Bradley, D., Diaz-Calderon, A., Happold, M., Herman, H., Mandelbaum, R., Pilarski, T., et al.: Toward reliable off road autonomous vehicles operating in challenging environments. The International Journal of Robotics Research 25(5-6), 449–483 (2006)
14. Chhaniyara, S., Brunskill, C., Yeomans, B., Matthews, M.C., Saaj, C., Ransom, S., Richter, L.: Terrain trafficability analysis and soil mechanical property identification for planetary rovers: A survey. Journal of Terramechanics 49(2), 115–128 (2012)
15. Cline, H.E., Anthony, T.R.: Heat treating and melting material with a scanning laser or electron beam. Journal of Applied Physics 48(9), 3895–3900 (1977)
16. Chen, I., Lee, S.: Transient temperature profiles in solids heated with scanning laser. Journal of Applied Physics 54(2), 1062–1066 (1983)
17. Carslaw, H.S., Jaeger, J.C.: Conduction of heat in solids. Oxford University Press, Oxford (1959)
18. Haberman, R.: Elementary applied partial differential equations: with Fourier series and boundary value problems. Prentice-Hall, Englewood Cliffs (1987)
19. Sanders, D.J.: Temperature distributions produced by scanning gaussian laser beams. Appl. Opt. 23(1), 30–35 (1984)
20. Moody, J.E., Hendel, R.H.: Temperature profiles induced by a scanning cw laser beam. Journal of Applied Physics 53(6), 4364–4371 (1982)
21. Liu, J.M.: Simple technique for measurements of pulsed gaussian-beam spot sizes. Opt. Lett. 7(5), 196–198 (1982)
22. Zeng, X., He, C., Oravec, H., Wilkinson, A., Agui, J., Asnani, V.: Geotechnical properties of jsc-1a lunar soil simulant. Journal of Aerospace Engineering 23(2), 111–116 (2009)
23. Stoeser, D.B., Rickman, D.L., Wilson, S.: Preliminary geological findings on the bp-1 simulant (2010)
24. Oravec, H.A., Zeng, X., Asnani, V.M.: Design and characterization of grc-1: A soil for lunar terramechanics testing in earth-ambient conditions. Journal of Terramechanics 47(6), 361–377 (2010)
25. Wong, J.Y.: Theory of ground vehicles. John Wiley & Sons (2001)
26. Wong, J.Y.: Terramechanics and off-road vehicle engineering: terrain behaviour, off-road vehicle performance and design. Butterworth-Heinemann (2009)
27. Skonieczny, K.: Lightweight Robotic Excavation. PhD thesis, Carnegie Mellon University (2013)

Modular Dynamic Simulation of Wheeled Mobile Robots

Neal Seegmiller and Alonzo Kelly

Abstract. This paper presents a modular method for 3D dynamic simulation of wheeled mobile robots (WMRs). Our method extends efficient dynamics algorithms based on spatial vector algebra to accommodate any articulated WMR configuration. In contrast to some alternatives, our method also supports complex, nonlinear wheel-ground contact models. Instead of directly adding contact forces, we solve for them in a novel differential algebraic equation (DAE) formulation. To make this possible we resolve issues of nonlinearity and overconstraint. We demonstrate our method's flexibility and speed through simulations of two state-of-the-art WMR platforms and wheel-ground contact models. Simulation accuracy is verified in a physical experiment.

1 Introduction

This paper presents a modular method for 3D dynamic simulation of wheeled mobile robots (WMRs). Here, "dynamic simulation" means a second-order physics-based motion model is used, which accounts for inertia and applied forces. This is an expansion on our previously published work on first-order velocity kinematic motion models [15]. "Modular" means that the method accommodates any articulated WMR configuration and any wheel-ground contact model expressed as a function with the specified inputs.

Our colleagues have recently made progress in model-predictive planning [10][16]. To perform well, these planners require accurate motion models that correctly account for wheel slip, rollover, actuator limits, etc. that can also be simulated much faster than real time. Our method strikes a favorable balance between these opposing criteria. We extend efficient dynamics algorithms originally developed for manipulators to account for a non-fixed base and the enforcement of wheel-ground contact

Neal Seegmiller · Alonzo Kelly
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
e-mail: {nseegmiller,alonzo}@cmu.edu

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_6, © Springer International Publishing Switzerland 2015

models. These models specify the nonlinear relationship between force and slip at the wheel-ground interface. The simplest way to incorporate these models is to compute contact forces once per time step based on current slip values, and directly add them at each wheel [1][19][11]. Then the dynamics equations are simply a system of ordinary differential equations; however, the system will be "stiff" requiring a solver that takes very small or adaptive time steps [1]. In contrast, we enforce wheel-ground contact models in a novel differential algebraic equation (DAE) formulation using Lagrange multipliers (Section 3.4). This requires the resolution of nonlinearity and overconstraint issues, but proves to be more stable and efficient.

While our contribution is theoretical, it has immediate practical applications. Using our method a wealth of literature on experimentally derived wheel-ground contact models can be readily applied to full vehicle simulations for planning, control, and estimation purposes. We demonstrate this through simulations of two state-of-the-art WMR platforms and wheel-ground contact models in Section 4.

2 Related Work

Modeling wheel-ground contact is still an active research area. For example, in robotics literature there are terramechanics-based models for rigid wheels in loose soil [11][5][14]. In automotive literature there are models for pneumatic tires [3]. While the ultimate objective of these wheel-level contact models is to improve vehicle-level simulations, few of these publications attempt to do so, perhaps because the available resources for simulation are unsuitable.

Some commercial resources exist for vehicle simulation such as Adams/Car and CarSim. These use very detailed models to the level of small parts and subsystems (e.g. engine mounts); as a result these require the knowledge of many parameters and can be slow. More importantly model configurations are limited to on-road vehicles like cars and trucks. JPL developed Rover Analysis, Modeling, and Simulation (ROAMS) software to model the full dynamics of the Mars Exploration Rovers, including wheel/soil slippage/sinkage interaction [13]. While ROAMS was helpful for design evaluation, it was not feasible for use in motion planning.

Physics-based models that account for wheel slip, rollover, actuator limits, etc. could greatly benefit WMR motion planning, but because implementation and computation costs are high they are seldom used. Ishigami et al. proposed a rover planning algorithm that performs complete dynamic simulation candidate paths, but cited computational cost issues; evaluating just 4 paths (that would each take the rover about 3 minutes to execute) took 47 minutes [12]. Eathakota et al. approximate WMR dynamics in their RRT-like planner to ensure paths satisfy quasi-static and friction cone constraints [7]. Muir and Tian et al. modeled WMR physics in 2D for feedback control purposes [20][23].

Due to its ease of use, Open Dynamics Engine (ODE) is often used to simulate WMRs [11][17][6][15], but it has limited options for wheel-ground contact modeling. ODE can only enforce a rudimentary contact model with Coulomb force limits approximated by a friction pyramid and slip velocities linearly proportional to force.

Our method supports a simulator with ODE's ease of use, that also enforces nonlinear wheel-ground contact models in an unprecedented DAE formulation. Furthermore, unlike ODE, our method is designed for efficiency specifically for WMRs. This is evident in our use of joint space dynamics algorithms, a limited collision engine, and inertia and bias force approximations as explained in Section 3.

Outside of the WMR application, there is relevant work on dynamic simulation. Some have published on the simulation of dynamic systems with *nonholonomic* constraints [18][25], of which wheel slip constraints are a type. Some have created or proposed modular simulators for space applications, such as SpaceDyn [24] and LRMAS [4]. Our method extends the spatial vector algorithms Featherstone originally developed for manipulator dynamics [8]. Orin applied these algorithms to a multilegged vehicle [19], but to our knowledge none have applied them to WMRs.

3 Simulation Mathematics

This section explains the mathematics of our simulation method, throughout which we use the following notational conventions:

- underline denotes a column vector of any dimension $\underline{u}$
- overset harpoon denotes a 3D Cartesian coordinate vector $\overset{\rightharpoonup}{u}$
- overset arrow denotes a 6D Plücker coordinate spatial vector $\overrightarrow{u}$
- ${}^{c}u_{a}^{b}$ indicates that the quantity u is of frame a with respect to frame b, expressed in the coordinates associated with frame c. u_{a}^{b} implies ${}^{b}u_{a}^{b}$.
- R_{a}^{b} denotes the rotation of frame a relative to frame b. Matrix multiplication is used to transform the coordinates that vectors are expressed in as follows: ${}^{b}\overset{\rightharpoonup}{u} = R_{a}^{b}({}^{a}\overset{\rightharpoonup}{u})$. Homogeneous and Plücker transforms (T,X) encode rotation and translation and use the same script notation.
- $[\overset{\rightharpoonup}{u}]_{\times}$ denotes a 3×3 skew symmetric matrix formed from the elements of $\overset{\rightharpoonup}{u}$. This is used to represent cross products by matrix multiplication: $\overset{\rightharpoonup}{a} \times \overset{\rightharpoonup}{b} = [\overset{\rightharpoonup}{a}]_{\times}\overset{\rightharpoonup}{b}$
- $\underline{u}(i)$ denotes the i^{th} element of the vector $\underline{u}$. $\underline{u}(1:n)$ denotes a subvector comprised of elements 1 through n. $A(i,j)$ denotes the element of matrix A at row i, column j. $A(i,*)$ denotes the i^{th} row, and $A(*,j)$ denotes the j^{th} column of A.

3.1 Kinematic Model and State Space

First, a kinematic model of the WMR is constructed as a tree of frames. The root is the body frame which has 6 degrees of freedom (DOF) with respect to the navigation/world frame. Additional frames for steering, suspension, etc. are attached via 1-DOF revolute or prismatic joints. All branches terminate with wheel frames, which by convention are attached via revolute joints about their y-axes. Mass properties are also specified for each frame.

At each time step, a massless frame is also attached to each wheel in contact with the terrain. The origin of the contact frame is the point on the circumference of the wheel that most penetrates the terrain surface. This is computed by a limited collision

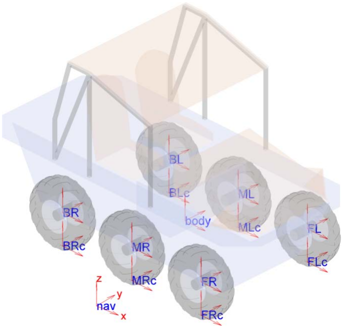


Fig. 1 LandTamer frames diagram

Table 1 LandTamer frames table. Made by PFM Manufacturing Inc.

i	Frame	Parent	Type	Act.	x	y	z	θ_x	θ_y	θ_z
1	body	nav								
2	FL	body	RY	Y	l	w	0	0	0	0
3	FR	body	RY	Y	l	-w	0	0	0	0
4	ML	body	RY	Y	0	w	0	0	0	0
5	MR	body	RY	Y	0	-w	0	0	0	0
6	BL	body	RY	Y	-l	w	0	0	0	0
7	BR	body	RY	Y	-l	-w	0	0	0	0

in inches: l=42, w=32.25, wheel radius=16.5, total mass = 3225 lbm

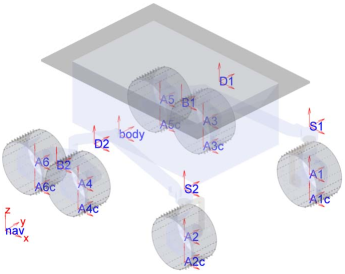


Fig. 2 Rocky 7 frames diagram

Table 2 Rocky 7 frames table [22]

i	Frame	Parent	Type	Act.	x	y	z	θ_x	θ_y	θ_z
1	body	nav								
2	D1	body	RY	N	k2	k3	k1	0	0	0
3	S1	D1	RZ	Y	k4	0	0	0	0	0
4	A1	S1	RY	Y	0	0	-k5	0	0	0
5	B1	D1	RY	N	-k6x	0	-k6z	0	0	0
6	A3	B1	RY	Y	k7	0	-k8	0	0	0
7	A5	B1	RY	Y	-k7	0	-k8	0	0	0
8	D2	body	RY	N	k2	-k3	k1	0	0	0
9	S2	D2	RZ	Y	k4	0	0	0	0	0
10	A2	S2	RY	Y	0	0	-k5	0	0	0
11	B2	D2	RY	N	-k6x	0	-k6z	0	0	0
12	A4	B1	RY	Y	k7	0	-k8	0	0	0
13	A6	B1	RY	Y	-k7	0	-k8	0	0	0

in centimeters: k1=10.5, k2=12.075, k3=20, k4=28.8, k5=12.5, k6x=16·sin(49°), k6z=16·cos(49°), k7=6.9, k8=2, wheel radius=6.5, total mass = 11 kg

engine that intersects wheel and surface geometries. Wheels can be represented as 3D circles (discretized into points) and surfaces can be bounded planes, elevation grids, triangular meshes, etc. The contact frame z-axis is aligned with the surface normal vector at the contact point. The x and y axes are aligned with the longitudinal and lateral slip directions.

Frame information is stored in an ordered list such that the index of any frame is greater than the index of its parent ($i > p(i)$, $i = 1$ is the body frame). Frame data for two example WMRs is provided in Tables 1 and 2. Joint types are revolute (R) or prismatic (P) about one of the axes (X,Y,Z). Act. means Actuated. The last six columns specify the pose of each frame with respect to its parent frame when joint displacement is zero.

We chose to use generalized (or reduced) coordinates for our method. The first elements of the state vector ($\underline{q}$) are the pose of the body frame with respect to the world frame ($\underline{\rho}$). Orientation ($\underline{o}$) may be expressed using either Euler angles or quaternions. The subsequent elements of the state vector are joint displacements ($\underline{\theta}$) in the same order as the frames list.

$$\underline{q} = \begin{bmatrix} \underline{\rho} \\ \underline{\theta} \end{bmatrix} \quad \underline{\rho} = \begin{bmatrix} \vec{t} \\ \underline{o} \end{bmatrix} \tag{1}$$

Open Dynamics Engine and Baraff [2] use a different state space which contains the 6-DOF pose of each frame. This necessitates numerous constraint equations for lower pairs; each 1-DOF revolute or prismatic joint requires a 5-DOF constraint. Big matrices must be inverted at every time step to solve for Lagrange multipliers, but computational cost can be mitigated somewhat by exploiting sparsity.

Given the frames list data and state vector one can compute the forward kinematics, or the homogeneous transform between each frame and its parent frame ($T_i^{p(i)}$) or the world frame (T_i^w). Homogeneous transforms are 4×4 matrices that encode rotation and translation:

$$T_a^b = \begin{bmatrix} R_a^b & \vec{t}_a^{\,b} \\ \underline{0}^T & 1 \end{bmatrix} \tag{2}$$

3.2 Spatial Vector Algebra Dynamics Algorithms

We express WMR dynamics using spatial vector algebra as published by Featherstone [8][9]. This is compatible with our prior work on a vector algebra formulation of WMR kinematics [15]. Spatial vectors are 6D and inhabit two vector spaces: motion and force. Spatial velocity and acceleration ($\vec{v}$,$\vec{a}$) are motion vectors; they contain 3D angular and linear components (on top and bottom respectively). Spatial force ($\vec{f}$) likewise contains 3D moment and linear force components.

$$\vec{v} = \begin{bmatrix} \vec{\omega} \\ \vec{v} \end{bmatrix} \quad \vec{a} = \begin{bmatrix} \vec{\alpha} \\ \vec{a} \end{bmatrix} \quad \vec{f} = \begin{bmatrix} \vec{\tau} \\ \vec{f} \end{bmatrix} \tag{3}$$

The unconstrained dynamic equation is:

$$M\underline{\ddot{q}}_s + \underline{c}(\underline{q},\underline{\dot{q}}_s) = \underline{\tau} \tag{4}$$

$\underline{\dot{q}}_s$ is equivalent to the first time derivative of state ($\underline{\dot{q}}$), except that the time derivative of pose ($\underline{\dot{\rho}}$) is replaced with the spatial velocity of the body frame (${}^b\vec{v}_b^w$). Likewise $\underline{\ddot{q}}_s$ contains the spatial acceleration. $\underline{\tau}$ is a vector of actuator torques/forces applied at the joints.

We use the Recursive Newton-Euler Algorithm (RNEA) to compute the joint space bias force $\underline{c}$, which includes the Coriolis and centripetal force terms, as shown in Algorithm 1. External wheel contact forces may be added directly on line 10 (as do [1][19][11]). Instead, we include these forces via constraints as explained in Section 3.3. Instead of adding gravitational force to each frame, we simply accelerate the base (line 2). n_f is the total number of frames, and n_w is the number of wheel frames. $X_{p(i)}^i$ is a 6×6 Plücker transform that converts *motion* spatial vectors from parent to child coordinates; its transpose converts *force* spatial vectors from child to parent coordinates. I_i is the 6×6 spatial inertia of frame i (which encodes mass, center of mass, moment of inertia). The function $s(i)$ maps the joint type of frame i to a spatial vector index (RX=1, RY=2, RZ=3, PX=4, PY=5, PZ=6).

We use the Composite-Rigid-Body Algorithm (CRBA) to compute the joint space inertia, as shown in Algorithm 2. I_i^c denotes the composite inertia of the subtree rooted at frame i. Note that M is symmetric. Explanations of the original RNEA and CRBA are available in [9]. Our Algorithms 1 and 2 are modified to account for the special structure of WMR kinematics: a non-fixed base, only 1-DOF joints, and n_w contact frames at the list's end which are massless and fixed with respect to their parent wheel frames.

Algorithm 1. RNEA, joint space bias force

```
$\vec{v}_1 = \underline{\dot{q}}_s(1:6)$
$\vec{a}_1 = {}^b\vec{g}$
for $i = 1$ to $n_f$ do
    if $i > 1$ then
        $\vec{h} = 0$
        if $i \le (n_f - n_w)$ then $\vec{h}(s(i)) = \underline{\dot{q}}_s(i+5)$
        $\vec{v}_i = X^i_{p(i)} \vec{v}_{p(i)} + \vec{h}$
        $\vec{a}_i = X^i_{p(i)} \vec{a}_{p(i)} + \vec{v}_i \times \vec{h}$
    end if
    $\vec{f}_i = I_i \vec{a}_i + \vec{v}_i \times I_i \vec{v}_i (+\vec{f}_i^{ext})$
end for
for $i = n_f$ to 2 by -1 do
    if $i \le (n_f - n_w)$ then $\underline{c}(i+5) = \vec{f}_i(s(i))$
    $\vec{f}_{p(i)} = \vec{f}_{p(i)} + (X^i_{p(i)})^T \vec{f}_i$
end for
$\underline{c}(1:6) = \vec{f}_1$
```

Algorithm 2. CRBA, joint space inertia

```
for $i = 1$ to $(n_f - n_w)$ do $I_i^c = I_i$
for $i = (n_f - n_w)$ to 2 by -1 do
    $I^c_{p(i)} = I^c_{p(i)} + (X^i_{p(i)})^T I_i^c (X^i_{p(i)})$
end for
$M = 0$
$M(1:6, 1:6) = I_1^c$
for $i = 2$ to $(n_f - n_w)$ do
    $\vec{f}^c = I_i^c(*, s(i))$
    $M(i+5, i+5) = \vec{f}^c(s(i))$
    $j = i$
    while $j > 1$ do
        $\vec{f}^c = (X^i_{p(i)})^T \vec{f}^c$
        $j = p(j)$
        if $j = 1$ then
            $M(1:6, i+5) = \vec{f}^c$
            $M(i+5, 1:6) = M(i+5, 1:6)^T$
        else
            $M(j+5, i+5) = \vec{f}^c(s(j))$
            $M(i+5, j+5) = M(j+5, i+5)$
        end if
    end while
end for
```

3.3 Wheel-Ground Contact Constraints

Each wheel-ground contact frame has three constraints: one holonomic surface contact constraint which restricts motion along its z-axis, and two nonholonomic slip velocity constraints which restrict motion along its x and y axes. As in [25], holonomic constraints are converted to velocity constraints by differentiation. For a single wheel, the constraint equations are of the form:

$$A\underline{\dot{q}}_s = \vec{v}_c \tag{5}$$

Algorithm 3. Wheel constraint matrix

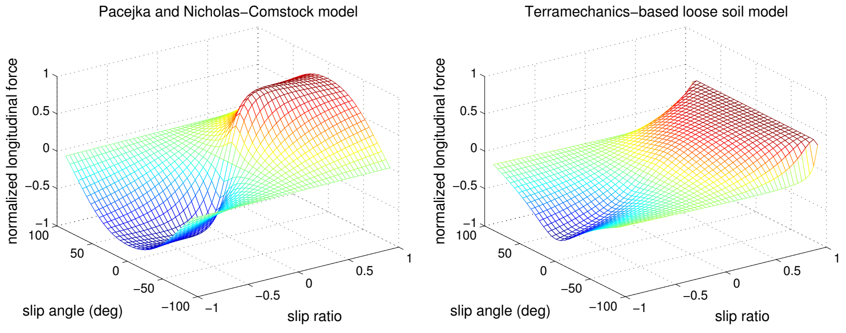
```
1.  $c$ = contact frame index
2.  $i = p(c)$
3.  while $i > 1$ do
4.      if $s(i) \in \{1,2,3\}$ then
5.          $A(*,i+5) = R_i^w(*,s(i)) \times (\vec{t}_c^{\,w} - \vec{t}_i^{\,w})$
6.      else if $s(i) \in \{4,5,6\}$ then
7.          $A(*,i+5) = R_i^w(*,s(i)-3)$
8.      end if
9.      $i = p(i)$
10. end while
11. $A(*,1:3) = [\vec{t}_c^{\,w} - \vec{t}_1^{\,w}]_\times^T R_1^w$
12. $A(*,4:6) = R_1^w$
13. $A = R_w^c A$
```

The matrix A is computed by Algorithm 3, which works backwards along the kinematic chain from contact to body frame. On line 11, the identity $\vec{\omega} \times \vec{t} = -\vec{t} \times \vec{\omega} = [\vec{t}]_\times^T \vec{\omega}$ is used, as $\underline{\dot{q}}_s$ contains the angular velocity of the body frame. The right-hand side $\vec{v}_c$ is short for ${}^c\vec{v}_c^{\,w}$: the velocity of the contact frame with respect to the ground expressed in contact coordinates. This is not constant, but is solved for by optimization as explained in Section 3.4.

We express all wheel-ground contact models as functions in a common format:

$$\vec{f}_c = f(\vec{v}_c, R\omega, \Delta z) \tag{6}$$

The forces exerted by the ground on the wheel ($\vec{f}_c$) are dependent on $\vec{v}_c$, the product of wheel radius and angular rate ($R\omega$), and the displacement between the contact point and terrain surface (Δz) due to sinkage or compression. Plots of longitudinal force vs. slip ratio and angle for two example models are shown in Figure 3.



(a) (b)

Fig. 3 Normalized longitudinal force (f_x/f_z) vs. slip ratio and angle for two wheel-ground contact models. (a) and (b) use equations and parameters in [3] and [11] respectively. These plots are for fixed Δz. Though not shown, these models also determine lateral and normal force.

Constraints for all wheels are stacked into one matrix equation with $3n_w$ rows. Hereafter let A denote the stacked matrix and $\underline{v}_c$ the stacked vector for all wheel constraints. Additional constraints may be appended to account for mechanical restrictions on joint displacements (such as roll/pitch averaging) or to enforce desired speeds for actuated joints.

3.4 Force-Balance Optimization

This section explains perhaps this paper's most important theoretical contribution, the enforcement of *nonlinear* wheel-ground contact models in a DAE formulation. The dynamics equation (4) is modified to include constraints as follows:

$$\begin{bmatrix} M & A^T \\ A & C \end{bmatrix} \begin{bmatrix} \underline{\ddot{q}}_s \\ \underline{\lambda} \end{bmatrix} = \begin{bmatrix} \underline{\tau} - \underline{c} \\ \underline{b} \end{bmatrix} \tag{7}$$

where C is a matrix of zeros except for potentially non-zero "constraint force mixing" values on the diagonal for appended holonomic constraints on joint displacements. These are used just as in Open Dynamics Engine to introduce compliance. (7) can be rearranged to solve for the vector of Lagrange multipliers ($\underline{\lambda}$) which represent constraint forces:

$$\underline{\lambda} = [AM^{-1}A^T + C]^{-1}(\underline{b} - AM^{-1}(\underline{\tau} - \underline{c})) \tag{8}$$

Note that the constraints on state velocity ($\underline{\dot{q}}_s$) have been converted to constraints on state acceleration ($\underline{\ddot{q}}_s$) as follows:

$$A\underline{\dot{q}}_s[i+1] = \underline{v}_c[i+1] \tag{9}$$

$$A(\underline{\dot{q}}_s[i] + \underline{\ddot{q}}_s \Delta t) = \underline{v}_c[i+1] \tag{10}$$

$$A\underline{\ddot{q}}_s = (\underline{v}_c[i+1] - A\underline{\dot{q}}_s[i])/\Delta t \tag{11}$$

$$A\underline{\ddot{q}}_s = (\underline{v}_c[i+1] - \underline{v}_c[i])/\Delta t = \underline{b} \tag{12}$$

$[i]$ and $[i+1]$ denote the current and next time step. $\underline{v}_c[i]$ is already computed in Algorithm 1, whereas $\underline{v}_c[i+1]$ must be computed by optimization:

$$\underset{\underline{v}_c[i+1]}{\arg\min} \, ||\underline{\lambda}(i|i \in W) - \underline{f}_c|| \tag{13}$$

In short, contact point velocities are chosen such that constraint forces computed by the dynamics equation (8) match those computed by the wheel-ground contact model (6). $\underline{f}_c$ denotes the stacked vector of contact model forces for all wheels. W denotes the set of wheel constraint indices (excludes appended constraints).

This optimization can be performed efficiently using Newton's method. Let $\underline{x}$ denote the argument $\underline{v}_c[i+1]$ and $f(\underline{x})$ the objective function; then our guess for $\underline{x}$ is updated as follows:

$$\underline{x}_{n+1} = \underline{x}_n - \gamma[Hf(\underline{x}_n)]^{-1}\nabla f(\underline{x}_n), \quad 0 \leq \gamma \leq 1 \tag{14}$$

Computing the Hessian ($Hf(\underline{x}_n)$) and gradient ($\nabla f(\underline{x}_n)$) requires the Jacobian of the wheel-ground contact model output $\underline{f}_c$ with respect to its inputs. The step size γ is chosen such that the strong Wolfe conditions are satisfied, using a line search algorithm like Algorithm 3.2 in [21].

WMRs have six DOF for motion of the body frame, plus one DOF for each revolute/prismatic joint. They have three constraints for each wheel in contact, plus any appended constraints. Many WMR configurations are overconstrained which results in a rank-deficient A matrix. To address this we choose and enforce a linearly independent subset of the constraints. A well-conditioned subset can be chosen by QR decomposition of A^T. The subset contains all appended constraints, but only some of the wheel constraints. $\underline{f}_c$ in the objective function (13) is replaced with $P\underline{f}_c$ where P projects the full vector of contact forces onto the subset space.

Once $\underline{\ddot{q}}_s$ is solved for, state velocity and state can be updated using symplectic Euler integration as follows:

$$\underline{\dot{q}}_s[i+1] = \underline{\dot{q}}_s[i] + \underline{\ddot{q}}_s \Delta t \tag{15}$$

$$\underline{\dot{q}}[i+1] \leftarrow \underline{\dot{q}}_s[i+1] \tag{16}$$

$$\underline{q}[i+1] = \underline{q}[i] + \underline{\dot{q}}[i+1]\Delta t \tag{17}$$

Note that (16) requires the conversion of angular velocity to either Euler angle or quaternion rates. Higher-order integration methods such as Runge-Kutta are also possible. Unlike the “stiff” ordinary differential equation method, our DAE method is stable even for large time steps.

3.5 Recommendations for Computational Speed-up

No matter how fast your processor, a faster simulator can improve planning performance by enabling more candidate trajectories to be evaluated. One can improve computation time in several ways without compromising simulation accuracy. First, in the optimization initialize contact point velocities ($\underline{v}_c[i+1]$) to values at the previous time step ($\underline{v}_c[i]$). WMRs frequently execute steady-state maneuvers during which these change little. Specify a cost threshold below which optimization via Newton’s method is not required. Next, precompute lookup tables for the wheel-ground contact model and its Jacobian. This is particularly beneficial for the terramechanics-based models in [11][14], which require the costly integration of stress distributions along the wheel surface.

One can further speed up computation for reasonable compromises in accuracy. First, changes in the joint space inertia matrix M may have negligible impact on WMR motion within the predictive horizon. If so, only compute M and M^{-1} for the first time step and reuse these values on subsequent steps. Additionally, the effect of Coriolis and centripetal forces on internal articulations may be negligible. If so, the joint space bias force $\underline{c}$ can be approximated by a vector of zeros except for:

$$\underline{c}(1:6) = I_1^c({}^b\vec{g}) + \vec{v}_1 \times I_1^c \vec{v}_1 \tag{18}$$

This considers the WMR to be a single rigid body (with all joints locked). I_1^c is the composite inertia of the WMR rooted at the body frame, which like M can be computed only once.

4 Results

In this section we evaluate our simulation method in three tests. For the first test, we simulate the LandTamer vehicle (Figure 1) with a Pacejka wheel-ground contact model (Figure 3(a)) driving over a 20° ramp then turning to the left and right at 2 m/s (Figure 4). Figure 5 shows slip ratios and angles for two simulations of this trajectory: one using our method of handling contact forces via constraints, the other adding them directly. Using constraints eliminates jitter and reduces computation time from 21.40s to 2.03s (implemented in MATLAB, using a 2.83 GHz processor). Both simulations benefit from our method's reduced state space and use of the RNEA/CRBA; this speed-up will be quantified in future work. Both simulations use constraints to control wheel velocities, as PID torque controllers can make the dynamics very stiff.

An adaptive integrator (MATLAB's ode45) is required for the direct addition method. Figure 6(a) shows that, to prevent jitter, the integrator takes very small steps relative to the .04s steps taken by our method. If .04s steps were taken by the direct addition method, jitter would become severe instability. Figure 6(b) shows the number of Newton's method iterations required for force-balance optimization in our method (Section 3.4); zero iterations are required during steady-state periods.

We also validated our dynamic model of the LandTamer in a physical experiment (the second test). We drove the LandTamer in various arcs (at up to 2.5 m/s and 0.5 rad/s) in a parking lot at the Taylor test site near Pittsburgh, PA. We tuned parameters of the dynamic model (including tire model parameters and the center of mass) to fit one portion of the dataset, then evaluated on the remainder (Figure 7). Table 3 compares the mean position and absolute yaw error for our tuned dynamic model and a kinematic model that minimizes slip velocity at the six wheels, i.e. $||\underline{v}_c||$. For reasonable planning horizons of 5-20m our predicted position error is 1-2% of distance traveled. Both position and yaw prediction errors using our dynamic model are 88-93% less than using the kinematic model.

For the third test, we simulate the Rocky 7 rover (Figure 2) with a terramechanics-based wheel-ground contact model (Figure 3(b)). The rover traverses uneven, random terrain while making wide turns for 10 seconds at 0.5 m/s (Figure 8). Fig-

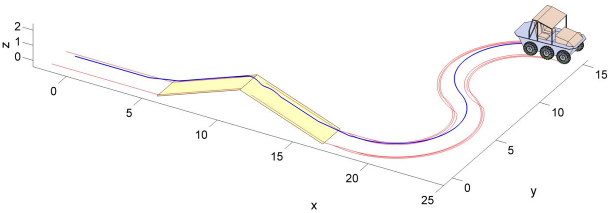


Fig. 4 LandTamer animation screenshot. The path of the body frame is traced in blue; the paths of wheel-ground contact points are traced in red.

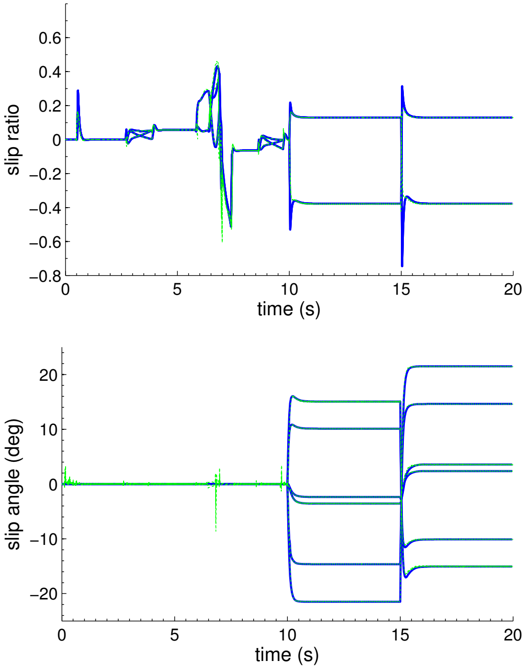


Fig. 5 Slip ratios and angles for all wheels for two simulations of the LandTamer executing the trajectory shown in Figure 4. Our method of handling contact forces via constraints (solid lines) reduces jitter compared to the more common method of adding contact forces directly (dashed lines). As expected, the slip ratios are positive while ascending the ramp (4-6s) and negative while descending (7-9s) due to gravity. The slip angles indicate lateral slip away from the center of curvature when turning (10-20s).

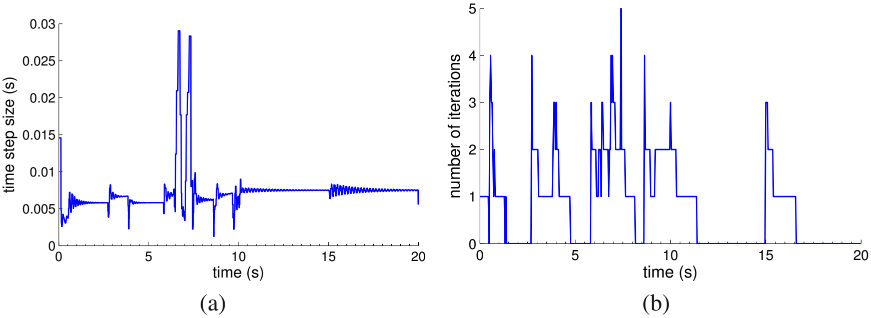


Fig. 6 (a) Integration time step size for the direct addition method (compare to .04s for our method). (b) Number of Newton's method iterations required for our method.

ure 9(a) shows how computation time decreases exponentially with larger time step size. Computation times are normalized by dividing by simulation time; values less than one indicate faster than real time. Each of the approximations suggested in Section 3.5 reduces computation time by approximately 10%. In Figure 9(b), error is the difference in predicted terminal pose with respect to the prediction using no approximations and the minimum (.005s) time step size, for 5m of travel. Error increases linearly (not exponentially) with time step size, and only modestly with

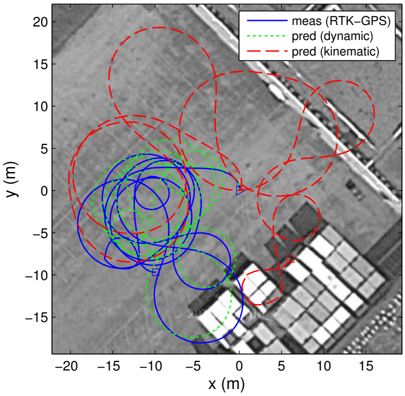


Fig. 7 Evaluation path of the LandTamer at the Taylor parking lot (shipping containers seen in aerial image were not present then). Lines represent: (solid) path measured via Real Time Kinematic GPS, (short-dash) predicted via dynamic model, (long-dash) predicted via kinematic model. Accurate prediction over the entire path length is hard; prediction errors over reasonable horizons are quantified in Table 3.

Table 3 LandTamer model prediction errors

	Dynamic model		Kinematic model	
Horizon	pos.	$\lvert\text{yaw}\rvert$	pos.	$\lvert\text{yaw}\rvert$
5m	0.0669	0.0169	0.5442	0.2006
10m	0.1470	0.0292	1.8076	0.3818
20m	0.3839	0.0515	5.1336	0.6796

Position (m) and absolute yaw (rad) error values are averages over > 50 trials, equally spaced over the 273m evaluation path. 10 and 20m trials overlapped.

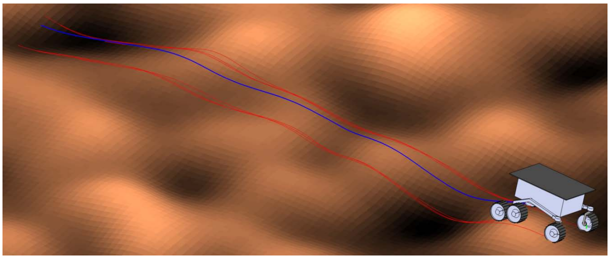


Fig. 8 Rocky 7 rover animation screenshot. The path of the body frame is traced in blue; the paths of the wheel-ground contact points are traced in red.

approximation. Modest errors may be an acceptable tradeoff for significant speed-up in some planning applications.

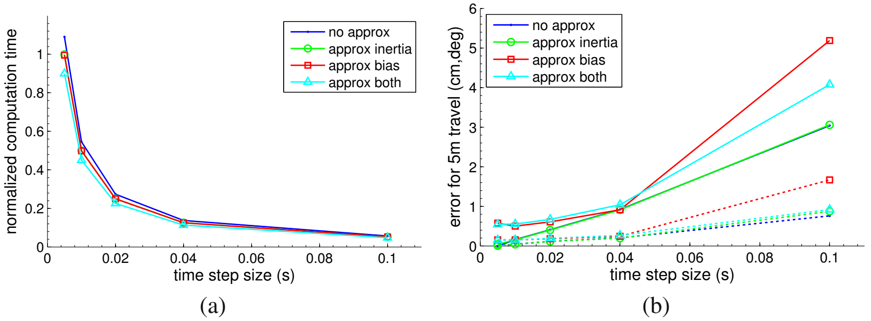


Fig. 9 Computation time and error vs. time step size for our dynamics method, with/without approximation of the joint space inertia and bias force. Averaged over 30 trials. In (b) solid lines are for 2D position error, dashed lines are for absolute yaw error.

5 Conclusions and Future Work

We presented a modular method for the simulation of wheeled mobile robots. In contrast to existing resources such as Open Dynamics Engine, our method uses spatial vector algebra dynamics algorithms which are particularly efficient for WMRs. More importantly, in contrast to ODE, our method can accommodate complex, nonlinear wheel-ground contact models. Our enforcement of these models via constraints in a DAE formulation was demonstrated to be more stable and efficient than the common method of directly adding contact forces in an ordinary differential equation formulation (Section 4, first test).

More physical experiments will be presented in future work on the calibration of 3D WMR motion models. We plan to convert our MATLAB implementation of the WMR simulator to C++; only then can we fairly compare computational speed with alternatives (like ODE, CarSim, etc.). We also plan to make this software publicly available. This will enable existing and future research on wheel-ground contact models to be readily applied to the prediction of full vehicle mobility. The simulator should be fast and accurate enough for improved model-predictive planning in many challenging applications.

Acknowledgements. This research was made with U.S. Government support by the Army Research Laboratory (W911NF-10-2-0016) and by the National Science Foundation Graduate Research Fellowship (0946825). Our colleague Forrest Rogers-Marcovitz collected data at the Taylor test site.

References

1. Balakrishna, R., Ghosal, A.: Modeling of Slip for Wheeled Mobile Robots. IEEE Transactions on Robotics and Automation 11(1) (1995)

2. Baraff, D.: Linear-time Dynamics Using Lagrange Multipliers. In: Proc. SIGGRAPH, pp. 137–146 (1996)
3. Brach, R.M., Brach, R.M.: Tire Models for Vehicle Dynamic Simulation and Accident Reconstruction. SAE Technical Paper (2009)
4. Ding, L., Gao, H., Deng, Z., Song, P., Liu, R.: Design of Comprehensive High-fidelity/High-speed Virtual Simulation System for Lunar Rover. In: Proc. IEEE Conference on Robotics, Automation and Mechatronics, pp. 1118–1123 (2008)
5. Ding, L., Yoshida, K., Nagatani, K., Gao, H., Deng, Z.: Parameter Identification for Planetary Soil Based on a Decoupled Analytical Wheel-Soil Interaction Terramechanics Model. In: Proc. IEEE International Conference on Intelligent Robots and Systems, pp. 4122–4127 (2009)
6. Drumwright, E., Hsu, J., Koenig, N., Shell, D.: Extending Open Dynamics Engine for Robotics Simulation. In: Ando, N., Balakirsky, S., Hemker, T., Reggiani, M., von Stryk, O. (eds.) SIMPAR 2010. LNCS, vol. 6472, pp. 38–50. Springer, Heidelberg (2010)
7. Eathakota, V., Aditya, G., Krishna, M.: Quasi-static motion planning on uneven terrain for a wheeled mobile robot. In: Proc. IEEE International Conference on Intelligent Robots and Systems, pp. 4314–4320 (2011)
8. Featherstone, R.: Robot dynamics algorithms. Kluwer Academic Publishers, Boston (1987)
9. Featherstone, R., Orin, D.: Robot dynamics: equations and algorithms. In: Proc. IEEE International Conference on Robotics and Automation, pp. 826–834 (2000)
10. Howard, T.: Adaptive model-predictive motion planning for navigation in complex environments. Carnegie Mellon University, Robotics Institute Tech. Report, CMU-RI-TR-09-32 (2009)
11. Ishigami, G., Miwa, A., Nagatani, K., Yoshida, K.: Terramechanics-Based Model for Steering Maneuver of Planetary Exploration Rovers on Loose Soil. Journal of Field Robotics 24(3), 233–250 (2007)
12. Ishigami, G., Nagatani, K., Yoshida, K.: Path Planning and Evaluation for Planetary Rovers Based on Dynamic Mobility Index. In: Proc. IEEE International Conference on Intelligent Robots and Systems, pp. 601–606 (2011)
13. Jain, A., Guineau, J., Lim, C., Lincoln, W., Pomerantz, M., Sohl, G., Steele, R.: ROAMS: Planetary Surface Rover Simulation Environment. In: Proc. International Symposium on Artificial Intelligence, Robotics and Automation in Space (2003)
14. Jia, Z., Smith, W., Peng, H.: Fast Computation of Wheel-Soil Interactions for Safe and Efficient Operation of Mobile Robots. In: Proc. IEEE International Conference on Intelligent Robots and Systems, pp. 3004–3010 (2011)
15. Kelly, A., Seegmiller, N.: A Vector Algebra Formulation of Mobile Robot Velocity Kinematics. In: Yoshida, K., Tadokoro, S. (eds.) Field and Service Robotics. STAR, vol. 92, pp. 613–627. Springer, Heidelberg (2014)
16. Knepper, R.: On the fundamental relationships among path planning alternatives. Carnegie Mellon University, Robotics Institute Tech. Report, CMU-RI-TR-11-19 (2009)
17. Lamon, P., Siegwart, R.: 3D Position Tracking in Challenging Terrain. International Journal of Robotics Research 26(2), 167–186 (2007)
18. Luca, A.D., Oriolo, G.: Chapter 7: Modeling and Control of Nonholonomic Mechanical Systems. In: Kinematics and Dynamics of Multi-Body Systems, pp. 277–342. Springer (1995)
19. McMillan, S., Orin, D.E.: Forward Dynamics of Multilegged Vehicles Using the Composite Rigid Body Method. In: Proc. IEEE International Conference on Robotics and Automation, pp. 464–470 (May 1998)

20. Muir, P.: Modeling and control of wheeled mobile robots. Carnegie Mellon University, Robotics Institute Tech. Report, CMU-RI-TR-88-20 (2009)
21. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, New York (1999)
22. Tarokh, M., McDermott, G.: Kinematics modeling and analyses of articulated rovers. IEEE Transactions on Robotics 21(4), 539–553 (2005)
23. Tian, Y., Sidek, N., Sarkar, N.: Modeling and Control of a Nonholonomic Wheeled Mobile Robot with Wheel Slip Dynamics. In: Proc. IEEE Symposium on Computational Intelligence in Control and Automation (2009)
24. Yoshida, K.: The SpaceDyn: a MATLAB toolbox for space and mobile robots. In: Proc. IEEE International Conference on Intelligent Robots and Systems, pp. 1633–1638 (1999)
25. Yun, X., Sarkar, N.: Unified Formulation of Robotic Systems with Holonomic and Nonholonomic Constraints. IEEE Transactions on Robotics 14(4), 640–650 (1998)

Part III
Unmanned Aerial Vehicles

Autonomous River Exploration

Sezal Jain, Stephen Nuske, Andrew Chambers, Luke Yoder, Hugh Cover, Lyle Chamberlain, Sebastian Scherer, and Sanjiv Singh

Abstract. Mapping a rivers course and width provides valuable information to help understand the ecology, topology and health of a particular environment. Such maps can also be useful to determine whether specific surface vessels can traverse the rivers. While rivers can be mapped from satellite imagery, the presence of vegetation, sometimes so thick that the canopy completely occludes the river, complicates the process of mapping. Here we propose the use of a micro air vehicle flying under the canopy to create accurate maps of the environment. We study and present a system that can autonomously explore rivers without any prior information, and demonstrate an algorithm that can guide the vehicle based upon local sensors mounted on board the flying vehicle that can perceive the river, bank and obstacles. Our field experiments demonstrate what we believe is the first autonomous exploration of rivers by an autonomous vehicle. We show the 3D maps produced by our system over runs of 100-450 meters in length and compare guidance decisions made by our system to those made by a human piloting a boat carrying our system over multiple kilometers.

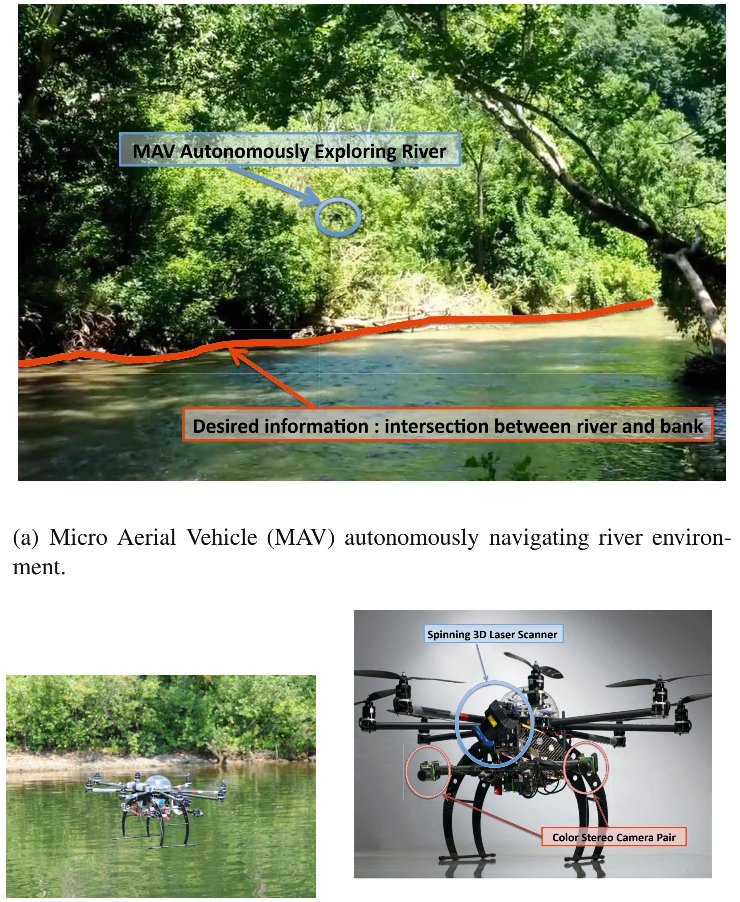
1 Introduction

Riverine systems are an increasingly important focus for many applications like mapping, monitoring and surveillance where it is desirable to use autonomous exploration to traverse the river and collect up to date information. A small lightweight system that can travel below the tree line to sense the river width, the river direction and canopy clearance is advantageous since this information is often not possible to measure from satellite imagery because tree canopy cover occludes the river from

Sezal Jain · Stephen Nuske · Andrew Chambers · Luke Yoder · Hugh Cover ·
Lyle Chamberlain · Sebastian Scherer · Sanjiv Singh
Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave,
Pittsburgh, PA, 15213, USA
e-mail: sezal@andrew.cmu.edu, nuske@cmu.edu

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_7, © Springer International Publishing Switzerland 2015

above. Further, narrow, densely-forested, rivers are difficult to navigate by surface craft because of submerged and semi-submerged obstacles and therefore we develop a Micro Aerial Vehicle (MAV) that is small and nimble and able to traverse the difficult terrain.


MAV Autonomously Exploring River
Desired information : intersection between river and bank
Spinning 3D Laser Scanner
Color Stereo Camera Pair

(a) Micro Aerial Vehicle (MAV) autonomously navigating river environment.

(b) Close-up of vehicle exploring river environment.

(c) Vehicle local sensing consists of a lightweight spinning laser scanner and stereo color camera pair.

Fig. 1 A Micro Aerial Vehicle is used to autonomously explore and map the river environment. The information of interest is the intersection between bank and river. The vehicle is lightweight and agile and not suseptible to submerged and semi-submerged obstacles such as would be hazardous to a surface vehicle. To avoid obstacles and to perceive the extent and course of the river the vehicle is fitted with a spinning 3D laser scanner and color stereo pair.

Existing applications for autonomous river operations focus on collecting information from robotic boats navigating based on stored directions. These existing works have used pre-determined GPS waypoints as a navigation guide. Often riverine systems are densely overgrown with vegetation and autonomous exploration cannot depend on the irregular and erroneous GPS measurements in these surroundings. Due to their dense canopy, estimating an initial map of the waterways from satellite images is also not a viable option. Further riverine environments are also continuously evolving, and current information of the width and course is often not available. For all of these reasons river environments must be explored without relying on pre-determined maps or waypoints. To date, there has not been a truly autonomous exploration system demonstrated in a river environment. We present what we believe is the first such system that can explore a river solely from local sensing, and in addition our vehicle is the first that can fly through river environments to quickly explore while avoiding submerged and semi-submerged obstacles.

We build on our existing work in river environments for autonomous perception, positioning, and obstacle avoidance work [1, 3, 4, 12, 11] and extend to add the key capability of truly autonomous exploration. Our contribution here is to present a new exploration algorithm based on a multi-variate cost function to maximize the information collected in a river map given the fixed duration of a mission. We use two sensor modailities which are different in range and accuracy, namely vision and laser in local sensing for the proposed algorithm. We demonstrate that our method is more adept than traditional exploration algorithms and can traverse the river to gather more information during a mission.

2 Related Work

Much work has been placed in development of autonomous vehicles for navigating waterways, using a variety of different types of crafts such as automated catamarans [5, 9], small lightweight fanboats [15], kayaks [8] or small inflatable craft [6]. Most of these existing waterway navigation works rely on predefined GPS-waypoints, or a pre-defined map generated from satellite imagery [6]. In contrast our work is focused on autonomous exploration, where the environment is perceived by onboard sensors and the vehicle reacts by planning routes that navigate the vehicle along the waterway and maps the direction and width of the river-bank.

We achieve this with a spinning 3D laser scanner, which has also been demonstrated for local obstacle avoidance [5] and [6] to navigate around obstacles discovered above the water surface. However, we do not use any prior information and rely on intelligent path and goal planning based on the information received by our local sensing. One somewhat related work is by [10] where rivers are detected and tracked from aerial vehicles, although unlike our work these are higher flying vehicles making them as unsuitable as satellite images, whereas our system operates beneath the tree-line, close to the river surface.

In terms of exploration strategies, a common approach is to optimize the robot pose accuracy and the accuracy of the resulting map, [2, 7]. In contrast we rely on separate positioning algorithms [11] for pose accuracy, and instead we focus our exploration algorithm to maximize the length of riverbank discovered. In some exploration strategies, information maximization is focused on reducing uncertainty in pose and likelihood of map cells being an obstacle or free space, [2, 7]. Other approaches more closely related to ours define exploration goals that select viewpoints that are expected to yield the highest increase in entropy [14, 13] resulting in the robot seeking out regions that have not yet been explored. Overall, these strategies are closely related to the standard frontier exploration systems [16]. Our method is similar in nature, although we introduce a multi-variate cost-map, which finds trajectories that maximize the length of a river explored for a given mission time.

3 Approach

The key aim of the work we present here is to plan goals for the vehicle to execute trajectories to realize the following behaviors:

- Follow river, whilst maintaining stable flight and avoiding obstacles
- Maximize the information collected over the course of the river

3.1 Environment Modeling and Sensing

The riverine system is modeled as a planar grid (χ). Each cell χ^i in the grid represents a cell in the world at location x_i, y_i , and the rivermap values of this cell in the grid is as follows:

$$\chi_r^i = \chi_r^{x_i,y_i} = \begin{cases} 1 & \text{if the cell is part of river} \\ -1 & \text{if the cell is part of bank} \\ 0 & \text{if the cell has not been observed} \end{cases} \tag{1}$$

Taking χ we form a function that defines the current information that we have of the river. We define the intersection between river and bank as the pertinent information for mapping the width and course and use these cells as a measure of information. To achieve this we form a new information map as follows to search for discontinuities in the current river model:

$$I^{(x_i,y_i)} = \sum_{u=x_i-1}^{u=x_i+1} \sum_{v=y_i-1}^{v=y_i+1} (\text{sign}(\chi_r^{(x_i,y_i)}) \neq \text{sign}(\chi_r^{(u,v)})) \tag{2}$$

Our exploration algorithm seeks to extract desirable trajectories for the vehicle that will maximize the entropy in I.

The above formulation is derived from data collected from local sensing mounted onboard the vehicle, see Fig.1(c). The laser scanner and the camera onboard generate environment maps which are used for goal planning. We segment images

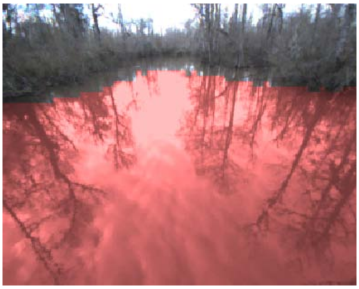


Fig. 2 Pixels with probability greater than .5 after river segmentation are classified as river and marked red [1]. They are then projected onto river plane to create an initial map of the environment. This image is from a dataset taken from McCarthy river, Mississipi.

received through a camera and use the probabilities generated to create a river map by projecting them onto the river surface using the method from [1] as shown in-Fig.2.

We use a lightweight spinning laser range scanner to create a 3D scan of the environment (see Fig.1(c) and [12]). The maximum range of this scanner depends on ambient illumination and reflectance characteristics of objects, but in typical outdoor environments we observe maximum ranges of approximately 15 meters. We use this range to determine which laser missed returns are due to limited range and which are due to water absorbtion (i.e. we can detect the river from these laser misses). The laser range measurements are converted into a 3D point cloud in the world frame using an accurate positioning system that operates in GPS denied environments ([12] and [11]). In addition to global positioning we measure the current height above the river surface, which cannot be derived purely from the global frame, since the height will vary according to current water level. To achieve this we extract specular returns from the water surface in a tight cone directly below the vehicle.

Once the global position and relative height above the surface is known, we can then proceed to use the laser measurements to form our environment map. In particular the following rules are applied:

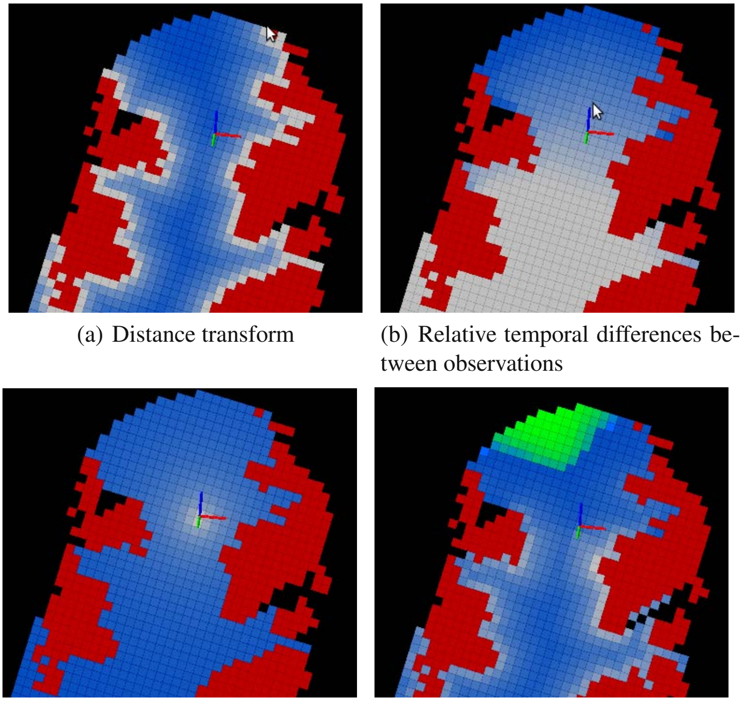
- All *missed* laser returns (those with the maximum laser range), that pass through the river plane, are considered as river cells at the intersection of the ray and the river plane $\{\chi_r^i = 1\}$
- All laser hits less than maximum range are projected on the environment grid and based on the density of these projected hits in a cell, the cell is classified as part of the river bank. $\{\chi_r^i = -1\}$

3.2 Goal Planning for River Following

The main task in autonomous exploration is to take local perception of the environment and to extract goals for the vehicle to traverse towards. The goals are then fed in as input to the low-level motion planning algorithm. The low-level motion planning we have developed in earlier in [4]. The exploration algorithm we present here sets goals that seek to *maximize the information gained during the mission*.

To achieve desired behaviors we introduce multivariate cost maps, that respect the characteristics of the sensing and extend the abilities of more simplistic traditional frontier exploration algorithms [16]. In particular the costs we derive enable the vehicle to observe the maximum amount of the riverbank whilst following the course of the river, and where possible avoid returning to unexplored portions of the river that are behind the vehicle. Unexplored frontier that was not observed as the vehicle passed by initially, may become larger in size than a narrow passage that the vehicle encounters directly ahead, however, it is suboptimal to return to these locations as little new information is collected on the journey back to previously explored areas.

For one we develop a riverbank hugging behavior which uses a distance transform based cost function, $C_d(\cdot)$ that aims to keep the vehicle away-from but near-



(a) Distance transform

(b) Relative temporal differences between observations

(c) Range of the cell to the current vehicle location

(d) Combined multi-variate cost function

Fig. 3 Visualizations of the multi-variate cost functions. Obstacles are highlighted in red, the cells observed as river are rendered a shade from white to blue, where deep blue represents low cost to go. Cells with lowest cost in a map represent the next goal point for navigation. For the combined cost functions we indicate the cluster of lowest cost cells in green.

enough distance to the riverbank to both assist the 3D mapping of the bank and ensure the local motion estimation is functional. This range is designed to result in maximal information gain of the riverbank. To arrive at $C_D(i)$ we compute a distance transform $f_D(i)$ that returns the distance to the nearest obstacle ($\chi_r^j < 0$). We very efficiently compute this distance cost as described in detail in [12]. After calculating the distance transform we apply a function to penalize goals very close to obstacles and also penalize goals far away from obstacles using a desired distance κ_D as follows:

$$C_D(i) = 1 - \exp(k_D(f_D(i) - \kappa_D)^2 + \frac{1}{f_D(i)}) \tag{3}$$

$$f_D(i) = \underset{\substack{j=1:N \\ \chi_r^j < 0}}{\operatorname{argmin}} ||(x_i, y_i) - (x_j, y_j)|| \tag{4}$$

where k_d is a tuning constant. The resulting functional is depicted in Fig. 3(a), where the cost is high near the obstacles and descends to a minima at κ_D.

The next cost we introduce is designed to avoid retracing steps, in particular we assign cells that have been observed more recently with lower cost than those behind the vehicle, that were observed further in the past. We take the elapsed time since the ith cell was last observed as χ_t^i and use it to penalize retracing through cells seen previously as follows:

$$C_T(i) = t - \chi_t^i \tag{5}$$

Fig. 3(b), visualizes this temporal observation cost. An important cost we introduce is $C_R(i)$. The range the cell is from the current vehicle-location, which is designed to maximize distance traversed along river:

$$C_R(i) = ||(x_i, y_i) - (x_t, y_t)|| \tag{6}$$

where (x_t, y_t) is the current position of the vehicle, see Fig. 3(c).

Next we introduce a cost to favor the vehicle continuing on its current course to avoid the issue of isotropic sensor input that typically occurs at commencement of a mission when no obstacles are within range and the aforementioned costs are at an equilibrium and do not return stable goals.

$$C_H(i) = \exp^{\kappa_H(\theta_z^v - \Delta_\theta)} \tag{7}$$

$$\Delta_\theta = arctan((\frac{x_t - x_i}{y_t - y_i})^2) \tag{8}$$

Where k_H and κ_H are constants that are empirically determined to create a dip in cost around zero heading to enable vehicle to maintain its course when the sensory inputs do not provide stable goals, such as in open waters.

Finally an obstacle path cost $C_O(i)$ is derived from the set of cells (P) connecting the vehicle position with cell i:

$$C_O(i) = \underset{p \in P}{\operatorname{argmax}}(f_O(p)) \tag{9}$$

$$f_O(p) = \begin{cases} 0 & \text{if } \chi_r^p == 1 \\ \kappa_O & \text{otherwise} \end{cases} \tag{10}$$

Where κ_O is a suitably large constant to avoid obstacles. Individually these costs do not produce desirable behavior, however when correctly fused together, the vehicle maintains course. Therefore, the final objective of the goal planning algorithm is to combine the costs and extract the resulting goal χ^G that is to be passed to the motion planning algorithm.

Then to extract goals we compile a set Ψ that contains the cells with the lowest $n\%$ cost, then find a weighted mean over this set:

$$G = \underset{\psi \in \Psi}{\operatorname{argmin}}(\sum_{i=1:N}(||(x_\psi, y_\psi) - (x_i, y_i)|| \cdot C(\psi))) \tag{11}$$

4 Results

We validate our method in a set of experiments, beginning with controlled simulations executed within maps of real-world data, continuing with results from autonomous flights over rivers and waterways and then we present open loop comparison with human operator on real-world data.

4.1 Simulated Exploration on Real-world Data

For simulation we use an environment model generated from data collected over a section of McCarthy River in Mississippi, USA. A 3D point cloud registered in world coordinates is generated from data collected through the sensor suite carried on a boat traversing the river. From this point cloud we can simulate the laser measurements given a particular pose of the robot and the known characteristics of our laser scanner. This gives us the means to evaluate our autonomous exploration algorithm based on multi-variate cost maps against a traditional frontier exploration algorithm.

We use planning cycles executed every 10 seconds and use a fixed number of planning cycles to give a fair evaluation. Each algorithm is given the same initial conditions and we measure the information gained at each time step during the simulated mission. In Fig. 4 the information gained with approach is plotted against time, where clearly the more traditional frontier approach has difficulties maintaining advantageous trajectories for mapping the riverbank. When narrow passages appear in the river, the frontier algorithm oscillates between returning to explore the earlier unexplored frontier segments and returning to the narrow passages. Our method both maintains optimal distance from the riverbank to avoid suboptimal trajectories and also selects trajectories with higher probability of maintaining course along the river and avoid backtracking down the river to observe portions of the bank.

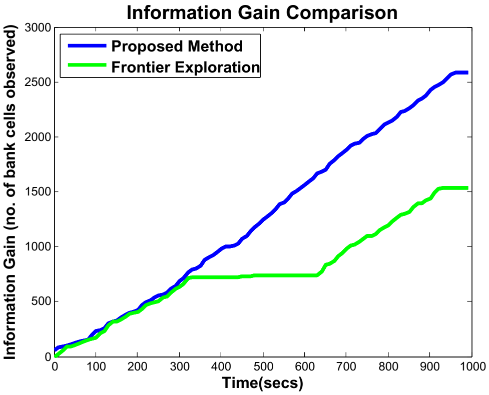


Fig. 4 Simulation of autonomous exploration comparing our method using multi-variate cost maps against a more traditional frontier exploration algorithm [16]. The simulation is given the same initial conditions within an environment model formed from real-data collected from a section of McCarthy River in Mississippi, USA. The information gained using each algorithm is compared. When presented with a narrow passage appears, the frontier algorithm oscillates between returning to earlier unexplored segments of the river, whereas our algorithm continues in the trajectory that is far greater probability of increasing the information collected of the riverbank.

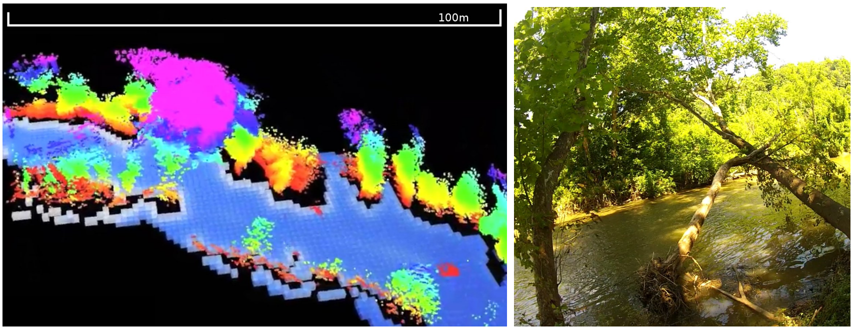
4.2 Autonomous Flights

After demonstrating in simulation that our exploration algorithm is more optimal at returning favorable trajectories, we now proceed to evaluate our approach in real-world truly autonomous flights over rivers and waterways. We manually bring our system into a hover over a river and switch into autonomous mode and from there let the vehicle explore the river autonomously at a velocity of 1 m/s with no further human inputs.

The flights are on tight and densely vegetated sections of a river, and demonstrate the complete system for planning trajectories that avoid overhanging tree obstacles and maintaining course along the river. The algorithm is able to plan trajectories which enable the system to stay in range of both riverbanks where possible resulting in an optimal trajectory for maximizing information.

Fig. 5 shows an example of the cluttered river environment the robot is flying through during an autonomous 100m flight. The canopy and the river/bank classification map from this flight are shown in Fig. 6. This experiment demonstrates the advantage of our algorithm against getting information from satellite images as we are able to classify the areas lying underneath the canopy as river or bank.

Finally we demonstrate the ability of the system to navigate a river over long distances. In the longest autonomous run on a river, the robot flew for about 450m along the length of a narrow river. The limiting factor in the distance covered during this test was the battery life of the vehicle.



(a) (b)

Fig. 5 (a)Laser-point cloud collected from the autonomous flight through the densely-forested river environment. The detected river extent is colored with blue cells and the bank and overhanging trees are colored by height above river surface. (b) Environment where we test the system's ability to fly autonomously through a densely forested river.

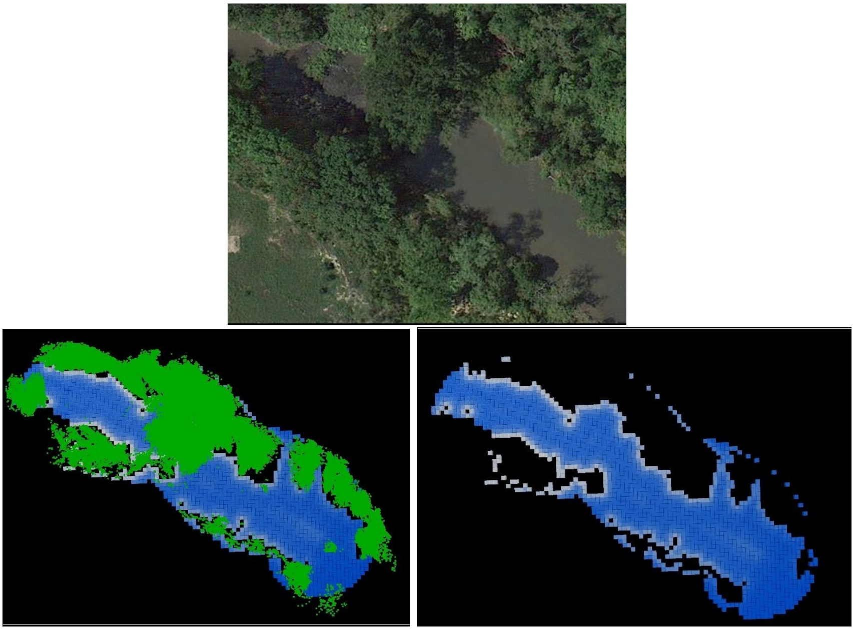


Fig. 6 Map generated from autonomous flight through narrow and densely vegetated river-segment. Top: Satellite view of river segment. Left: Map of river where green is the bank and overhanging trees detected by the laser scanner. Right: Traversable river extent detected by the laser scanner. Notice the overhanging trees in the middle of the segment are removed and a clear and traversable path is discovered underneath by the flying vehicle. The traversable river map, for example, could be used by a surface craft following the flying vehicle enabling it to have knowledge of where it is safe to travel.

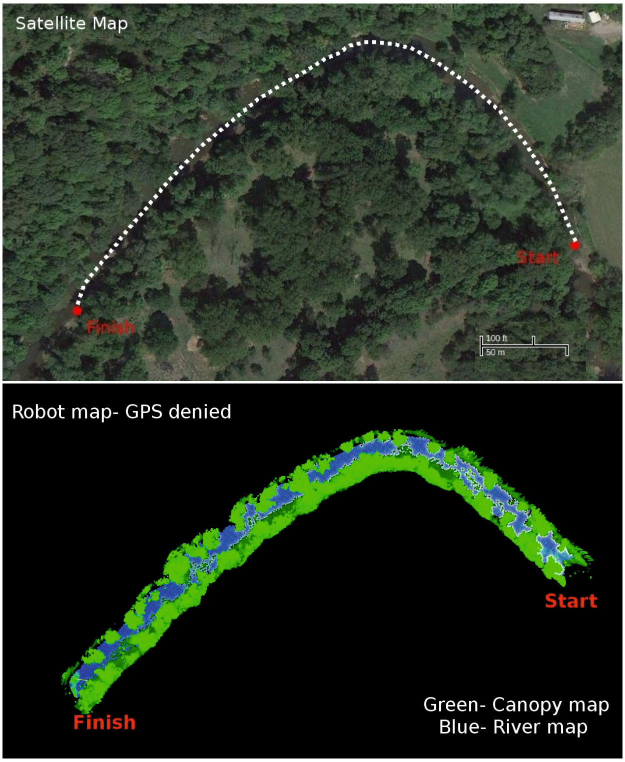
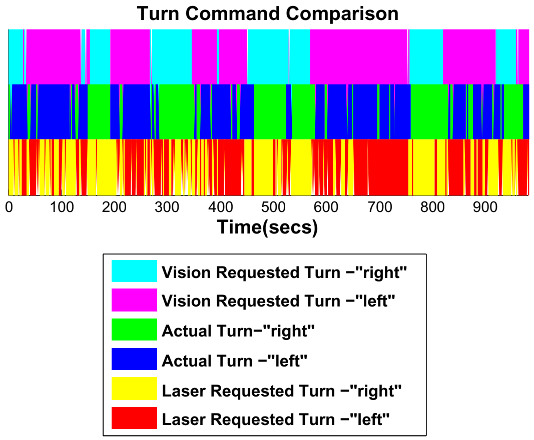


Fig. 7 **Autonomous flight**: Top: Satellite image with overlaid flight trajectory and Bottom: The river/bank map (blue) and canopy map (green) generated by the robot from data collected in a 450m autonomous flight along a river

This experiment was conducted on a very shallow river about 10-15m in width with dense vegetation on both banks. The robot localizes itself without any GPS input and is able to classify the cluttered environment into river and obstacles to explore and plan through it. There were some pauses in robot trajectory in some sections of the river due to trees with branches hanging over the river blocking the path for the robot. Shallow areas in the river made the problem harder as they would be classified as obstacles due to large number of laser returns. Wind was also a challenge as a slight drift from the trajectory would take the robot too close to an obstacle. The final map of the bank and the trajectory of the vehicle during this experiment is displayed in Fig. 7. The system operates without manual intervention successfully exploring the river and turning according to the river direction.

4.3 Open Loop Comparison

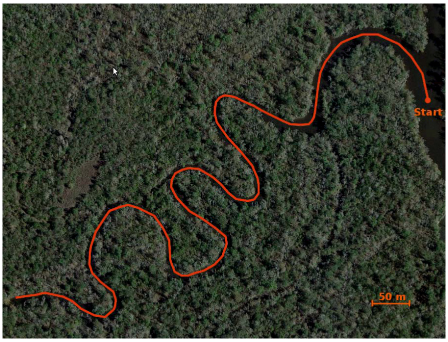
Finally, we compare vision and laser sensors for navigation against a human operator, whom we consider to make expert decisions on how to navigate along the river. Human decisions are either *left* or *right* turn decisions which are measured



(a)

	50-200m	full path
Vision	79.3%	77%
Laser	54%	66%

(b)



(c)

Fig. 8 (a) Comparison of turn commands over time. Long range vision commands are more stable and do not require the vehicle to change directions unnecessarily. Laser commands are more jittery as they consider a much smaller environment while planning and react suddenly to any changes. (b) % accuracy of vision and laser turn requests by comparing them to the ground truth. Results from the 50-200m wide stretch show a larger difference in vision and laser accuracies. (c)The 1.5 km path followed on McCarthy river.

from heading changes in pose estimate. We perform an open loop comparison of the sensors and human operator on data collected by the robot platform fixed on a boat driven along McCarthy river.

We compare the decisions made by the human operator against the turn decisions made based on goal-points received from the multi-variate exploration algorithm using laser as well as vision sensing.

Since the laser scanner has a short range, laser based navigation follows the contours of a bank closely making more reactive decisions. The vehicle is able to look further ahead using vision to make intelligent and time-efficient decisions. This difference is emphasized in a wide river, where vision will lead the vehicle down the river following its general course, but laser navigation will stick to the bank and follow its contours which is time-inefficient and in contradiction to what a human operator would do. After an initial 50m, the next 150m stretch is more than 25m wide and vision performs much better than laser for this stretch, see Fig. 8(b), (c).

5 Conclusion

Our work demonstrates that autonomous autonomous exploration is possible in river environments and that neither GPS waypoints nor prior maps are necessary. In addition we also demonstrate GPS denied flight with planning algorithms that can robustly extract goal points in challenging unstructured terrain. While there is much work in autonomous exploration for ground-vehicles, these algorithms do not directly translate to river environments. Our system is developed to respect the specific physical layout and properties of the river and bank and the behavior of the sensors and perception algorithms in these environments.

In future work, we still see challenges to increase the operating velocity in a safe manner. One avenue to explore is to predict ahead the course of the river with more accuracy giving confidence of which directions are most likely to possess free-space and when turns, dead-ends or forks are likely to appear. We also see persistent monitoring a waterway as an important means to detect pertinent changes to the environment. Further we see that small flying vehicles while fast and nimble, have limited time of flight and must be combined with a supporting vehicle which is larger and trails behind offering the ability for the flying vehicle to return for landing and recharging. These non-homogenous teams of vehicles pose many interesting research challenges, in both high fidelity localization and tracking and with relative motion planning for high-speed take-offs and landings ,in addition to information sharing to exploit the different sensing characteristics and viewing perspectives of the vehicles.

References

1. Achar, S., Sankaran, B., Nuske, S., Scherer, S., Singh, S.: Self-supervised segmentation of river scenes. In: 2011 IEEE International Conference on Robotics and Automation, pp. 6227–6232. IEEE (May 2011), doi:10.1109/ICRA.2011.5980157
2. Amigoni, F., Caglioti, V.: An information-based exploration strategy for environment mapping with mobile robots. Robotics and Autonomous Systems 58(5), 684–699 (2010), doi:10.1016/j.robot.2009.11.005
3. Chambers, A., Achar, S., Nuske, S., Rehder, J., Kitt, B., Chamberlain, L., Haines, J., Scherer, S., Singh, S.: Perception for a river mapping robot. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 227–234. IEEE (2011), doi:10.1109/IROS.2011.6095040
4. Cover, H., Choudhury, S., Scherer, S., Singh, S.: Sparse tangential network (spartan): Motion planning for micro aerial vehicles. In: International Conference on Robotics and Automation (2013)

5. Dunbabin, M., Grinham, A., Udy, J.: An autonomous surface vehicle for water quality monitoring. In: Australasian Conference on Robotics and Automation, ACRA (2009)
6. Gadre, A., Du, S., Stilwell, D.: A topological map based approach to long range operation of an unmanned surface vehicle. In: American Control Conference, pp. 5401–5407 (June 2012)
7. Kollar, T., Roy, N.: Trajectory optimization using reinforcement learning for map exploration. The International Journal of Robotics Research (June 2008)
8. Leedekerken, J.C., Fallon, M.F., Leonard, J.J.: Mapping complex marine environments with autonomous surface craft. In: Khatib, O., Kumar, V., Sukhatme, G. (eds.) Experimental Robotics. STAR, vol. 79, pp. 525–539. Springer, Heidelberg (2014)
9. Pradalier, C., Posch, T., Pernthaler, J., Siegwart, R.: Design and Application of a Surface Vessel for Autonomous Inland Water Monitoring. IEEE Robotics & Automation Magazine, 1–9 (2012)
10. Rathinam, S., Almeida, P., Kim, Z., Jackson, S., Tinka, A., Grossman, W., Sengupta, R.: Autonomous Searching and Tracking of a River using an UAV. In: American Control Conference, pp. 359–364. IEEE (2007), doi:10.1109/ACC.2007.4282475
11. Rehder, J., Gupta, K., Nuske, S., Singh, S.: Global pose estimation with limited gps and long range visual odometry. In: Proceedings of the 2012 IEEE/RSJ International Conference on Robotics and Automation (2012)
12. Scherer, S., Rehder, J., Achar, S., Cover, H., Chambers, A., Nuske, S., Singh, S.: River mapping from a flying robot: state estimation, river detection, and obstacle mapping. Autonomous Robots 33(1-2), 189–214 (2012), doi:10.1007/s10514-012-9293-0
13. Stachniss, C., Burgard, W.: Exploring unknown environments with mobile robots using coverage maps. In: International Joint Conference on Artificial Intelligence, pp. 1127–1132 (2003)
14. Moorehead, S.J., Simmons, R., Whittaker, W.L.: Autonomous exploration using multiple sources of information. In: IEEE International Conference on Robotics and Automation (2001)
15. Valada, A., Velagapudi, P., Kannan, B., Tomaszewski, C., Kantor, G.A., Scerri, P.: Development of a low cost multi-robot autonomous marine surface platform. In: Yoshida, K., Tadokoro, S. (eds.) Field and Service Robotics. STAR, vol. 92, pp. 643–658. Springer, Heidelberg (2014)
16. Yamauchi, B.: A frontier-based approach for autonomous exploration. In: Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA 1997, pp. 146–151. IEEE Comput. Soc. Press (1997), doi:10.1109/CIRA.1997.613851

Outdoor Flight Testing of a Pole Inspection UAV Incorporating High-speed Vision

Inkyu Sa, Stefan Hrabar, and Peter Corke

Abstract. We present a pole inspection system for outdoor environments comprising a high-speed camera on a vertical take-off and landing (VTOL) aerial platform. The pole inspection task requires a vehicle to fly close to a structure while maintaining a fixed stand-off distance from it. Typical GPS errors make GPS-based navigation unsuitable for this task however. When flying outdoors a vehicle is also affected by aerodynamics disturbances such as wind gusts, so the onboard controller must be robust to these disturbances in order to maintain the stand-off distance. Two problems must therefor be addressed: fast and accurate state estimation without GPS, and the design of a robust controller. We resolve these problems by a) performing visual + inertial relative state estimation and b) using a robust line tracker and a nested controller design. Our state estimation exploits high-speed camera images (100 Hz) and 70 Hz IMU data fused in an Extended Kalman Filter (EKF). We demonstrate results from outdoor experiments for pole-relative hovering, and pole circumnavigation where the operator provides only yaw commands. Lastly, we show results for image-based 3D reconstruction and texture mapping of a pole to demonstrate the usefulness for inspection tasks.

1 Introduction

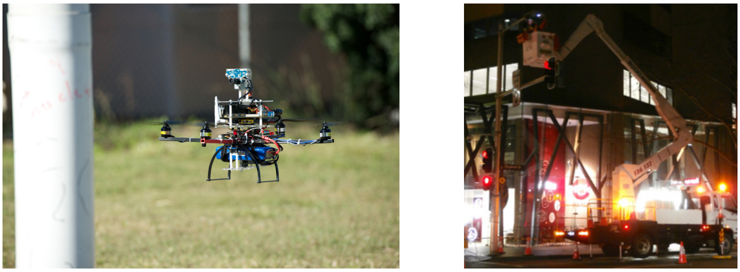
Our work is motivated by the problem of inspecting vertical infrastructure such as street lights or electrical distribution poles. There are more than 175 million street lights in the world which need to be inspected periodically[1]. The options for in-

Inkyu Sa · Peter Corke
Queensland University of Technology, Brisbane, Australia
e-mail: {i.sa,peter.corke}@qut.edu.au

Stefan Hrabar
CSIRO Computational Informatics, Brisbane, Australia
e-mail: Stefan.Hrabar@csiro.au

[1] 2007 Echelon, Monitored Outdoor Lighting, http://www.echelon.com

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_8, © Springer International Publishing Switzerland 2015



(a) (b)

Fig. 1 (a) Our MikroKopter platform hovering relative to a pole in the foreground. (b) Traditional pole inspections can require lengthly setups and road blockages for vehicle access which disrupts traffic.

specting these are limited. Ladders can be used up to a height of 10∼15 m however are quite dangerous: each year in the United States more than 160 people are killed in ladder accidents and 242,000 are injured[2]. A ‘cherry picker’ can be used for taller structures (Fig. 1(b)) however vehicle access is required and the setup time is significant. In recent years we have seen significant advances in small vertical take-off and landing (VTOL) platforms, in particular multirotors, driven by advances in integrated circuit techniques and MEMS sensors. These systems are low-cost and have sufficient payload and endurance for useful inspection missions of individual poles. They are also low-weight which reduces the hazard due to their deployment. Such platforms are however not trivial to fly, especially close to solid, unforgiving structures and beyond line-of-sight. It is particularly hard for an operator to judge the stand-off distance to vertical infrastructure from his viewpoint on the ground.

To address the pole inspection task in outdoor environments we present the system shown in Fig. 1(a) comprising a VTOL platform carrying a 100 Hz front-facing camera and a 70 Hz inertial measurement unit (IMU). This builds on our previous work [1][2] and makes the following new contributions:

- Estimation of the camera latency and inclusion of the latency in the control loop.
- Evaluation of state estimation and control for outdoor flight tests, using a laser tracker for ground truth.
- Demonstration of a pole inspection scenario where the system maintains its pole-relative pose leaving the operator free to control only yaw and height.
- Demonstration of the feasibility of image-based 3D reconstruction and surface texturing of a pole using images captured during flight.

The remainder of the paper is structured as follows: Section 2 presents state-of-the-art inspection systems using a VTOL platform and bio-inspired robots. Section 3 defines the coordinate systems used while Sections 4 and 5 describe the estimation

[2] 2011 National Electronic Injury Surveillance System Data Highlights. `http://www.cpsc.gov`

and controller design respectively. Section 6 describes how the user's controllable degrees of freedom (DOFs) are reduced. Experimental results are presented in Section 7 and 3D pole reconstruction is discussed in Section 8. Conclusions are drawn in Section 9.

2 Related Work

Considerable growth in sensor and integrated circuit technology has accelerated small and light-weight flying robot development for inspections. Voigt et al.[3] present an embedded stereo camera based egomotion estimation approach and demonstrate its applicability to boiler inspections. Based on this work, Burri et al.[4] and Kikolic et al.[8] show visual inspection of a thermal power plant boiler system using a quadrotor. They develop an FPGA-based visual-inertial stereo Simultaneous Localization and Mapping (SLAM) sensor and state updates at 10 Hz. A Model Predictive Controller (MPC) is used for closed-loop flights in industrial boiler environments. Ortiz et al.[9] demonstrate autonomous vessel inspection using a quadrotor platform. A laser scanner is utilized for horizontal SLAM and a downward-facing camera holds the vehicle in its vertical axis. Vision-based relative state estimation offers a weight and power consumption advantage over laser-based estimation, and the image data is often more useful for inspection purposes.

High-update rate image sensing techniques in light-weight aerial robotics are gaining momentum. These sensors allow quick response to disturbances and yield robust, smooth maneuvers. Recent research has demonstrated the advantages of using these fast rate sensors [10][12]. Barry demonstrates a bird-inspired high-speed (7 m/s) aircraft system. The vehicle is able to fly through a vertical gap which is narrower than its wingspan by rolling to vertical. He argues that high precision roll-rate and velocity estimation are required and these are obtained by using a high-speed camera and an IMU. In [2] we discuss the impact of high-update rate sensing by demonstrating state estimation accuracy with different sample rates. Although high measurement rates may improve overall quality [5], there is a trade-off between computational power and flight time for aerial robots. To the best of our knowledge, the ability to process QVGA images above 100 Hz including Sobel masking, RANSAC and line tracking onboard a small multi rotor is still a challenge given state-of-the-art technology.

Climbing robots [6] [7] offer an alternative for pole inspections. They can potentially carry larger payloads and allow for contact-based inspection techniques however they are unable to offer vantage points of protruding hardware such as the cross-arms of power distribution poles. They are also unable to bypass hardware mounted part-way up the poles such as transformers.

3 Coordinate System Definition

Four right-handed coordinate frames are defined for this work as shown in Fig. 2: the World $\{W\}$, Body $\{B\}$, Camera $\{C\}$ and Laser Tracker $\{L\}$ frames. Note that

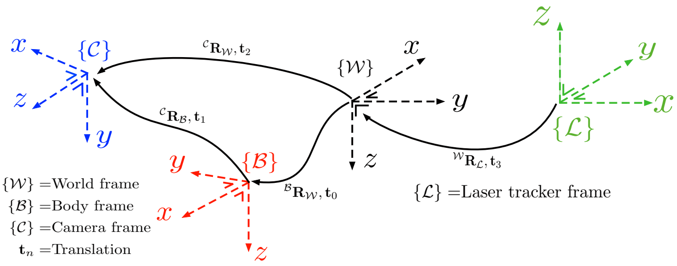

x
$\{\mathcal{C}\}$
z
y
${}^{c}\mathbf{R}_{\mathcal{W}}, \mathbf{t}_2$
$\{\mathcal{W}\}$
x
y
z
${}^{c}\mathbf{R}_{\mathcal{B}}, \mathbf{t}_1$
y
$\{\mathcal{B}\}$
x
z
${}^{\mathcal{B}}\mathbf{R}_{\mathcal{W}}, \mathbf{t}_0$
z
y
x
$\{\mathcal{L}\}$
${}^{\mathcal{W}}\mathbf{R}_{\mathcal{L}}, \mathbf{t}_3$
$\{\mathcal{L}\}$ =Laser tracker frame
$\{\mathcal{W}\}$ =World frame
$\{\mathcal{B}\}$ =Body frame
$\{\mathcal{C}\}$ =Camera frame
$\mathbf{t}_n$ =Translation

Fig. 2 Illustration of the coordinate systems used. We assume the transformation between $\{B\}$ and $\{C\}$ is constant.

both $\{W\}$ and $\{B\}$ have downward-pointing z-axes while $\{C\}$ has its z-axis (camera optical axis) in the horizontal plane. We define the notation ${}^{a}\mathbf{R}_{b}$ which rotates a vector defined with respect to $\{b\}$ to a vector with respect to $\{a\}$. All measurements and state estimation are transformed to $\{W\}$.

4 Vision and Inertial-Based State Estimation

This section summarizes the vision-based feature tracking [11] and state estimation [2] used in this work. A camera tracks the two vertical edges of a pole and there are two phases: bootstrapping and tracking. Initially the two edges are extracted from a horizontal gradient (Sobel kernel) image using Canny edge detection and Hough transforms. A line tracker performs line searching and line model fitting using Random Sample Consensus (RANSAC) [19]. We implement an IMU-aided line tracker to improve tracking performance (e.g., during aggressive motions). The projection of the 3D pole edge into 2D image coordinates is predicted between frames using a linear feature velocity model and the IMU data. In order to calculate feature velocity, we compute an *image Jacobian* which describes how a line moves on the image plane as a function of camera spatial velocity. These tracked lines are fused in a 100 Hz Extended Kalman Filter (EKF) together with IMU data to estimate pole-relative vehicle horizontal states such as stand-off distance, lateral offset and angular and linear rates (the *Horizontal EKF*). We assume the diameter of the pole is known and use this to establish the unknown scale factor. This *Horizontal EKF* is validated by using a MATLAB camera simulation framework [20].

Note there is an ambiguity for sideway motions (e.g., left and right) and yaw motions as both result in the target appearing to move horizontally in the image. It is a challenge to decouple these motions without using additional sensors hence we omit heading angle estimation in the EKF states and assume it is controlled independently.

Height (above ground) is estimated by a Kalman Filter (KF) which fuses data from a downward-facing sonar with accelerometer rate estimates at 70 Hz (the *Vertical KF*) . Fig. 3 illustrates an input/output system diagram of the Vertical KF and

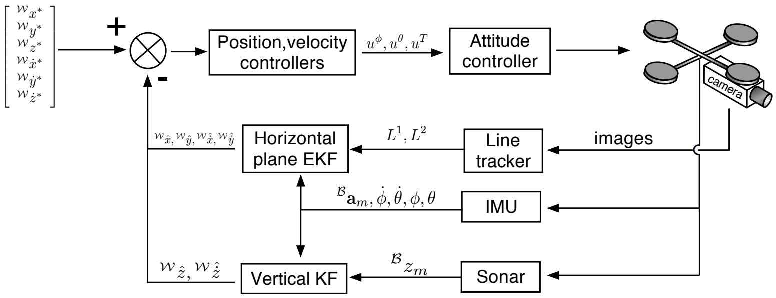

$\begin{bmatrix} {}^{\mathcal{W}}x^* \\ {}^{\mathcal{W}}y^* \\ {}^{\mathcal{W}}z^* \\ {}^{\mathcal{W}}\dot{x}^* \\ {}^{\mathcal{W}}\dot{y}^* \\ {}^{\mathcal{W}}\dot{z}^* \end{bmatrix}$
+
-
Position,velocity controllers
$u^{\phi}, u^{\theta}, u^{T}$
Attitude controller
camera
$^{\mathcal{W}}\hat{x}, {}^{\mathcal{W}}\hat{y}, {}^{\mathcal{W}}\hat{\dot{x}}, {}^{\mathcal{W}}\hat{\dot{y}}$
Horizontal plane EKF
L^1, L^2
Line tracker
images
$^{\mathcal{B}}\mathbf{a}_m, \dot{\phi}, \dot{\theta}, \phi, \theta$
IMU
$^{\mathcal{W}}\hat{z}, {}^{\mathcal{W}}\hat{\dot{z}}$
Vertical KF
$^{\mathcal{B}}z_m$
Sonar

Fig. 3 Input/output diagram for the Vertical KF and Horizontal EKF. L^1 and L^2 denote two observed lines in the image plane. An IMU provides angular rates, $\dot{\theta}$ and $\dot{\phi}$, angles, θ and ϕ, and acceleration measurements, $^{B}\mathbf{a}_m$, in $\{B\}$. $^{B}z_m$ is a height measurement. The estimated filter outputs and the desired goals are denoted with a hat and a star superscription respectively. u^{ϕ}, u^{θ} and u^{T} are control outputs for pitch, roll and thrust. We fuse different sample rates of data: the 100 Hz line tracker (vision), a 70 Hz IMU and a 20 Hz sonar. The latest-updated measurement is used for slow update rate sensors.

Horizontal EKF. We use two separate filters since the front camera line measurement is nonlinear and needs to be linearized whereas the height measurement is linear and observable. 70 Hz is sufficient for height control considering its slower dynamics compared to pitch/roll. Although pitch/roll/thrust dynamics are coupled, we can treat them independently when assuming small attitude angles.

5 Vision-Based Control with Camera Latency Estimates

A quadrotor platform is an under-actuated dynamic system that is force actuated and undamped. This makes it challenging to control as high-quality velocity signals are required. Latencies in a system are a crucial dynamic characteristic and significantly effect control performance. For a machine vision camera-computer system there is a pixel transport delay which accounts for the latency between an image being exposed on the camera sensor and it being available in an image buffer on the computer for processing (See Fig. 5). Our quadrotor controller design is presented in [2] however this previous work did not account for the camera latency. We augment the controller by estimating the latency as described in Section 5.1 and including it in the control loop as shown in Figure 4.

5.1 Camera Latency Estimation

We estimate the pixel transport delay of our system by using an LED and an onboard microprocessor (setup shown in Fig. 6). A command is sent from a computer to the microprocessor which turns on/off an LED with micro-second latency and records the timestamp at that moment. At the same time a high-speed camera captures im-

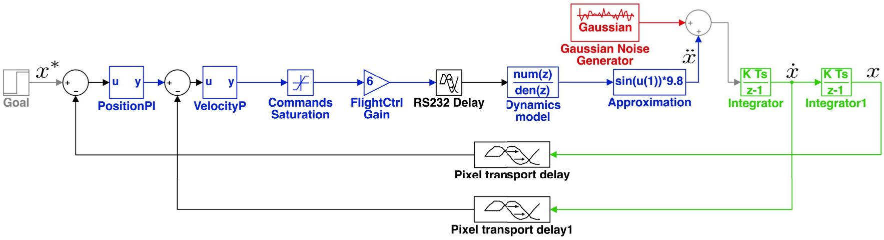


Fig. 4 x-axis position controller with outer proportional-integral position loop and inner proportional velocity loop. Communication (8.3 ms) and pixel transport delay (12 ms) are included along with zero-mean Gaussian, $N_{\ddot{x}} \sim (0, 0.5\,\mathrm{m/s^2})$ representing acceleration noise.

ages of this event at 100 Hz together with timestamps. The images and timestamps are analysed to measure the pixel transport delay for each event. The mean latency for 24 trigger events was measured at 8.4 ms with standard deviation 4.23 ms. We incorporated this latency in the control loop shown in Fig. 4.

6 Reduction of User-Controllable DOFs

Our proposed system is modelled on Sheridan's "Supervisory Control" architecture [23], specifically system 4, in which the control loop is closed through a computer but there are still human interventions. This approach allows the robot to close the high-bandwidth control loops while the on-demand "high level" commands from the human are treated as requested goal states.

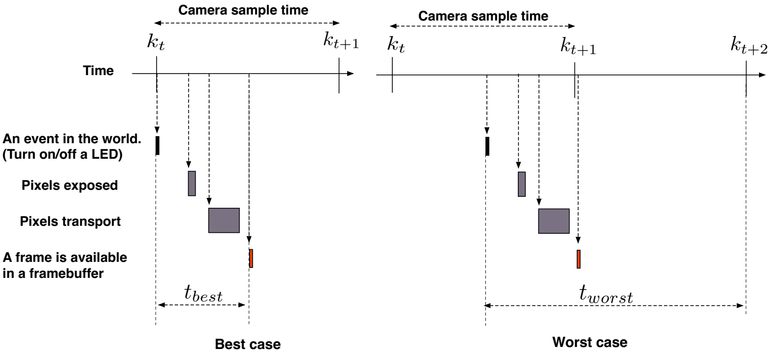


Fig. 5 The left and right figure illustrate the best and the worst scenarios. t_{best} and t_{worst} denote the corresponding latency between the time when an actual event happens and an image is stored in a framebuffer. It is practically difficult to measure pixel transport delay, however we can guarantee the delay has to be within $t_{best} \sim t_{worst}$. The measurement shows that it lies within this range. We assume the pixel transport latency is shorter than the camera sample time.

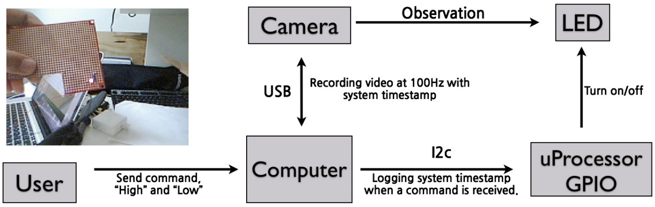

Camera
Observation
LED
USB
Recording video at 100Hz with system timestamp
Turn on/off
User
Send command, "High" and "Low"
Computer
I2c
Logging system timestamp when a command is received.
uProcessor GPIO

Fig. 6 Experiment setup for measurement of pixel transport latency. The interface latency between the computer and microprocessor is negligible compared to the pixel transport time. We assume an actual event occurs at the same time as the microprocessor event.

A typical attitude-stabilised quadrotor has four user-controllable degrees of freedom (DOF), namely horizontal position (x,y), height (z), and heading (ψ). These are usually controlled indirectly with joysticks where the stick positions are mapped to rates (e.g. 'throttle' stick position is mapped to climb rate). Significant operator skill is therefor required to control position in 3-dimensional Cartesian space. Even more skill is required when flying close to structures as it is hard to judge the position of the vehicle relative to the structure, and there is little room for error when correcting for wind and turbulence. We propose reducing the operator's cognitive load and level of skill required by reducing the controllable DOFs and letting the system control the remaining DOFs automatically. Additionally, some of the DOFs are controlled in a more direct, intuitive manner rather than indirectly via rate commands.

The proposed concept is shown in Fig. 7(d) for a pole inspection task, where the operator controls only 2 DOF: altitude of the vehicle and angle around the pole. Given a position controller that keeps the pole centred in the field of view, and at a constant stand-off distance, then Fig. 7(a) $\sim$ Fig. 7(c) illustrates how the vehicle moves around the pole when a yaw command is given. This is sufficient for inspection of the entire pole area and easy to control.

7 Flight Experiments

In this section the experimental setup is described and we present results for pole-relative hovering and user-controlled pole circumnavigation.

7.1 Experimental Setup

Our MikroKopter quadrotor platform carries a forward-facing camera for line detection (75° field of view (FOV), 320×240 images at 100 Hz). An ultrasonic sensor provides height measurements at 20 Hz. All processing is performed by an onboard single board computer (SBC). Further details are provided in [2]. An actuated surveying laser (Leica TS12) is used to track a reflective prism on the vehicle providing ground truth position with millimeter accuracy at 5 Hz.

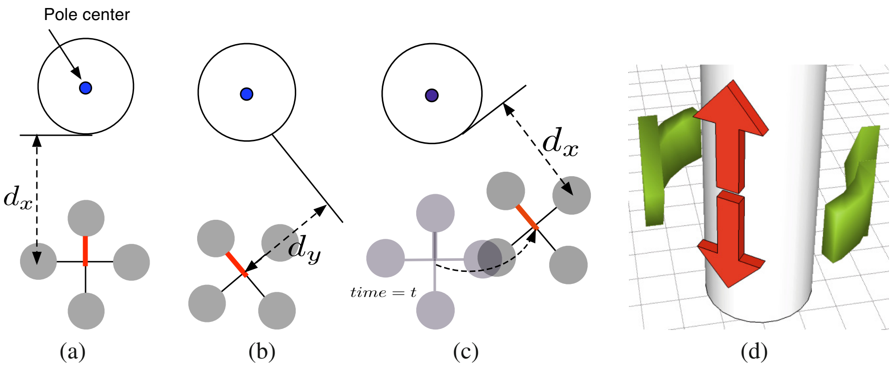


Fig. 7 Changing yaw angle makes the quadcopter circle around the pole (red bar indicates the *front* rotor). References for x,y position controllers are d_x and 0 respectively. The robot hovers by keeping d_x distance at $time = t$. (b) An operator sends a yaw command and it introduces d_y distance at $time = t+1$. (c) The robot moves to the right to eliminate d_y and keeps d_x distance at $time = t+2$. (d) Reduced controllable DOFs. The operator is only allowed to move along the arrow directions which are sufficient for inspection purposes.

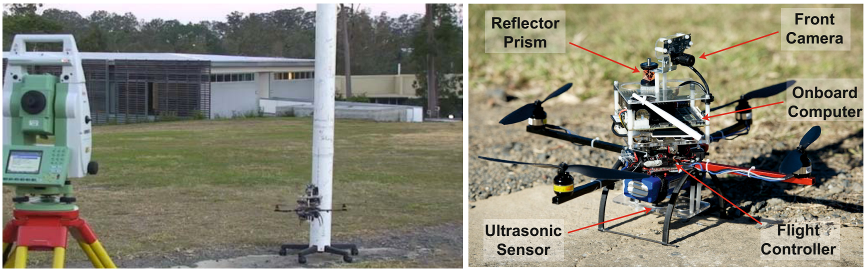


Fig. 8 Experimental setup showing the Leica laser tracker in the foreground and the UAV and pole in the background (left) and a close-up view of the UAV (right)

Fig. 8 shows the vehicle and the experimental environment while Fig. 9 shows the system diagram. Different colors in the figure denote different sampling rates and arrows denote data flow at a given frequency. Each box is an individual Robot Operating System (ROS) [26] node implemented using C++. Precision Time Protocol (PTP) is utilized for time synchronization between the onboard computer and the ground station. The IMU on a flight control board provides $[\boldsymbol{\Phi}, \dot{\boldsymbol{\Phi}}, \mathbf{a}]$ where $\boldsymbol{\Phi}$ is the roll-pitch-yaw angles $[\phi, \theta, \psi]$, $\dot{\boldsymbol{\Phi}}$ the RPY angle rates and $\mathbf{a}$ the 3-axis acceleration at 70 Hz. $\mathbf{u}$ denotes computed pitch, roll and thrust commands from controllers, $[u^p, u^r, u^h]$.

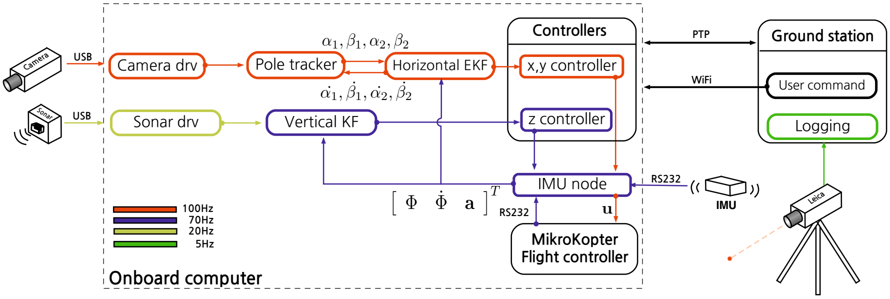


Fig. 9 Software system diagram. Different colors denote corresponding sampling rates. All software is implemented using ROS.

7.2 *State Estimation Results*

We performed 32 pole-relative hovering flights using the experimental setup described in Section 7.1 and observed a success rate of 78% (25/32) where the system was able to track the pole and maintain a hover position relative to it. An altitude controller maintained a constant height allowing us to evaluate the horizontal controller performance independently. The results for the period 10 $\sim$ 60s of a 70 second flight are summarised in Table 1. The performance of the 100 Hz horizontal EKF and 70 Hz vertical KF estimators for the flight are shown in Fig. 10. The estimated position and velocity are evaluated by down-sampling the filter estimation result to 5 Hz and computing the standard deviation of errors between this and the laser tracker ground truth.

Table 1 Standard deviations of state estimation errors for the 100 Hz horizontal EKF and 70 Hz vertical KF

State variable	Vertical KF	Horizontal EKF	units
$\hat{x}$	—	0.055	m
$\hat{y}$	—	0.05	m
$\hat{z}$	0.011	—	m
$\hat{\dot{x}}$	—	0.197	m/s
$\hat{\dot{y}}$	—	0.092	m/s
$\hat{\dot{z}}$	0.065	—	m/s
Interval	10∼60	10∼60	sec

The estimation results track the ground truth but appear noisier since the sample rate of the estimators is up to 20 times higher than the ground truth measurement and the quadrotor plant effectively behaves as a low-pass filter. We see a similar effect on velocities as shown in Fig. 11.

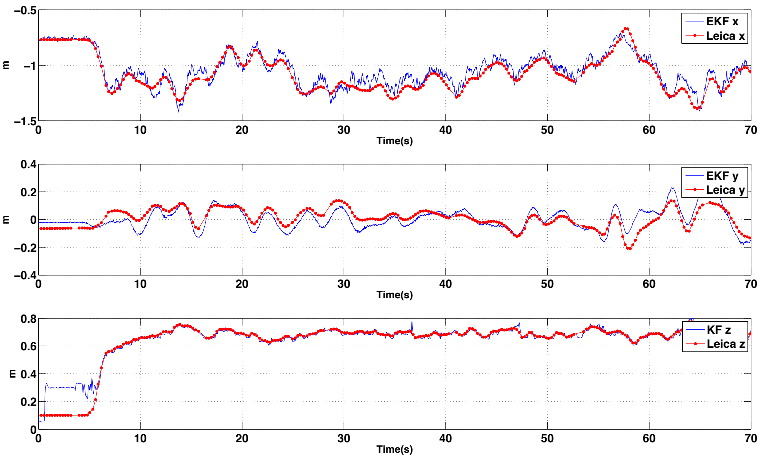


Fig. 10 Experimental results for position estimation while hovering. The first and second rows show the 100 Hz horizontal EKF estimation with the 5 Hz ground truth whereas the third is the 70 Hz vertical KF estimation with the ground truth. -1 m, 0 m and 0.6 m are the desired position for ${}^{W}\hat{x}$, ${}^{W}\hat{y}$ and ${}^{W}\hat{z}$. Note that z-axis is inverted for visualization.

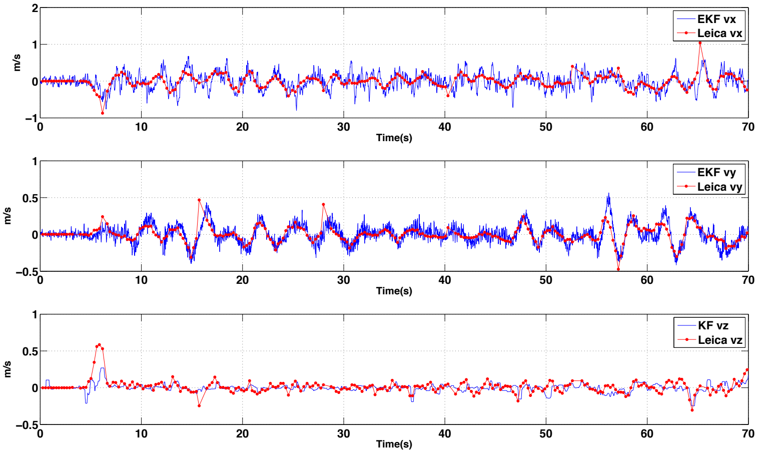
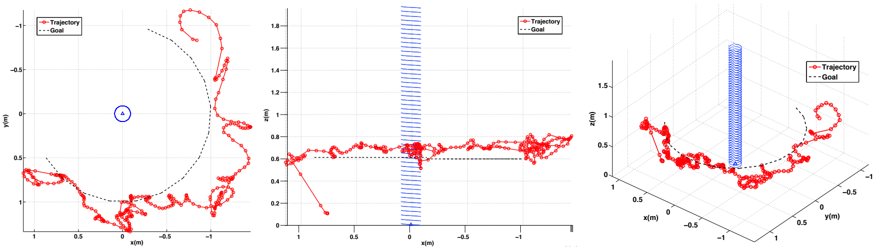


Fig. 11 Experimental results for velocity estimation, 100 Hz ${}^{W}\hat{\dot{x}}$, ${}^{W}\hat{\dot{y}}$ and 70 Hz ${}^{W}\hat{\dot{z}}$, with the 5 Hz ground truth while hovering. The desired velocity is taken as the output of the position controller as shown Fig. 4.

The spikes in the ground truth velocity data (e.g. at around 6, 18 and 29 seconds) correspond to occasions where the laser tracker momentarily lost a fix on the reflective prism so did not produce data at a consistent rate (usually when the vehicle was



(a) Top view (b) Side view (c) Perspective view

Fig. 12 Ground truth trajectory for a pole inspection flight. An operator only sends yaw commands using the RC transmitter and the UAV keeps the desired distance, d_x, d_y, d_z =[1, 0, 0.6] (m), from the pole. Black denotes the reference. Note that only the ground truth trajectory is presented due to difficulty in estimating yaw angle with a low performance gyroscope.

moving too fast for the tracker). The corresponding video demonstration is available on our YouTube channel[3].

7.3 *Pole Inspection Task*

The envisioned mode of operation for a pole inspection is for the operator to place the UAV on the ground with the camera facing the pole to be inspected and at the desired stand-off distance for inspection. The operator will then command only height and yaw to move the UAV around the pole at different heights while capturing inspection images. The system will keep the camera oriented towards the pole and maintain the stand-off distance. We emulated this task with the experimental setup described in Section 7.1 however the stand-off distance (d_x) was pre-set to 1 m. We performed 12 flights and for five of these the vehicle was able to circumnavigate the pole successfully. Failures occurred when the pole left the camera FOV and tracking was lost. Fig 12 displays different viewpoints of the trajectory for one of these flights. At the time the average wind speed was 1.5 m/s with gusts of up to 2.4 m/s (See the demonstration video). Note that the pole was successfully circumnavigated for this experiment however ground truth is only available for part of the flight as the laser tracker could not track the vehicle when it was occluded by the pole.

7.4 *Discussion of Results and Limitations*

Figs 10 and 11 show that vision + imu-based state estimation allows for pole-relative hovering and Fig. 12 shows that a user can command the UAV to circumnavigate a pole by providing only yaw commands. Failures did occur however and these were primarily when the pole left the camera FOV and could no longer be tracked. The

[3] YouTube channel, `http://youtu.be/Bv55g6wTw0c`

top view of Fig. 12 shows that the initial stand-off distance was roughly maintained, with the vehicle deviating up to approximately 70 cm from the desired radius around the pole.

A current limitation of the system is due to the poor yaw estimates produced by the onboard IMU. Yaw control is based on this estimate so a drift in yaw estimation causes a yaw rotation of the vehicle, which in turn yields a y-axis controller error (as shown in Fig. 7). The vehicle therefore keeps rotating about the pole even without operator input. Improving the yaw angle estimates by means of a magnetometer or a visual compass is an area of future work.

Another limitation of the system is its susceptibility to the effects of direct sunlight and shadows found in outdoor environments. Edges are weak under strong sunlight due to the small intensity difference between the pole edges and the background. Shadows on the other hand can create an intensity gradient on the surface of the pole and this may be falsely detected and tracked as the pole edge. To avoid these challenges our experiments were conducted in the absence of direct sunlight (early morning or late afternoon), and we will improve the robustness to these effects in the future.

8 3D Pole Reconstruction

For inspection purposes it is important to record the quality of an asset over its life-time. This is typically done with high resolution images however individual images only offer discrete viewing angles. Computer vision techniques allow 3D reconstruction of an object from multiple images, after which it can be viewed from a variety of angles. To demonstrate the feasibility of reconstructing a pole in 3D and the utility of having a texture-mapped 3D pole for inspection purposes, we utilised two software tools on an image sequence in the workflow shown in Fig. 13. A GoPro camera (170° FOV) was mounted to the vehicle and used to capture images at 240 FPS while flying around the pole. The high capture rate was used to reduce motion blur, but the sequence was subsampled to 10 Hz for processing. The undistorted image sequence was first fed into a Structure from Motion (SfM) software tool to obtain camera poses [24]. A texturisation software tool was then used to produce a dense point cloud and a textured mesh of the scene [25]. The reconstruction results are shown in Fig. 14. The software tools rely on point features and since the pole surface is relatively textureless, the surface reconstruction is rather coarse. Once texture-mapped it still however provides the user with a 3D view of the pole from a variety of angles with sufficient clarity for inspection purposes.

More views are shown in the demonstration video[4].

[4] YouTube channel, `http://youtu.be/Bv55g6wTw0c`

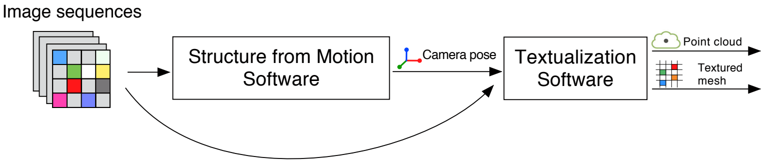


Fig. 13 Pole reconstruction workflow. Input for reconstruction is an undistorted image sequence which are recorded at 240 Hz and sub-sampled to 10 Hz. Full pairwise matching is performed.

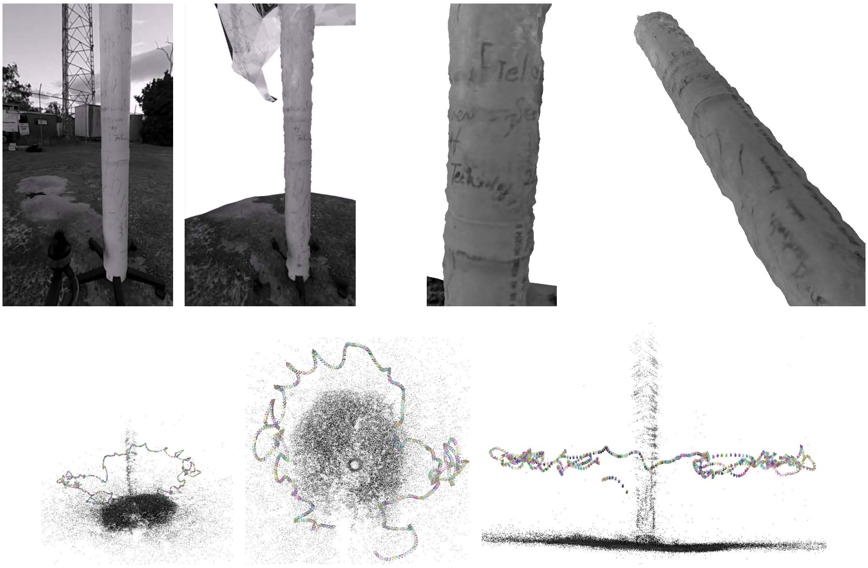


Fig. 14 3D pole reconstruction results. An original image (top left) and various views of the texture-mapped surface (top row). SfM-based camera trajectory estimation (bottom row). The scale of the trajectory is arbitrary, up-to-scale, since we use a monocular camera.

9 Conclusion and Future Work

We have presented a UAV-based pole inspection system using an onboard high-speed camera and IMU for pole-relative navigation in outdoor environments. Pixel transport latency for the system is measured and incorporated in the control loop. Translational position and velocity estimation are evaluated through outdoor flight tests with accurate ground truth data. By reducing the user-controllable DOFs we show a user is easily able to fly the UAV around a pole at a fixed stand-off distance by only giving yaw commands. We also demonstrate the feasibility of reconstructing a 3D texture-mapped model of the pole and the utility of this for inspection purposes.

Our system has a number of limitations that will be addressed in the future. These include using a visual compass from a downward-facing camera to improve yaw estimates, and making the line tracking algorithm more robust in direct sunlight conditions. Since the sonar only works reliably up to 2 m, we plan to integrate visual odometry along the vertical axis of the structure to estimate height.

Acknowledgements. We would like to thank Navinda Kottege from CSIRO for providing the Leica ROS software and Edward Pepperell for supplying a GoPro3 camera.

References

1. Sa, I., Corke, P.: Vertical Infrastructure Inspection using a Quadcopter and Shared Autonomy Control. In: Yoshida, K., Tadokoro, S. (eds.) Field and Service Robotics. STAR, vol. 92, pp. 219–232. Springer, Heidelberg (2012)
2. Sa, I., Corke, P.: Close-quarters Quadrotor Position Based Visual Servoing with a 100Hz Monocular Camera. In: IEEE International Conference on Robotics and Automation (submitted, 2014)
3. Voigt, R., Nikolic, J., Hurzeler, C., Weiss, S., Kneip, L., Siegwart, R.: Robust Embedded Egomotion Estimation. In: International Conference on Intelligent Robots and Systems, pp. 2694–2699 (2011)
4. Burri, M., Nikolic, J., Hurzeler, C., Caprari, G., Siegwart, R.: Aerial service robots for visual inspection of thermal power plant boiler systems. In: International Conference on Applied Robotics for the Power Industry, pp. 70–75 (2012)
5. Handa, A., Newcombe, R.A., Angeli, A., Davison, A.J.: Real-Time camera tracking: when is high frame-rate best? In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VII. LNCS, vol. 7578, pp. 222–235. Springer, Heidelberg (2012)
6. Allan, J.F., Lavoie, S., Reiher, S., Lambert, G.: Climbing and pole line hardware installation robot for construction of distribution lines. In: 2010 1st International Conference on Applied Robotics for the Power Industry, CARPI (2010)
7. Ahmadabadi, M.N., Moradi, H., Sadeghi, A., Madani, A., Farahnak, M.: The evolution of UT pole climbing robots. In: 2010 1st International Conference on Applied Robotics for the Power Industry, CARPI (2010)
8. Nikolic, J., Burri, M., Rehder, J., Leutenegger, S., Huerzeler, C., Siegwart, R.: A UAV system for inspection of industrial facilities. In: IEEE Conference on Aerospace Conference, pp. 1–8 (2013)
9. Ortiz, A., Bonnin-Pascual, F., Garcia-Fidalgo, E.: Vessel Inspection: A Micro-Aerial Vehicle-based Approach. Journal of Intelligent & Robotic Systems, 1–17 (2013)
10. Cutler, M., Michini, B., How, P.: Lightweight Infrared Sensing for Relative Navigation of Quadrotors. In: International Conference on Unmanned Aircraft Systems (2013)
11. Sa, I., Corke, P.: Improved line tracker using IMU and Vision for visual servoing. In: Australasian Conference on Robotics and Automation (2013)
12. Barry, A.: Flying Between Obstacles with an Autonomous Knife-Edge Maneuver. Masters Thesis, Massachusetts Institute of Technology (2012)

13. Bartoli, A., Sturm, P.: Structure-from-motion using lines: Representation, triangulation, and bundle adjustment. Computer Vision and Image Understanding 100, 416–441 (2005)
14. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press (2003)
15. Mahony, R., Kumar, V., Corke, P.: Modeling, Estimation and Control of Quadrotor Aerial Vehicles. IEEE Robotics Automation Magazine (2012)
16. Sa, I., Corke, P.: 100Hz Onboard Vision for Quadrotor State Estimation. In: Australasian Conference on Robotics and Automation (2012)
17. Grabe, V., Bulthoff, H.H., Giordano, P.R.: On-board velocity estimation and closed-loop control of a quadrotor UAV based on optical flow. In: IEEE International Conference on Robotics and Automation, pp. 491–497 (2012)
18. Shiyu, Z., Feng, L., Kemao, P., Ben, M.C., Tong, H.L.: Homography-based Vision-aided Inertial Navigation of UAVs in Unknown Environments. In: AIAA Guidance, Navigation, and Control Conference (2012)
19. Martin, F., Robert, B.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM (1981)
20. Corke, P.: Robotics, Vision & Control Fundamental algorithms in MATLAB. STAR, vol. 73. Springer, Heidelberg (2011)
21. Kalman, R.E.: A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME - Journal of Basic Engineering 82, 35–45 (1960)
22. Sa, I., Corke, P.: System Identification, Estimation and Control for a Cost Effective Open-Source Quadcopter. In: IEEE International Conference on Robotics and Automation, pp. 2202–2209 (2012)
23. Sheridan, T.: Telerobotics, automation, and human supervisory control. The MIT Press (1992)
24. Wu, C.: Towards Linear-time Incremental Structure from Motion. In: 3D Vision Conference (2013)
25. Jancosek, M., Pajdla, T.: Multi-View Reconstruction Preserving Weakly-Supported Surfaces. In: IEEE Conference on Computer Vision and Pattern Recognition (2011)
26. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T.B., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: an open-source Robot Operating System. In: ICRA Workshop on Open Source Software (2009)

Inspection of Penstocks and Featureless Tunnel-like Environments Using Micro UAVs

Tolga Özaslan, Shaojie Shen, Yash Mulgaonkar, Nathan Michael, and Vijay Kumar

Abstract. Micro UAVs are receiving a great deal of attention in many diverse applications. In this paper, we are interested in a unique application, surveillance for maintenance of large infrastructure assets such as dams and penstocks, where the goal is to periodically inspect and map the structure to detect features that might indicate the potential for failures. Availability of architecture drawings of these constructions makes the mapping problem easier. However large buildings with featureless geometries pose a significant problem since it is difficult to design a robust localization algorithm for inspection operations. In this paper we show how a small quadrotor equipped with minimal sensors can be used for inspection of tunnel-like environments such as seen in dam penstocks. Penstocks in particular lack features and do not provide adequate structure for robot localization, especially along the tunnel axis. We develop a Rao-Blackwellized particle filter based localization algorithm which uses a derivative of the ICP for integrating laser measurements and IMU for short-to-medium range pose estimation. To our knowledge, this is the only study in the literature focusing on localization and autonomous control of a UAV in 3-D, featureless tunnel-like environments. We show the success of our work with results from real experiments.

1 Introduction

Recently, micro UAVs have attracted significant attention in a variety of civilian applications due to their low cost and superior mobility. One possible application is the use UAVs in inspection of large buildings such as dams and penstocks. Penstocks

Tolga Özaslan · Shaojie Shen · Yash Mulgaonkar · Vijay Kumar
University of Pennsylvania, 19104, Philadelphia, PA, USA

Nathan Michael
Carnegie Mellon University, 15213, Pittsburgh, PA, USA
e-mail: {ozaslan,shaojie,yashm,kumar}@seas.upenn.edu,
nmichael@cmu.edu

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_9, © Springer International Publishing Switzerland 2015

are constructions that require regular maintenance, and this in turn requires visual inspection of the interior. However penstocks are dark, long and featureless tunnels that slope steeply down hillsides. Because of this, it is hard for humans to climb up penstocks and perform visual inspection. We propose, as an alternative, the use of quadrotors equipped with minimal sensors that can fly through the tunnels and collect data for remote inspection.

In order to reduce the operator workload, we require a high level of autonomy of the quadrotor. This in turn requires that the robot is able to localize itself with respect to features in the environment. In our case, we are given engineering drawings of the penstock, which can be converted into a map of the environment. Hence we focus on solving the problem of pose estimation (localization) and autonomous control in penstocks (tunnel-like) buildings. This work allows us to build autonomous UAVs that can collect imagery from inside penstocks for inspection with only high-level user commands.

Penstocks are almost perfectly cylindrical in cross section, and have two long, non-parallel, straight portions as shown in Fig. 2. In most penstocks, the first portion is on a horizontal plane and the second part slopes upwards. The interior of the penstock is built with rectangular shaped steel plates of approximately 6 square meters each bent into a cylindrical geometry. Using the given engineering drawings (the map), IMU data and scanner readings, it is always possible to determine the orientation, height and lateral position of the quadrotor. However the position along the tunnel axis cannot be always derived due to the special geometry of the map and the available sensors. For example, when the distance between the robot and the junction of the tunnel is greater than the maximum measurement range of the laser scanner, the position along the tunnel cannot be determined. However, during the transition between the horizontal and inclined portions of the tunnel, scanner readings show significant differences which can help in localizing the robot along the axis of the tunnel at that particular region.

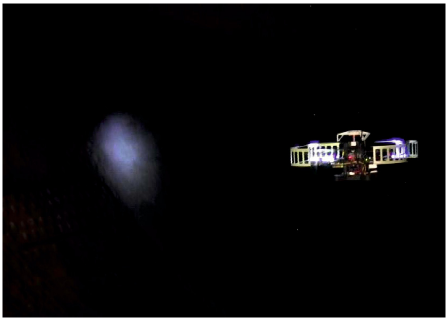


Fig. 1 A quadrotor flying inside a penstock at Allatoona Dam, GA. Lighting is provided by a portable spot light. The quadrotor is equipped with a 1.6 GHz Atom Intel processor, Hokuyo [2] laser scanner and an IMU unit. Note that the installation of lights for illuminating the entire tunnel is impractical. Therefore, we equip our quadrotor with LED lights as shown in the figure.

We stress that it is difficult for ground robots to operate in the tunnel. While a slope of 23 degrees (see Fig. 2) can be easily negotiated by tracked vehicles, the tunnel is very slippery and smooth and it is difficult to use vehicles that require traction in the tunnel. Furthermore, since the tunnel is not illuminated, inspection of ceilings and walls can be difficult and may require auxiliary lighting equipment which can be heavy and require a lot of power. Fortunately, since our quadrotor can fly close to the walls (and the ceiling) it can use energy efficient LED lights and obtain illumination for collecting imagery. In addition to lights, the platform can be equipped with different inspection sensors such as infrared cameras.

As shown in Fig. 1, we customize a Pelican quadrotor by Ascending Technologies [1]. The robot is equipped with a 1.6 GHz Atom Intel processor, a Hokuyo [2] laser scanner, an IMU unit and LED lights (Fig. 3).

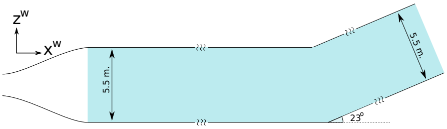


Fig. 2 Side view of a representative penstock (exact diameter and slope from the Carter Dam penstock)

A hallmark reference [7] introduced the framework of Monte Carlo Localization (MCL). Variants of such localization algorithms can be seen for museum guide robots [21] , human operated backpack [16] and robot with 3-D laser scans [15]. However, the highly symmetric and feature-less tunnel environment poses problems for existing localization algorithms. Furthermore, processing a large amount of data using low power, light weight on-board computer proves to be challenging. Also, algorithms relying on GPS are not practical for quadrotors flying inside a tunnel.

There is also extensive literature on localization using cameras. [3] fuses stereo vision, GPS, and IMU to perform outdoor localization. In another outdoor localization study, [14] tests image-based localization with wide angle cameras. Scale-invariant features are used in [19] to both localize and build 3D maps of office environments. However, it is hard to apply this method in real-time due to the limited on-board computation. Further, none of the above approaches will work in a penstock due to poor lighting conditions. In our case, although we use lighting, we need it only for detecting rust and cracking in the interior surface and not for localization.

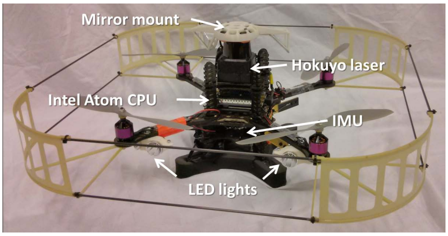


Fig. 3 Our quadrotor prototype equipped with a Hokuyo laser scanner, on-board IMU and two flash lights. A 3-D printed laser mount redirects some of the laser beams upward and downward.

The rest of the paper is organized as follows: We start by reviewing basic background in Section 2. We then present our system for localization of a quadrotor in the penstock in Section 3. The key contributions in this paper are the novel measurement models those are designed based on the unique geometry of the penstock and semi-autonomous operation in featureless tunnels. These are both presented in Section 3. Finally, field experimental results are presented in Section 4.

2 Background

2.1 *Quadrotor Dynamics*

Quadrotors are basically helicopters with four propellers located at corners of a square shape. A schematic of a quadrotor is given in Fig. 4. Each propeller is located at equal distances from the geometric center of the quadrotor. Motors mounted on opposite sides rotate in the same direction, while the others in the opposite direction. Ideally, while the quadrotor is stationary, moments due to the propellers rotating in opposite directions cancel each other so that the yaw is kept constant.

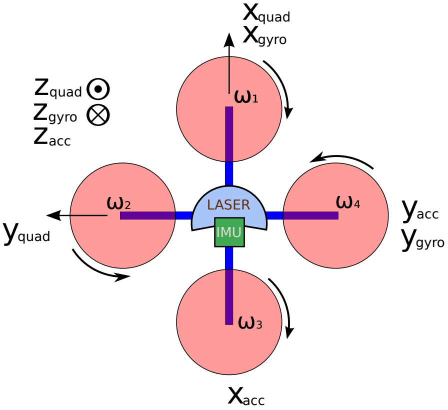


Fig. 4 Coordinate frame definitions of quadrotor. Due to the manufacturer, gyroscope and accelerometer has different orientations which are shown with subscripts of *gyro* and *acc*. And body frame is denoted by the subscript *quad*. Dot in a circle means a vector pointing out of the paper plane and an × means the opposite [17].

As the standard reference triad (SRT) for inertial frame, we use $\{\hat{x}^{\mathcal{W}}, \hat{y}^{\mathcal{W}}, \hat{z}^{\mathcal{W}}\}$ basis vectors. Then a vector in this frame is represented by the vector $\left[x^{\mathcal{W}}, y^{\mathcal{W}}, z^{\mathcal{W}}\right]^T$. Whereas SRT of the body frame is defined with the basis vectors $\{\hat{x}^{\mathcal{B}}, \hat{y}^{\mathcal{B}}, \hat{z}^{\mathcal{B}}\}$ and a vector in this frame is represented as $\left[x^{\mathcal{B}}, y^{\mathcal{B}}, z^{\mathcal{B}}\right]^T$. $\hat{x}^{\mathcal{B}}$ is the heading direction of the quadrotor which can be selected arbitrarily. $\hat{z}^{\mathcal{B}}$ is preferably selected as the upwards direction when the quadrotor is hovering and $\hat{z}^{\mathcal{W}}$ is selected to be pointing in the opposite direction to the gravitation (see Fig. 4 and its caption for illustration). Rotation between these two frames is carried through multiplication with a rotation matrix $R \in SO(3)$ and denoted by ${}^{\mathcal{B}}R_{\mathcal{W}}$. Subscript is the frame from which the vector will be transformed and the pre-superscript is the goal frame.

We use $Z - X - Y$ Euler angles to represent rotation from world to body frame [18]. Yaw, pitch and roll angles are denoted as ψ, θ and ϕ respectively. Angular velocity in the body frame is denoted by the vector $\left[p, q, r\right]^{\mathcal{B}}$

We refer to the work by Mellinger [18] where detailed derivations of dynamic equations are given. They also linearize about the hover state and present a linear controller based on this model.

2.2 Robot Localization

Robot localization, environment mapping and the merging of these two problems, Simultaneous Localization and Mapping (SLAM), has been studied extensively [4, 5, 8, 9, 20, 21]. Filtering based approaches are commonly used for solving the localization problem. Two mostly used approaches are based on the Kalman filter and the particle filter.

For systems that satisfy the Gaussian uncertainty model, the Kalman filter and its nonlinear variants (referred to as KF from this point on) yield efficient and robust results. We choose the Unscented Kalman Filter (instead of the standard Kalman filter) due to its ability to approximate the propagation of Gaussian random vectors through nonlinear functions via the propagation of stochastic linearization points [20].

On the other hand, there are many systems with multi-modal, widely spread, and other uncertainty models that are cannot be modeled as Gaussian distributions. For such distributions, the nonparametric particle filter-based approach and variants (referred to as PF from this point on), also known as Monte Carlo methods [7, 21], provide approximate representations of arbitrary probabilistic distributions. They are more powerful compared to the the parametric KF-based approaches. However, for systems with relatively large number of degrees of freedom (such as quadrotors), the number of particles that is required to accurately represent the distribution can be prohibitively large.

The Rao-Blackwellized particle filter decomposes the configuration space in order to reduce the dimension of the particle-based distribution approximation. The main goal is to reduce the required particle count for the particle filter [8, 12] by designing a hybrid filter achieved by merging the PF and the KF. That is, for some of the parameters, estimation is done through KF and for others PF is used. In our application, since a robot moves through a featureless tunnel, the localization uncertainty for the position along the axis of the tunnel is high and it is hardly a proper Gaussian distribution. However , the uncertainties in position for the other two directions are small and they can be well approximated by Gaussian distributions. For the former case, use of PF is meaningful and in the latter case KF is a reasonable choice.

2.3 Controller Design

We use the linear controller design of [18]. Since our target application requires mostly stable flight with minimum linear acceleration, linearization of dynamic equations around the hover position can be justified. Our controller utilizes a backstepping architecture that consists of a position controller and an attitude controller. The high level position controller generates desired orientations based on user specified way-points and the on-board localization feedback. The low level attitude controller drives the robot to the desired orientation by adjusting motor RPMs.

As shown in Fig. 6, a trajectory generator is used to generate a trajectory from the current pose to the goal pose. At this level we can also incorporate constraints such as closest distance to walls, maximum linear and rotational speeds, and other constraints.

3 Methodology

3.1 Process and Observation Models

We define the process model with the equation

$$\mathbf{x}_{t+\Delta t} = f(\mathbf{x}_t, \mathbf{u}_t, \Delta t) \tag{1}$$

where $\mathbf{x}$ is the state vector and $\mathbf{u}$ is the control input derived from IMU. Vectors $\mathbf{x}$ and $\mathbf{u}$ are defined as:

$$\mathbf{x}^T = [x, y, z, \dot{x}, \dot{y}, \dot{z}, \psi, \theta, \phi]^{\mathcal{W}}, \tag{2}$$

$$\mathbf{u}^T = [\ddot{x}, \ddot{y}, \ddot{z}, p, q, r]^{\mathcal{B}}. \tag{3}$$

The process model implements dynamics of a quadrotor which have detailed explanations in [18]. As it is the case for MEMs sensors, our IMU has both bias and random errors. Then the true IMU data becomes

$$\mathbf{u}^* = \mathbf{u} - \mathbf{u}_{bias} - \mathbf{u}_{rnd} \tag{4}$$

where $\mathbf{u}_{rnd}$ is a random vector drawn from a normal distribution and $\mathbf{u}_{bias}$ is the bias error. The process noise in the $\hat{x}^{\mathcal{W}}$ direction is modeled by an additive random disturbance which is distributed normally with known variance.

We are using a Hokuyo laser scanner [2] which can take measurements with a 180 degrees span in the $x^{\mathcal{B}} - y^{\mathcal{B}}$ plane. A 3-D printed dual-mirror mount is fixed on top of the laser scanner to reflect rays in upward ($+\hat{z}^{\mathcal{B}}$) and downward ($-\hat{z}^{\mathcal{B}}$) directions [13] (Fig. 3). These measurements together with the orientation estimate and the knowledge of the map are used to localize robot on the $y^{\mathcal{W}} - z^{\mathcal{W}}$ plane of the tunnel using a derivative of the ICP algorithm. This algorithm uses rays emanating in the four directions $\pm\hat{z}^{\mathcal{W}}$ and $\pm\hat{y}^{\mathcal{W}}$. Note that no rays might be exactly in these directions due to the orientation of the robot, in case which we select the closest rays. In following explanations we will call these vectors with $\mathbf{u}^{\mathcal{W}}, \mathbf{d}^{\mathcal{W}}, \mathbf{r}^{\mathcal{W}}$ and $\mathbf{l}^{\mathcal{W}}$ which refer to laser beams closest to the upwards, downwards, rightwards and leftwards directions in the world frame.

We do ray-casting to determine the intersections of the above four sets of vectors ($\mathbf{u}^{\mathcal{W}}, \mathbf{d}^{\mathcal{W}}, \mathbf{r}^{\mathcal{W}}$ and $\mathbf{l}^{\mathcal{W}}$) with the map. We call these as $\mathbf{u}_c^{\mathcal{W}}, \mathbf{d}_c^{\mathcal{W}}, \mathbf{r}_c^{\mathcal{W}}$ and $\mathbf{l}_c^{\mathcal{W}}$. Casting is done against an occupancy grid map with resolution of 5 cm. After ray-casting, we update robot $y^{\mathcal{W}}$, $z^{\mathcal{W}}$ positions such that the discrepancy between the measured rays and the casted rays reduces. A snapshot of this procedure is illustrated in Fig. 7. Also Algorithm 1 explains this method. Due to the convexity of the tunnel cross-section, this algorithm is guaranteed to converge to the correct position.

The on-board attitude estimator supplies roll and pitch data with drift correction; but the yaw needs to be corrected using the laser because the IMU cannot measure the global yaw angle. However, due to the metal interior of the tunnel, we cannot use the magnetometer output as a global reference to the yaw angle. For this reason, we estimate the yaw angle with respect to the tunnel using laser scans.We propose a geometric solution to this problem using the fact that intersection of a cylinder (tunnel) and a plane is always an ellipse. It is easy to see that the intersection of a plane with a cylindrical tunnel can result in three different curves which are circle, ellipse and two parallel lines. This curve is a circle only when $\hat{x}^{\mathcal{W}}$ and $\hat{z}^{\mathcal{B}}$ are aligned, which is very unlikely to happen in our case. Other two cases are more likely to be observed and both can be treated as an ellipse since two parallel lines correspond to the special case of an ellipse with infinite major axis length. So we fit an ellipse to scans and then orientation of the major axis gives negative of the yaw angle up to π radians ambiguity. While we define $\psi = 0$ to be the case when $+\hat{x}^{\mathcal{W}}$ and $+\hat{x}^{\mathcal{B}}$ are coincident, the source of ambiguity is due to the lack of any clues to distinguish whether a scan is taken when robot's heading is $\psi = \psi_0$ or $\psi = \psi_0 + \pi$. In both cases the curve due to the laser has the exact same shape. We choose the yaw measurement that is closest to the current UKF yaw estimate for measurement update.

As seen in Fig. 5, laser data can be noisy due to unmodeled obstacles in the environment, inherent noise in the laser scanner and complete failures. A direct fit to such data is very probable to give wrong estimates which we experienced several times during experiments in development stage and caused crashes. In order to get rid of this problem, we use RANSAC [10] which obviously improves fit quality. Since we do not make fast maneuvers, we make a reasonable assumption that quadrotor is almost in hover state, in other words $\phi \approx 0$ and $\theta \approx 0$. Otherwise resultant ellipse fit would also reflect the effect of non-zero ϕ and θ angles and we would need to decouple these effects to obtain the actual yaw angle. We leave the details of ellipse fitting algorithm to [11].

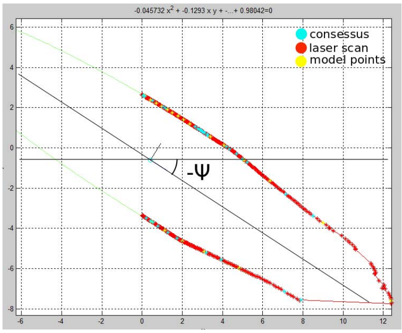


Fig. 5 A sample laser scan data. Ellipse is fit using the method in [11]. In order to eliminate outliers, we use RANSAC. Outliers are due to operators moving together with the quadrotor, noise and laser failures.

3.2 Rao-Blackwellized Particle Filter Design

In this model, we carry the well-known UKF prediction using the IMU output. Measurement updates for positions and velocities in the $y^{\mathcal{W}} - z^{\mathcal{W}}$ directions, as well as the roll, pitch, and yaw orientation are performed within the UKF framework as well.

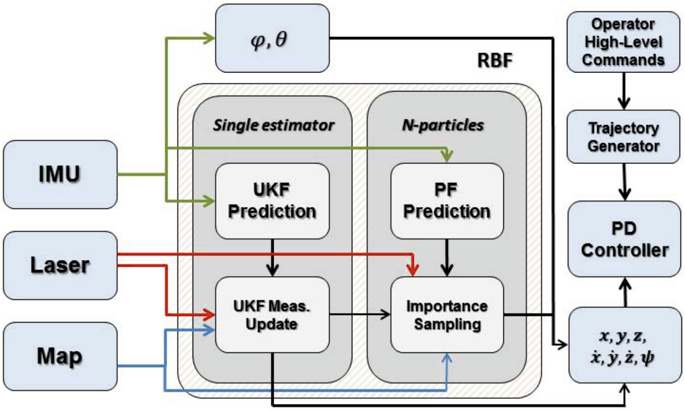


Fig. 6 The estimator based on the Rao Blackwell Filter and the PD controller for autonomous flight in a tunnel of known cross section. A particle filter with N particles is used to model the propagation of state estimates and the uncertainty in the $\hat{x}^{\mathcal{W}}$ direction, while a UKF is used to estimate the remaining states.

A particle filter is used to estimate $x^{\mathcal{W}}$ position of robot (Fig. 6) . That is, during the prediction step of UKF we make use of gyroscope and accelerometer data and in the measurement update stage we integrate information from the measurement models above. The reason using orientation information from the IMU twice which are the gyroscope data (angular velocity) and the on-board roll-pitch estimation, is because of the computational constraints. IMU supplies estimates (roll and pitch) at a rate of 100Hz which we know to be reliable due to the drift correction. But making measurement updates at this rate consumes valuable CPU time. Instead we integrate them at the same rate of laser scanner (30Hz) and carry the low-cost prediction update at 100Hz using the gyroscope data (angular velocity). Note that running a measurement update (UKF update) requires calculation of matrix square root which is of complexity $O(n^3)$. With our current setup, we have chosen not to spend CPU power with frequent measurement updates.

The overall system design is shown in Fig. 6. We run a particle filter for estimating the position and velocity along $\hat{x}^{\mathcal{W}}$ and an UKF common to all particles to estimate the remaining state variables which are $y^{\mathcal{W}}$, $z^{\mathcal{W}}$ and their derivatives and the three Euler angles, ψ, θ, ϕ. The inputs are data from the laser scanner, the IMU and a grid map. Unless we are close to the junction region of the horizontal and inclined portions of the tunnel, we don't have measurements to estimate $x^{\mathcal{W}}$. This implies that in such cases uncertainty along this direction can be in any form which may not be have a closed-form representation. However for all the other states, including lateral and vertical positions and orientation, we always have laser measurements. We expect a unimodal uncertainty model for these states and use the UKF to estimate them.

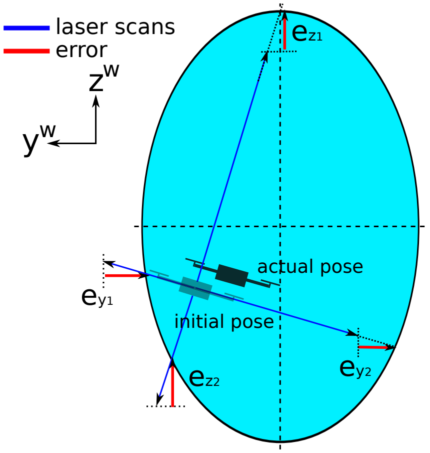


Fig. 7 Starting from an initial pose, ICP iteratively refines $y^{\mathcal{W}} - z^{\mathcal{W}}$ positions to reduce discrepancy between laser data and robot pose. Red vectors are the error vectors to be minimized. Although in the horizontal region of the tunnel cross-section is circular, in inclined region it will be an ellipse as seen by the robot.

When the robot is away from the junction region of the two portions of the penstock, laser scanner cannot make any readings since the closest wall is farther than the maximum range of the laser scanner. This invalidates the measurement model explained for $y^{\mathcal{W}} - z^{\mathcal{W}}$ estimation. Instead we use Algorithm 2 as the measurement model to calculate the weight for each particle. When there are valid measurements, particles consistent with them will be given more importance hence will survive in the importance sampling. Otherwise all particles are given the same weight and importance sampling favors them equally. As we get consecutive measurement failures, distribution of the particles spread out widely according to the IMU noise model. Note the power in representing arbitrary distributions with particles is obviously not achievable with a Gaussian assumption.

In Algorithm 2, to find the weight of a particle, similar to what we do in Algorithm 1, we define a set of vectors, $\mathbf{f}^{\mathcal{W}}$, which are the closest laser beams to $\hat{x}^{\mathcal{W}}$ direction. Then we cast these vectors against the grid map to obtain $\mathbf{f}_c^{\mathcal{W}}$. The weight of a particle is the reciprocal of $|(\mathbf{f}^{\mathcal{W}} - \mathbf{f}_c^{\mathcal{W}})_x|^2$. In case we don't have a valid reading, we assign a non-significant weight.

Depending on the availability of valid laser measurements along the axis of the tunnel, we constrain the regions to resample particles in. In case of valid measurements, resamping is done only in the region close to the junction. Similarly, failure of laser implies robot is away from the junction and particles close to the junction are eliminated.

Algorithm 1. $[h_{yz}, \Sigma_{yz}]$ =measurement_model_yz($laser, map$)

```
$iter \leftarrow 0$
$\mathbf{r}^{\mathcal{W}} \leftarrow$ get_beams_in_dir$(-\hat{y}^{\mathcal{W}}, laser)$ ; $\mathbf{l}^{\mathcal{W}} \leftarrow$ get_beams_in_dir$(+\hat{y}^{\mathcal{W}}, laser)$
$\mathbf{u}^{\mathcal{W}} \leftarrow$ get_beams_in_dir$(+\hat{z}^{\mathcal{W}}, laser)$ ; $\mathbf{d}^{\mathcal{W}} \leftarrow$ get_beams_in_dir$(-\hat{z}^{\mathcal{W}}, laser)$
while $(err_x > thres \wedge err_y > thres) \vee iter < iter_{max}$ do
    $\mathbf{u}_c^{\mathcal{W}} \leftarrow$raycast($\mathbf{u}^{\mathcal{W}}, map$); $\mathbf{d}_c^{\mathcal{W}} \leftarrow$raycast($\mathbf{d}^{\mathcal{W}}, map$)
    $\mathbf{l}_c^{\mathcal{W}} \leftarrow$raycast($\mathbf{l}^{\mathcal{W}}, map$); $\mathbf{r}_c^{\mathcal{W}} \leftarrow$raycast($\mathbf{r}^{\mathcal{W}}, map$)
    $err_y \leftarrow (\mathbf{l}_{c,y}^{\mathcal{W}} - \mathbf{l}_y^{\mathcal{W}}) + (\mathbf{r}_{c,y}^{\mathcal{W}} - \mathbf{r}_y^{\mathcal{W}})$ ; $err_z \leftarrow (\mathbf{u}_{c,z}^{\mathcal{W}} - \mathbf{u}_z^{\mathcal{W}}) + (\mathbf{d}_{c,z}^{\mathcal{W}} - \mathbf{d}_z^{\mathcal{W}})$
    $\mathbf{p}_y^{\mathcal{W}} \leftarrow \mathbf{p}_y^{\mathcal{W}} + 1/2 err_y$ ; $\mathbf{p}_z^{\mathcal{W}} \leftarrow \mathbf{p}_z^{\mathcal{W}} + 1/2 err_z$
    $iter \leftarrow iter + 1$
end while
$h_{yz} = \mathbf{p}_{y,z}$
$\Sigma = Q^T \begin{bmatrix} err_x^2 & 0 \\ 0 & err_y^2 \end{bmatrix} Q$
$Q^T \Sigma Q$ transforms residual errors of ICP to its corresponding covariance matrix [6]
```

Algorithm 2. $[w_x]$ =measurement_model_x($laser, map$)

```
if $laser$ is not valid then
    $w_x \leftarrow 1/\sigma^2$
else
    $\mathbf{f}^{\mathcal{W}} \leftarrow$ get_beams_in_dir$(+\hat{x}^{\mathcal{W}}, laser)$
    $\mathbf{f}_c^{\mathcal{W}} \leftarrow$raycast($\mathbf{f}^{\mathcal{W}}, map$)
    $w_x \leftarrow 1 \,/\, |\mathbf{f}_{c,x}^{\mathcal{W}} - \mathbf{f}_x^{\mathcal{W}}|^2$
end if
```

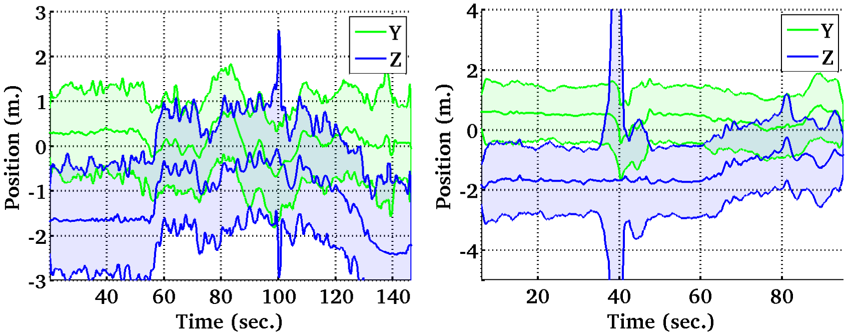
3.3 Control

The errors in localization exhibit anisotropy. They are significant in the position coordinate along the axis of the tunnel but more constrained in the other directions. Accordingly we advocate a semi-autonomous control scheme where the the operator goals (or goals from a planner) prescribe the yaw angle, lateral and vertical positions along the cross section of the tunnel, while the control along the axis of the tunnel is performed by the operator by directly commanding the acceleration through a joystick.

4 Experimental Work

In this section we present and interpret results of our experimental work. Data for the experimental work is collected in three different sites: the Carter Dam and the Allatoona Dam, both in Georgia, and in a long building hallway at the University of Pennsylvania.

In the visit to the Carter Dam, two datasets were collected. In the first flight the quadrotor traversed along the horizontal part of the penstock. And in the second dataset, it flew close to the junction region towards the inclined region. During these tests, the quadrotor was controlled manually. The proposed localization algorithm was run off-line using collected data sets.



(a) Experiment #1 in Allatoona Dam (b) Experiment #2 in Allatoona Dam

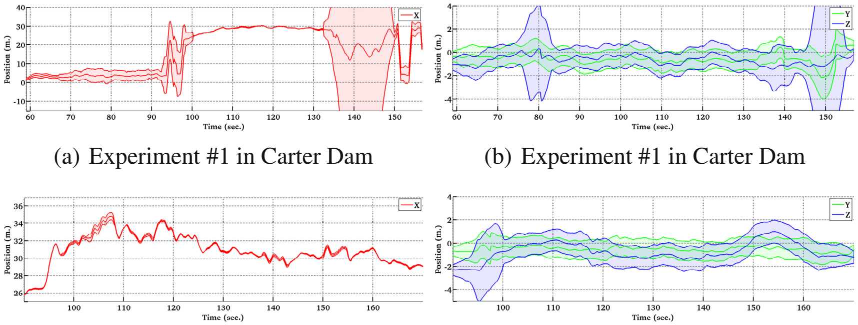
Fig. 8 These figures show estimation outputs for $y^{\mathcal{W}}$-$z^{\mathcal{W}}$ positions together with covariances as shaded regions. In these experiments quadrotor flew semi-autonomously. Due to reflective surfaces, laser scanner failed to return readings along $\hat{x}^{\mathcal{W}}$ direction. So we cannot estimate position along this direction. Failure was due to the distance to the junction region, wet surface and oblique surface w.r.t. ray direction.

Two semi-autonomous flights were conducted in the Allatoona Dam. The operator sets the desired $y^{\mathcal{W}}$, $z^{\mathcal{W}}$, and ψ through a radio controller. Then feedback control of these parameters is carried out by our controller. The operator controls the acceleration along the $\hat{x}^{\mathcal{W}}$ direction. We believe semi-autonomy proves accuracy and stability of our estimator along the tunnel cross section. Otherwise, as opposed to a ground robot, faults in controller or estimator would cause unrecoverable instabilities.

We conducted a third experiment in a building at the University of Pennsylvania, along a 42 meters long corridor while the quadrotor flew semi-autonomously. In the corridor experiment, although there are features, such as pillars and doors, the map we are using is a featureless rectangular prism. So there is no feature in our map that would help in estimating the $x^{\mathcal{W}}$ position. Actually those features behave as noise for yaw estimation which shows robustness of our estimator.

In Fig. 8-9-10 we give results for our Allatoona Dam, Carter Dam and university building experiments respectively. These experiments show quadrotors can be considered as a reasonable choice for inspection of tunnel-like environments. Only with a laser scanner and an IMU, as a requirement for semi-autonomy, localization along the cross-section of the tunnel can be achieved robustly. Also, when one end of the tunnel is in the range of laser scanner, localization along the tunnel axis is achieved as well.

In Fig. 8(b) at 40th seconds, increase in the covariance is due to a worker walking near the quadrotor. However, we can handle such cases and estimated position is not affected. In Fig. 9(a)-10(a), periods when the covariance gets larger is when the robot is away from the end of the tunnel/corridor with the following exceptions. In Fig. 9(a) around 160th seconds increase in uncertainty is due to failure of laser scanner due to water drainage behaving as a mirror. And increase in variance in



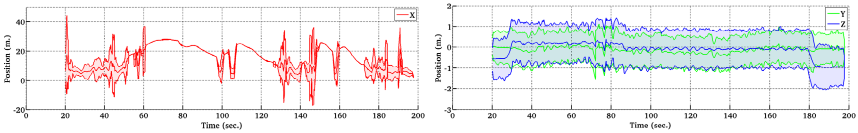
(a) Experiment #1 in Carter Dam

(b) Experiment #1 in Carter Dam

(c) Experiment #2 in Carter Dam

(d) Experiment #2 in Carter Dam

Fig. 9 These figures show estimation outputs for $x^{\mathcal{W}}$-$y^{\mathcal{W}}$-$z^{\mathcal{W}}$ positions together with covariances as shaded regions. Opposed to Allatoona Dam tests (see Fig. 8), since the walls of the penstock was not wet and reflective, we could get readings from the junction region of the tunnel. In Fig. 9(a) we can see that during a period of the flight we are able to localize along $\hat{x}^{\mathcal{W}}$ direction. In the second experiment we flew the quadrotor close to the junction region and have less time periods without valid readings along $\hat{x}^{\mathcal{W}}$ direction. This is shown in Fig. 9(c). High covariance regions in Fig. 9(a) correspond to localization failures.



(a) Experiment in university building

(b) Experiment in university building

Fig. 10 These figures show results for tests carried in a corridor of length 42 meters in a building of University of Pennsylvania. Estimation outputs are given for $x^{\mathcal{W}}$-$y^{\mathcal{W}}$-$z^{\mathcal{W}}$ positions together with covariances as shaded regions. Videos of this experiment can be found at: http://mrsl.grasp.upenn.edu/tolga/FSR2013.mp4

Fig. 10(a) around $100^{th} - 120^{th}$ and 160^{th} seconds is because quadrotor was tilted and laser scanner sees the floor. Since the floor is tiled with marble, it behaves as a mirror and laser scanner fails.

5 Conclusion and Future Work

This work presented results of localization and semi-autonomous control of a quadrotor flying in a dam penstock. We used a Rao-Blackwellized particle filter for localization consisting of a standard particle filter for localization along the tunnel axis and a UKF to represent estimates the other five directions. This way we can represent uncertainty along the tunnel axis, which is quite significant compared to the other directions, using an non parametric distribution. Because of this anisotropy, our experiments required the human operator to specify input (acceleration) along the tunnel axis while the low-level control software provides for regulation and trajectory tracking in the other five directions.

This work is significant because it can replace the tedious and expensive process of manual inspection involving building scaffolds with human inspectors with semi-autonomous quadrotors with cameras. We believe that with some training a modestly skilled operator can fly a quadrotor through a tunnel while inspecting images from onboard cameras for defects along the tunnel walls. While our experiments were performed in penstocks that are used in dams and hydroelectric power plants, the same approach can be used for other tunnels such as those encountered in transportation networks.

Our current work is directed toward addressing more complex (but known) geometries encountered in dams near turbines and to improve the estimation of localization errors along the tunnel axis using onboard illumination sources and visual odometry algorithms.

Acknowledgements. We gratefully acknowledge the support of ONR Grants N00014-07-1-0829 ARL Grant W911NF-08-2-0004, NSF grants 0742304, 1138110, and 1113830, and collaborations with Jennifer Wozencraft and Thomas Hood of the US Army Corps of Engineers.

References

1. Ascending Technologies GmbH
2. Hokuyo Automatic Co. Ltd.
3. Agrawal, M., Konolige, K.: Real-time localization in outdoor environments using stereo vision and inexpensive GPS. In: 18th International Conference on Pattern Recognition, ICPR 2006, vol. 3, pp. 1063–1068 (2006)
4. Burgard, W., Derr, A., Fox, D., Cremers, A.B.: Integrating global position estimation and position tracking for mobile robots: the dynamic Markov localization approach. In: Proceedings of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2, pp. 730–735 (1998)

5. Burgard, W., Cremers, A.B., Fox, D., Hähnel, D., Lakemeyer, G., Schulz, D., Steiner, W., Thrun, S.: Experiences with an interactive museum tour-guide robot. Artificial Intelligence 114(1-2), 3–55 (1999)
6. Censi, A.: An accurate closed-form estimate of ICP's covariance. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Rome, Italy, pp. 3167–3172 (April 2007)
7. Dellaert, F., Fox, D., Burgard, W., Thrun, S.: Monte Carlo localization for mobile robots. In: Proceedings of the 1999 IEEE International Conference on Robotics and Automation, vol. 2, pp. 1322–1328 (1999)
8. Doucet, A., de Freitas, N., Murphy, K.P., Russell, S.J.: Rao-Blackwellised particle filtering for dynamic bayesian networks. In: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, pp. 176–183 (2000)
9. Durrant-Whyte, H., Bailey, T.: Simultaneous localization and mapping: part i. IEEE Robotics Automation Magazine 13(2), 99–110 (2006)
10. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24(6), 381–395 (1981)
11. Fitzgibbon, A., Pilu, M., Fisher, R.B.: Direct least square fitting of ellipses. IEEE Transactions on Pattern Analysis and Machine Intelligence 21(5), 476–480 (1999)
12. Grisetti, G., Stachniss, C., Burgard, W.: Improved techniques for grid mapping with Rao-Blackwellized particle filters. IEEE Transactions on Robotics 23(1), 34–46 (2007)
13. Grzonka, S., Grisetti, G., Burgard, W.: Towards a navigation system for autonomous indoor flying. In: IEEE International Conference on Robotics and Automation, ICRA 2009, pp. 2878–2883 (2009)
14. Hansen, P., Corke, P., Boles, W.: Wide-angle visual feature matching for outdoor localization. Int. J. Rob. Res. 29(2-3), 267–297 (2010)
15. Hentschel, M., Wulf, O., Wagner, B.: A GPS and laser-based localization for urban and non-urban outdoor environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2008, pp. 149–154 (2008)
16. Liu, T., Carlberg, M., Chen, G., Chen, J., Kua, J., Zakhor, A.: Indoor localization and visualization using a human-operated backpack system. In: 2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN), pp. 1–10 (September 2010)
17. Mahony, R., Kumar, V., Corke, P.: Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. IEEE Robotics Automation Magazine PP(99), 1 (2012)
18. Michael, N., Mellinger, D., Lindsey, Q., Kumar, V.: The GRASP multiple micro-uav testbed. IEEE Robotics Automation Magazine 17(3), 56–65 (2010)
19. Se, S., Lowe, D.G., Little, J.J.: Vision-based global localization and mapping for mobile robots. IEEE Transactions on Robotics 21(3), 364–375 (2005)
20. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press (2005)
21. Thrun, S., Fox, D., Burgard, W., Dellaert, F.: Robust Monte Carlo localization for mobile robots. Artificial Intelligence 128(1-2), 99–141 (2001)

Autonomous Aerial Water Sampling

John-Paul Ore, Sebastian Elbaum, Amy Burgin,
Baoliang Zhao, and Carrick Detweiler

Abstract. Obtaining spatially separated, high-frequency water samples from rivers and lakes is critical to enhance our understanding and effective management of fresh water resources. In this paper we present an aerial water sampler and verify the system in field experiments. The aerial water sampler has the potential to vastly increase the speed and range at which scientists obtain water samples while reducing cost and effort. The water sampling system includes: 1) a mechanism to capture three 20 *ml* samples per mission; 2) sensors and algorithms for safe navigation and altitude approximation over water; and 3) software components that integrate and analyze sensor data, control the vehicle, and drive the sampling mechanism. In this paper we validate the system in the lab, characterize key sensors, and present results of outdoor experiments. We compare water samples from local lakes obtained by our system to samples obtained by traditional sampling techniques. We find that most water properties are consistent between the two techniques. These experiments show that despite the challenges associated with flying precisely over water, it is possible to quickly obtain water samples with an Unmanned Aerial Vehicle (UAV).

1 Introduction

Water quality varies due to the spatial distribution of water transport pathways and contaminant source areas. Characterizing this large-scale variability remains a crit-

John-Paul Ore · Sebastian Elbaum · Carrick Detweiler
Computer Science and Engineering, University of Nebraska, Lincoln, Nebraska, USA
e-mail: {jore,elbaum,carrick}@cse.unl.edu

Amy Burgin
School of Natural Resources, University of Nebraska, Lincoln, Nebraska, USA
e-mail: aburgin2@unl.edu

Baoliang Zhao
Mechanical and Materials Engineering, University of Nebraska, Lincoln, Nebraska, USA
e-mail: bzhao@huskers.unl.edu

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_10, © Springer International Publishing Switzerland 2015

ical bottleneck that inhibits understanding of transport processes and the development of effective management plans to address water quality issues. In the US, it is estimated that human-induced degradation of fresh water sources annually costs over $2.2 billion, but the full extent of the cost is poorly known due to insufficient data [1]. World-wide, water borne diseases cause the death of 1.5 million under-five children every year [2].

Current water sampling techniques are often based on grab sampling (e.g. dipping a bottle off the side of a kayak) [3], statically deployed collection systems [4], or using mobile sensors affixed to Autonomous Surface Vehicles (ASVs) [5] and Autonomous Underwater Vehicles (AUVs) [6]. Most autonomous systems are used on large, open water features such as seas, large lakes and rivers, and sample for long duration, in deep or distant places, with high quality. All of these methods are relatively slow, spatially restricted, costly, or difficult to deploy; none sample quickly at multiple locations while overcoming barriers, such as dams or land.

In this paper, we tackle these limitations through the development of a UAV-based water sampling system with a focus on enabling *safe and reliable* in-the-field water sampling. Fig. 1 shows the system collecting a water sample. We designed the system based on input from our limnologist collaborators who specified that the system be carried and deployed by a single person, collect multiple samples within kilometer ranges, and acquire at least 20 *ml* per sample[1].

Obtaining water samples from a UAV, however, poses challenges that must be addressed before these systems can be deployed in the wild. The contributions of this work include: 1) developing a UAV-based system that autonomously obtains three 20 *ml* water samples per flight; 2) integrating and characterizing sensors on the UAV to enable reliable, low-altitude flight (1.0 *m*) over water; 3) testing the system both indoors in a motion-capture room as well as in the field at lakes and waterways; and 4) validating that key water chemical properties are not biased by using a UAV-based mechanism. We also identify a number of outstanding challenges to be addressed in future work, such as determining the impact of waves, winds, and flowing water on altitude control.

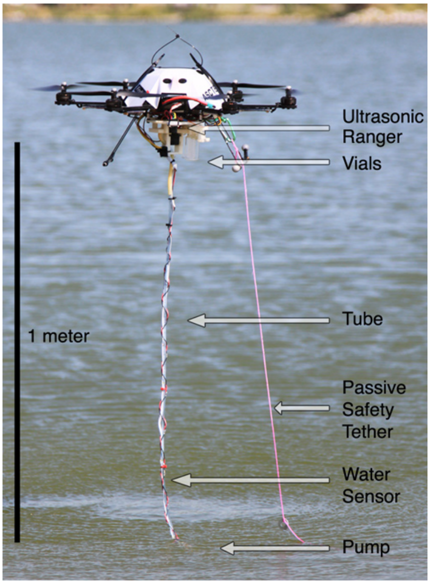


Fig. 1 UAV-Based Water Sampling

[1] The quantity, 20 *ml*, is enough to perform most standard water chemistry experiments.

2 Related Work

Existing efforts relate to this work in one of two ways: either an autonomous vehicle is used to take samples in aquatic environments or a UAV is controlled at low-altitude. We treat first the former and then the latter.

Autonomous vehicles used in water sampling are either Autonomous Surface Vehicles (ASVs) or Autonomous Underwater Vehicles (AUVs), both deployed in water features such as oceans or large lakes. For example, Dunbabin *et al.*'s [5] Lake Wivenhoe ASV is capable of navigating throughout complex inland waterways and measuring a range of water quality properties and greenhouse gas emissions. Underwater, Cruz *et al.*'s [6] [7] MARES AUV dives up to 100 *m* deep to monitor pollution, collect data, capture video, or follow the seabed. Other efforts such as Rahimi *et al.*'s [8] NIMS system explore semi-mobile sensor networks providing adaptive sampling. These vehicles and systems are good for long-duration sampling in deep or distant places. However, it is time-consuming and expensive to frequently re-deploy these systems. In contrast, our system can be carried in a backpack and quickly deployed to sample multiple disconnected water features from a single launch site. Further, *in situ* sampling cannot yet measure all desired water properties, identified by Erickson *et al.* [4], such as the presence of suspended solids, pathogens, and heavy metals.

Other UAV control systems related to our efforts include Merz *et al.* [9], who show techniques for low-altitude flight in rural areas, whereas our focus is low-altitude flight over water but does not include obstacle avoidance. Their system states, like ours, contain events indicating an unsafe circumstance, and transition to a state seeking safe recovery.

Other recent efforts for UAV height estimate include miniature radar altimeters and optical flow altitude estimation as summarized by Kendoul [10]. The lightest commercially available radar altimeters are still 375 *g*, heavy for a micro UAV, and are accurate to only ± 0.5 *m*, below the requirements of our system. Optical methods are easily perturbed by ambient light, so instead we chose ultrasonic rangers.

Our system flies with a small dangling pump. Although Sreenath *et al.* [11] explore the flight dynamics of cable-suspended loads, our system avoids this by hanging a small mass, which incurs small forces relative to those generated by our UAV.

The Aquacopter [12] UAV lands in and takes off from calm water. We do not adopt this platform or land in the water because: 1) fast-moving water or waves might make it impossible to take off; 2) the sampling mechanism and battery enclosure would be complete sealed, making removal difficult and decreasing the efficiency of swapping vials or batteries; and 3) radio strength attenuates near the water's surface and we want the UAV and base station in constant contact.

Our work most resembles the low-altitude UAV presented by Göktoğan *et al.* [13], wherein the authors surveil and spray aquatic weeds at low altitude using a RUAV ("rotary UAV"). This RUAV measures altitude with a laser altimeter, and like our system, requires a human backup pilot. Our work similarly does not address global planning and requires a human expert to decide where to perform tasks (weed experts in Göktoğan's case and lake experts in ours). Our work differs from this in that we use ultrasonic with pressure for altitude, since laser altimeters work poorly at short range over clear water, and we retrieve a liquid rather than depositing it. In addition, we focus on validating the utility of the system for water scientists.

3 Applications in Environmental Monitoring

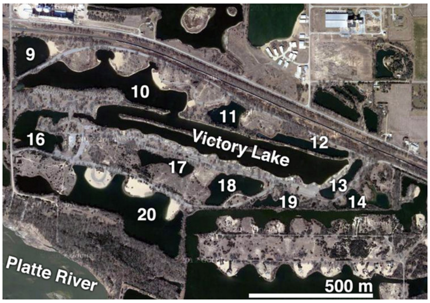


Fig. 2 Sandpit Lakes - Fremont, Nebraska, USA

Presently, limnologists and hydrochemists require water samples for lab analysis. They measure chemical properties of surface water, including phosphate, total phosphorus, nitrate/nitrite, nitrogen, and ammonia, as well as biological properties, such as the presence of toxic microcystins. Other useful properties can be measured *in situ*, but require a literal boatload of equipment, used to measure temperature, conductivity, pH, dissolved oxygen, light, turbidity, and Secchi transparency. All of these field measurements, along with lab analysis, together present much of the canonical data through which surface water phenomenon are understood [14]. By facilitating data collection, lightweight UAVs, together with our collaborators, will improve, if not “revolutionize” spatial ecology [15]. We see applications of UAV-based water sampling in two areas: 1) increasing the ease of capturing routine small samples from disconnected water features; and 2) improving the quality of event-based datasets by increasing spatial and temporal resolution.

For example, our collaborators study the Fremont Sandpit lakes (see Fig. 2). Each numbered lake is groundwater connected, surface water disconnected, chemically distinct, and must be sampled separately. Currently, a team of three scientists tow a boat to the lake, launch the boat, navigate to the sample location, collect samples and take measurements, return to dock, get the truck, put the boat back on the trailer, and drive to the next lake. Each of 10-15 lakes are sampled in this manner over a long 10-15 hour day. But in just two hours, one scientist with our UAV-system could sample all these lakes, enabling the possibility of capturing data with unprecedented spatiotemporal resolution.

4 Technical Approach

Through discussions with our hydrologist partners we derived a set of requirements for the aerial water sampler. First, it must capture at least three 20 *ml* water samples at predefined locations within 1 *km*. Second, it must be light and small enough to be carried by a single scientist, and sample autonomously once target locations are identified. Third, it must be reliable and safe to reduce cost and risk, since these are the primary barriers for adoption. Fourth, the new sampling system must not influence water properties. Additional requirements not addressed in the current work include a simple user interface for scientists to use and endurance and robustness to work in any climate. We chose to address first the core functionality of the system and save secondary requirements for future work.

We now describe how we address these requirements through: 1) mechanical design, including the UAV and sampling mechanism; 2) sensors for near water flight; and 3) the software system, including a discussion of the safety logic used to ensure the vehicle stays out of the water.

4.1 Design of UAV Water Sampling Mechanism

The water sampler is built onto an Ascending Technologies Firefly [16], a hexrotor with a maximum payload of 600 *g*. Total flight time is 15-20 minutes. The Firefly comes equipped with GPS, 3-axis accelerometers and gyroscopes, compass and an air pressure sensor. This UAV communicates with a human backup pilot using a radio link, and has two 2.4 *GHz* 802.15.4 radios for remote autonomous control and sensor feedback. To comply with local regulations regarding UAVs, we fly outdoors with a passive string tether connected to the frame of the vehicle and wrangled by a human operator. In practice, the tether limits the distance the UAV can travel but does not otherwise impact its mobility.

The water sampling mechanism consists of three spring-lidded chambers. The chambers are constructed so that a servo-rotated 'needle' lifts the lid and directs the water flow into one of three 20 *ml* glass vials (Fig. 3). Once the needle rotates away from the vial, it seals closed. The servo can also select an intermediate position to enable flushing of the needle and tubing between samples (Fig. 3). The duration of the flushing phase is configurable, defaulting to 20 *s*, three times the duration required to fill a 20 *ml* vial[2]. The needle is connected to a 1.05 *m* plastic tube hanging below the UAV with a micro submersible water pump [17] attached at the end of the tube. The tube is mounted below the center of mass of the unloaded vehicle, to minimize changes in flight dynamics while pumping. A break-away mechanism allows the pump and tube mechanism to release if subjected to a sufficient force, as might happen if the pump becomes entangled in the environment, and the UAV thrusts away from it.

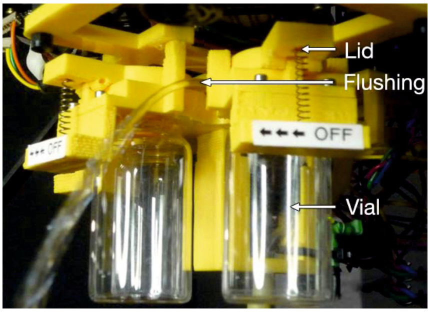


Fig. 3 Flushing the sample system

[2] Initial experiments show that 20 *s* flushing avoids cross-contamination. We plan to more rigorously characterize this in future work.

4.2 Sensors for Near Water Flight

The UAV includes an onboard pressure sensor. To improve height estimation, we augment the UAV with ultrasonic rangers and water conductivity sensors. We use two Maxbotix MB1240-EZ4 ultrasonic rangers [18] pointing straight down and flanking the sampling mechanism 10 *cm* from the center to increase the likelihood of an unobstructed path to the water's surface, which might otherwise be blocked by the swinging tube and pump.

Each ultrasonic ranger samples at 10 *hz* and we offset their sample time by 50 *ms* to prevent interference. This also increases the rate that altitude information is acquired to 20 *hz*. This rangefinder is well-suited to rotorcraft because of its resilience to motor noise, ±1 *cm* accuracy, and reliability below 3 *m*.

Water conductivity sensors are placed every 10 *cm* from the bottom of the sample tube, up to 50 *cm*, to ensure that the system knows when its too close to the water and also to regulate the pump. The pump must be submerged and primed prior to operation. An onboard controller turns on the pump only after being wet for more than 400 *ms* which allows it, as experimentally determined, to prime.

4.3 Software

The software system contains two sub-systems: 1) code on a control computer using the Robot Operating System [19] which handles low-level communication with the UAV, mission control, navigation, and high-level sampling tasks; and 2) onboard code on a custom built microcontroller mounted on the UAV that manages the 'needle' servo, regulates the water pump, reads ultrasonic and water sensor data, and broadcasts the water-sampling sub-system's state. Both sub-systems incorporate predicates to detect unsafe water sampling or navigating conditions based on the sensor readings, and restart a mission. In total, the system includes about 7K lines of C, C++, and Python code.

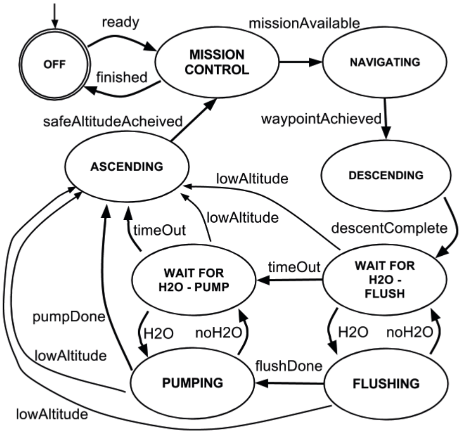


Fig. 4 Sampling States

The flow of water sampling activities is shown in Fig. 4, and follows a clockwise pattern. Overall, the system receives a mission, navigates to a sample location, descends near to the water surface, waits for the water sensors to confirm that the pump is wet, flushes, pumps, ascends, and navigates either to the next sample location or returns to the landing location. The software coordinates these activities through: 1) waypoints, which are compared to the measured location of the UAV, so that the UAV arrives at the desired sample location and descends to the target height; 2) timers, which track how long the pump has actually been pumping and infer that the tube has been sufficiently flushed or that the vial is full; and 3) safety predicates on sensor values which ensure the sampling altitude is safe. If the safety constraints are violated, the UAV retreats to a safe altitude and the mission continues.

5 Altitude Estimation Over Water

We form an altitude estimation in two ways: 1) at low altitude with a Kalman Filter of ultrasonic ranger and pressure sensor readings; and 2) at high altitudes with the pressure sensor plus an offset from the low-altitude Kalman estimate. In this section we characterize the ultrasonic sensors over water, discuss how the low altitude estimation is formed and then how the low and high altitude estimations are used to form a final altitude estimate.

The ultrasonic rangers are necessary because the pressure sensor alone drifts over time due to wind or changes in atmospheric pressure. We characterized the ultrasonic rangefinders over water by conducting indoor flight tests with ground truth from a Vicon motion capture system [20]. We tested their performance while flying over water. The results are show in Fig. 6, during which the UAV was over water, and the ultrasonic readings are shown offset by 15 *cm*, the height of the water in the fishtank. The data was gathered during autonomous flight, flying the UAV to 2 *m* above the fish tank, then descending to 1.5 *m* and 1.25 *m* before returning to 2 *m* and leaving the over water area. We placed acoustic foam over the fish tank (Fig. 5) to absorb the ultrasound readings so that the edge of the tank is not detected.

As seen in Fig. 6, the ultrasonics closely follow Vicon ground truth, although they lag slightly behind as the UAV descends. The lag is caused by the latency of the ultrasonics, but the lag is less important for our system since we're most concerned

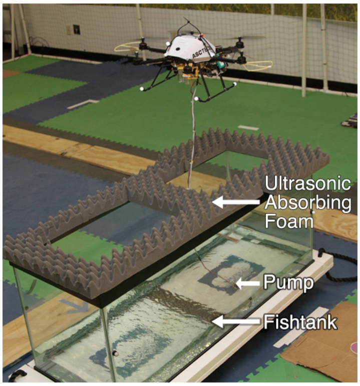


Fig. 5 Indoor Testbed for Water Sampling

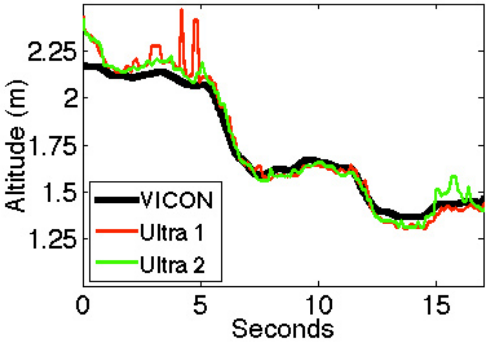


Fig. 6 Ultrasonic and Vicon Altitude Over water

with accurate readings when the UAV is hovering and since we limit the descent velocity so that the system has more time to detect the water's surface. In extreme cases, the ultrasonics exhibit large spikes at longer ranges ($+2\ m$), but this noise is usually brief and rarely affects both sensors simultaneously, so having more than one sensor is important to filter sporadic noisy readings. These experiments show that the ultrasonic sensors perform well over water on a flying UAV, especially when the UAV is hovering near the water's surface.

5.1 Kalman Filter Low-altitude Estimation

At low altitude, we merge the pressure and ultrasonic readings using a Kalman Filter and shown in Fig. 7. The ultrasonics must be pre-filtered before entering the Kalman Filter since the swinging tube causes non-Gaussian noise. The current readings from the two ultrasonic sensors are evaluated based on variance during the last second and proximity to the current Kalman estimate. We choose the reading with least variance, closest to the current Kalman estimate, giving preference to proximity. If both or neither satisfy these conditions, we average them. While its rare to have faulty readings from both sensors, experimentally we have determined that even if there is continuous faulty data from the ultrasonics, the Kalman estimate quickly converges to a good estimate once a single sensor yields accurate readings.

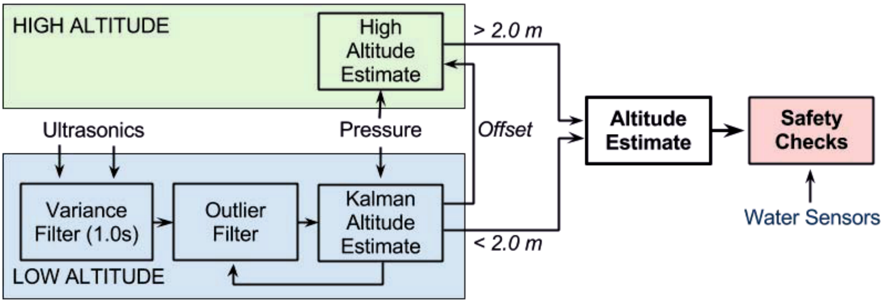


Fig. 7 Altitude Estimation Information Flow

5.2 Final Altitude Estimate

The final altitude estimate uses the Kalman estimate at low altitude and the pressure sensor with an offset at high altitude as shown in Fig. 7. At low altitudes, the Kalman estimate is accurate enough to assure vehicle safety, while at high altitude, the pressure sensor is sufficient and if sensor drift forces the system below two meters, the low-altitude controller will take over. Anytime the vehicle transitions from low to high altitude, the pressure sensor is offset with the last best estimate from the Kalman Filter. When descending, we limit velocity so that the UAV can stop before coming within one meter of the water.

We enforce additional safety checks with the water sensors on the tube. If the water sensors indicate that the tube is too deep, then the UAV ascends to a safer altitude. The water sensor data is not directly added to the Kalman Filter both because they are slow (0.5 s) and also because occasional water droplets from the pump cause false readings. In the next section we validate this approach with indoor and field experiments.

6 Altitude Experiments while Sampling

We performed experiments indoors and outdoors to validate the altitude estimate while sampling. The indoor experiments verified that the Kalman filter-based altitude estimate closely tracked the Vicon ground truth. Outdoors, the location was a human-made waterway along Antelope Creek in Lincoln, Nebraska, USA. The water at this location is $1-2$ m deep. For these outdoor tests we chose a calm day with wind speeds measured at less than 0.27 ms^{-1} with a hand-held anemometer. Fig. 1 depicts the system operating outdoors.

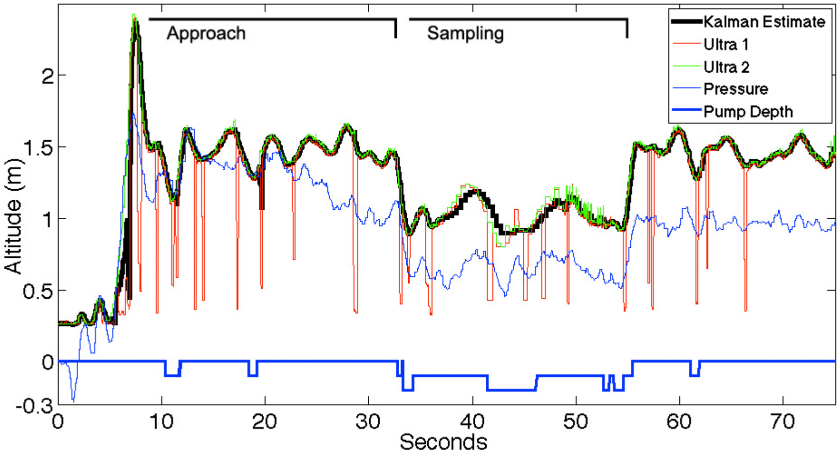


Fig. 8 Vehicle Altitude and Pump Depth While Sampling Outdoors

We recorded the ultrasonic, pressure sensor, and Kalman-filtered height estimate, as shown in Fig. 8. During this experiment the UAV always flew at low altitude. This figure shows the UAV while it ‘approaches’ the sample destination and the critical ‘sample’ stage when the UAV descends and maintains altitude to pump water. Compared with altitude tests indoors, the ultrasonic sensor readings had more spikes, indicating additional noise[3], but the dual ultrasonics still allowed for successful altitude control. The figure also shows the depth of the pump, as detected by the water

[3] The noise from Ultrasonic 1 in Fig. 8 is an extreme example, as there was faulty cabling. However, the altitude estimate tracks in spite of this noise.

sensors on the tube. Both the first and second water sensor are activated during sampling, but never the ones above. We noticed that the water sensor skimmed the surface as the UAV approached the sample location, which is reflected in Fig. 8. During the outdoor altitude tests, we observed a larger variation in x and y during sampling due to GPS inaccuracy, which impacts height as the UAV tilts as it tries to adjust its location. These tests confirm that our filtered altitude estimate works well at near proximity to water in calm conditions. Future tests will stress the system with stronger winds and larger waves.

7 Water Sampler Effectiveness Experiments

We tested the water sampling system both indoors and outdoors. Indoors, we perform autonomous missions that launch the UAV to 2.0 m, fly over the fish tank (Fig. 5), descend to the sampling height where the pump is submerged, take a sample, and then ascend back to 2.0 m. Each test consisted of three samples, and afterward the water sample vials were checked. Any amount less than the top of the 'neck' of the sample vial was recorded as less than full. We completed a total of 30 trials. Each trial took 4-5 minutes flying, with an additional 5-10 minutes to set up the system, empty the vials, and periodically change batteries.

Table 1 summarizes the results. Overall, from the 90 consecutive collected samples indoors (30 trials with 3 samples each), 81 were full (90% success). To better understand the relation between the success rate and the use of our ultrasound and pressure altitude controller, half of the samples were collected using the altitude reported by the Vicon motion capture system. The first and second rows of Table 1 show that the success rate is nearly the same for both Vicon and ultrasonic altitude, which indicates that ultrasonic rangers are suitable for height estimation over water.

Of the indoor sample failures, six of nine were over half-full. Failures were caused by the pump landing outside the fishtank or the pump failing to self-prime.

Likewise, we performed outdoor experiments to test the effectiveness of the sampling system when controlled autonomously over water. We programmed the system to navigate to GPS waypoints and obtain three samples. The results of this test are shown in Table 1. The success rate for fully-filled vials was 69%, with 7 of 12 failures caused by a faulty lid mechanism which we have now fixed. Three of the

Table 1 Sampling Success Rate

Altitude	Trials	Samples	Full	$> \frac{1}{2}$	$< \frac{1}{2}$	% Full
Vicon	15	45	41	3	1	91.1
Ultrasonic	15	45	40	3	2	88.9
Total Indoor	30	90	81	6	3	90.0
Outdoor	13	39	27	4	8	69.2
Grand Total	43	129	108	10	11	83.7

remaining five "failures to fill" occurred on the third vial when the backup pilot took over control after perceiving that the UAV was trending too close to water, especially as the wind increased during the experiment. We believe pilot aborts will occur less frequently in the future as we improve hover stability in gusty conditions and as safety pilot confidence increases. Thirteen total sample trials were conducted, until all available batteries were discharged. Overall, within the wind and environmental constraints, the system demonstrated the ability to maintain altitude and retrieve samples.

8 Sampling Technique Comparison: Hand vs. UAV-Mechanism

We conducted an experiment to ensure that water samples collected by the UAV-mechanism exhibit similar water chemical properties as samples obtained through traditional hand sampling methods. Potential differences include those caused by pumping, transit through the tube, agitation during flight, and changes in water properties during the delay between sample acquisition and sample measurement on land. The UAV was not flown, but rather held by a human operator in a kayak to ensure that both the hand and UAV samples were taken at the same time and place.

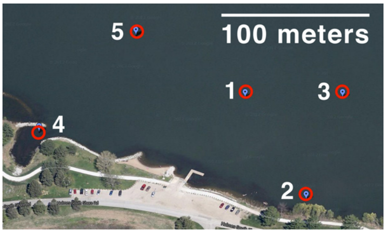


Fig. 9 Holmes Lake

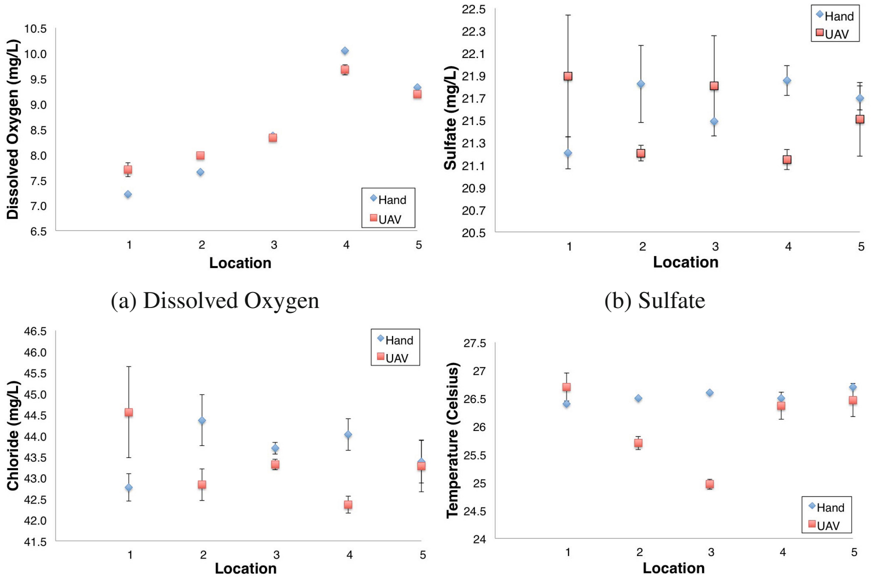
In order to verify the consistency between manual and UAV-based sampling, we sampled at five locations on Holmes Lake, Lincoln, NE, USA. We collected two samples near shore and three closer to the middle of the lake, as shown in Fig. 9. At each location, we took three samples by hand and three with the UAV-mechanism for a total of fifteen samples by each method. Overall it took approximately 2 hours to collect this data due to the time to kayak, collect manual and UAV-mechanism samples, and to perform some on-site analysis and filtering. We estimate that collecting the samples with the UAV flying would take 20 minutes.

At each location we measured temperature, dissolved oxygen (DO)[4], sulfate, and chloride. By sampling both a dissolved gas and representative ions we can assess the suitability of the UAV-mechanism for scientific water sampling. Temperature and DO are measured at the sample location for the manual measurements and at shore once the UAV returns, since these properties change rapidly. Chloride and sulfate ions are measured in the lab using equipment[5] which is not easily portable and these properties don't change rapidly after sampling and filtering. We measured DO as it is a key indicator of biological activity and because we suspected the UAV-mechanism might bias the measurement through degassing during pumping

[4] For DO and temperature a single reading was obtained with the hand sensor at the location, but for the UAV-mechanism it was tested on each of the three samples.

[5] Lab measurements use a Dionex Ion Chromatograph AS14A, made by ThermoFisher

or continued photosynthesis during transit. Sulfate and chloride ions occur naturally in most water and their ratio in freshwater can indicate proximity to a saltwater source. But inland, chloride comes from many sources including lawn fertilizers and road salt. High concentrations of chloride in organisms can induce osmotic stress, reduced fitness, or mortality.



(a) Dissolved Oxygen

(b) Sulfate

(c) Chloride

(d) Temperature

Fig. 10 Water Chemistry measurements from Hand Sampling and UAV-mechanism. Points represent the average of three replicate measurements, and error bars indicate ±1 standard error of the mean.

We are primarily interested in verifying that the UAV-mechanism does not induce a bias in the measurements. Fig. 10a shows the DO as measured by hand at the location and with the UAV-mechanism. The values at the five sample locations are close and show the same general trend in all five locations, implying that the UAV-mechanism and delay (longer by kayak than by flying) has little impact on the DO. Also visible in this figure is the general upward trend between the sample locations. This was probably caused by increased photosynthesis over the two hours of data collection, although sample location may also play a role in this variation. For instance, location 4 is probably higher than the general trend because it is closer to an enclosed bay and therefore likely to have more plants near the surface. Obtaining samples quickly by UAV could help to disambiguate these factors.

Sulfate and chloride concentrations shown in Fig. 10b-10c revealed some differences between hand methods and the UAV-mechanism. These differences, however, can likely be attributed to typical sampling variation and neither indicates a strong bias induced by the UAV-mechanism. Further, the typical range for sulfate in lakes is between $10-60\ mg/L$ [21] and for chloride varies seasonally but usually is between $10-100\ mg/L$ [22], so the observed variation is minimal. We plan to perform additional field and lab tests to verify that these measurements are unbiased.

In contrast to the other measurements, Fig. 10d, shows that the temperature measured by hand at the sample location is nearly constant, while the temperature measured in samples from the UAV-mechanism changed during transit, especially at locations two and three. Future versions of system should measure water temperature at the sample location by mounting a temperature probe at the end of the pumping tube.

These experiments show the UAV-mechanism can collect samples that replace those collected be hand. The UAV system greatly reduces the effort and time to collect samples. This permits water scientists to obtain more samples within a single lake or river to develop a high-resolution map, for instance, after a rainstorm to identify the source of the influx of chemical or biological contaminates. In addition, reducing the collection time is critical since many water properties, such as DO, fluctuate within hours and using our UAV system would reduce collection time by nearly an order of magnitude.

9 Conclusions and Future Work

Water sampling has become a key activity in effectively managing our fresh water resources and maintaining public health. Developing approaches and systems for efficient and effective water monitoring will increase in importance over the coming decades. In this paper, we have demonstrated a novel mechanism for sampling water autonomously from a UAV that requires significantly less effort than existing techniques and is nearly an order of magnitude faster. The system can safely fly close to water and collect three 20 *ml* samples per flight. We verified that the water properties of the samples collected by the UAV match those collected through traditional manual sampling techniques. This shows that this system can be used by water scientists to improve the spatiotemporal resolution of water sampling.

Our future efforts include further operation and evolution of the system outdoors, especially in the presence of varying wind speeds and wave sizes, as well as with moving water. We are in the process of implementing and evaluating the usability of a user interface for the limnologists and non-expert operators that balances manual control with autonomous behavior with the goal of maintaining system and operator safety. We also intend to explore how this platform might be used with adaptive sampling, and in combination with other sensing and sampling mechanisms deployed in bodies of water. We plan to examine the duration of the ‘flushing’ phase with our collaborators to ensure clean samples. Further, we would like to push some water analysis onto the platform to avoid collecting samples that do not meet required criteria. In addition, we will explore a line of inquiry pertaining to operational safety, as these systems are intended to be reliable tools in the hands of field scientists.

Finally, we are pursing approval from the US Federal Aviation Administration to conduct larger-scale outdoor tests at critical test sites identified by water scientists.

Acknowledgements. We would like to thank our other limnologist and environmental engineering partners Dr. Michael Hamilton and Dr. Sally Thompson, for their continuous support of these efforts. We would also like to acknowledge the valued assistance of Hengle Jiang, Christa Webber, Emily Waring, Dr. Seth McNeil, and the NIMBUS Lab. This work was partially supported by USDA #2013-67021-20947, AFOSR #FA9550-10-1-0406, NSF IIS-1116221, NSF CSR-1217400, NDEQ grant #56-1131 and a development grant from ORED-UNL. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of these agencies.

References

1. Dodds, W.K., Bouska, W.W., Eitzmann, J.L., Pilger, T.J., Pitts, K.L., Riley, A.J., Schloesser, J.T., Thornbrugh, D.J.: Eutrophication of U.S. freshwaters: Analysis of potential economic damages. Environmental Science & Technology 43(1), 12–19 (2009)
2. Johansson, E., Wardlaw, T.: Diarrhoea: Why children are still dying and what can be done. In: UNICEF/WHO Library Cataloging-in-Publication Data (January 2009)
3. Wilde, F.D., Radtke, D.B., Geological Survey (US): National Field Manual for the Collection of Water-quality Data: Field Measurements. US Department of the Interior, US Geological Survey (1998)
4. Erickson, A.J., Weiss, P.T., Gulliver, J.S.: Water sampling methods. In: Optimizing Stormwater Treatment Practices, pp. 163–192. Springer, New York (2013)
5. Dunbabin, M., Grinham, A., Udy, J.: An autonomous surface vehicle for water quality monitoring. In: Proc. Australasian Conference on Robotics and Automation (ACRA), vol. 13 (December 2009)
6. Cruz, N.A., Matos, A.C.: The MARES AUV, a modular autonomous robot for environment sampling. In: OCEANS 2008, pp. 1–6. IEEE (2008)
7. Melo, J., Matos, A.: Bottom estimation and following with the MARES AUV. In: Oceans 2012, pp. 1–8. IEEE (2012)
8. Rahimi, M., Pon, R., Kaiser, W., Sukhatme, G., Estrin, D., Srivastava, M.: Adaptive sampling for environmental robotics. In: Proc. IEEE Int. Conf. on Robotics and Automation, vol. 4, pp. 3537–3544 (2004)
9. Merz, T., Kendoul, F.: Dependable low-altitude obstacle avoidance for robotic helicopters operating in rural areas. Journal of Field Robotics 30(3), 439–471 (2013)
10. Kendoul, F.: Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. Journal of Field Robotics 29(2), 315–378 (2012)
11. Sreenath, K., Michael, N., Kumar, V.: Trajectory generation and control of a quadrotor with a cable-suspended load: A differentially-flat hybrid system. In: Proc. IEEE Int. Conf. on Robotics and Automation, pp. 4888–4895 (2013)
12. Aquacopters, `http://www.aquacopters.com`
13. Göktoğan, A.H., Sukkarieh, S., Bryson, M., Randle, J., Lupton, T., Hung, C.: A rotary-wing unmanned air vehicle for aquatic weed surveillance and management. Journal of Intelligent and Robotic Systems 57, 467–484 (2010)
14. Eaton, A.D., Franson, M.A.H.: Standard methods for the examination of water & wastewater. American Public Health Association (2005)

15. Anderson, K., Gaston, K.J.: Lightweight unmanned aerial vehicles will revolutionize spatial ecology. Frontiers in Ecology and the Environment 11(3), 138–146 (2013)
16. Ascending Technologies, `http://www.asctec.de`
17. TCS micropumps, UK. Model M200S-SUB,
`http://micropumps.co.uk/TCSM200range.htm`
18. Maxbotix, `http://www.maxbotix.com`
19. Robot Operating System, `http://www.ros.org`
20. VICON, `http://www.vicon.com`
21. Orem, W.H.: Impacts of sulfate contamination on the Florida Everglades ecosystem. Fact Sheet FS 109-03. Reston, Virginia, U.S. Geological Survey (2004)
22. Dodds, W.K.: Freshwater ecology: concepts and environmental applications. Academic Press, San Diego (2002)

Tightly-Coupled Model Aided Visual-Inertial Fusion for Quadrotor Micro Air Vehicles

Dinuka Abeywardena and Gamini Dissanayake

Abstract. The main contribution of this paper is a tightly-coupled visual-inertial fusion algorithm for simultaneous localisation and mapping (SLAM) for a quadrotor micro aerial vehicle (MAV). Proposed algorithm is based on an extended Kalman filter that uses a platform specific dynamic model to integrate information from an inertial measurement unit (IMU) and a monocular camera on board the MAV. MAV dynamic model exploits the unique characteristics of the quadrotor, making it possible to generate relatively accurate motion predictions. This, together with an undelayed feature initialisation strategy based on inverse depth parametrisation enables more effective feature tracking and reliable visual SLAM with a small number of features even during rapid manoeuvres. Experimental results are presented to demonstrate the effectiveness of the proposed algorithm.

1 Introduction

Quadrotor Micro Aerial Vehicles (MAV) are becoming an increasingly popular aerial platform in the robotics community due to its simple construction. In the simplest form, a quadrotor MAV consists of two counter-rotating fixed pitch propeller pairs rigidly attached to a cross-like frame along with the electronics required to control the speed of each propeller. Like most other MAVs, quadrotors are underactuated and exhibit non-linear, coupled dynamics that should be stabilized with a control system, which in-turn requires fast and accurate state estimates.

All MAVs, inherently, have a limited payload capacity and power budget. These limitations make the task of obtaining fast and accurate state estimates a difficult one. Inertial Measurement Units (IMU) have been successfully used to obtain a drift free estimate of the roll and pitch angles of quadrotor MAVs [1]. However, if the aim is to make MAVs truly autonomous then other important states such as

Dinuka Abeywardena · Gamini Dissanayake
Centre Autonomous Systems (CAS), University of Technology, Sydney
e-mail: fisrtname.lastname@uts.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_11, © Springer International Publishing Switzerland 2015

yaw angle, velocity and position need to be estimated. Though absolute positioning systems such as GPS facilitate the estimation of these states, such sensing modalities are not available in most of the environments that MAVs are expected to operate in. Recent advances in sensing technology have resulted in three main alternatives to absolute positioning systems. These, namely laser range finders [2], depth cameras [3] and RGB cameras [4], have already been evaluated in the context of quadrotor state estimation.

However, none of the above sensing modalities alone provide a suitable solution for the MAV state estimation problem. In many cases, a combination of sensors is required for a successful field deployment. The focus of this paper is on one such combination that has emerged as promising possibility - the fusion of visual information from a monocular camera and inertial information from a triad of body mounted accelerometers and gyroscopes - for quadrotor MAV state estimation.

Two basic forms of Visual-Inertial Fusion (VIF) can be found in the robotics literature. The most common is what is called the "loosely-coupled" formulation, in which the algorithm processing the visual information is independent of the inertial measurements. Typically, a sequence of images from a monocular camera is processed using a Visual Simultaneous Localization and Mapping (VSLAM) algorithm to produce estimates for position (up-to-a-scale) and orientation of a robotics platform. Measurements from the inertial sensors are then fused together with the estimated produced by VSLAM in a separate filter, typically an Extended Kalman Filter (EKF). Techniques for VSLAM are well established but they are likely to produce inaccurate estimates in situations where the camera motion is fast and where the number of distinctive environmental features tracked is small. Increasing the number of features leads to higher computational costs and is not desirable for a MAV with limited no-board computing. Furthermore, given the basic motion models that are employed by VSLAM algorithms, and fast dynamics of lightweight MAV platforms, tracking features from one frame to another requires search over a large image region, again leading to higher computational costs.

These shortcomings of loosely-coupled implementations can be overcome by "tightly-coupling" visual and inertial measurements. In this form, the direct visual measurements - pixel positions of tracked features - are combined with inertial measurements within a single estimator, typically an EKF. Inertial aiding can increase the robustness of the visual feature tracking while at the same time reducing computational cost by making it possible to search only a small target region of the image during feature extraction. Thus a tightly-coupled VIF has the potential to outperform the loosely-coupled counterpart in most scenarios that involve fast manoeuvring MAVs.

Despite the advantages, examples of tightly-coupled VIF estimators are rare in the robotics literature, mainly due to their implementation complexities as compared to the loosely-coupled versions. The key contribution of this paper is to present the results of a tightly coupled VIF estimator for a quadrotor MAV flying aggressively in an indoor environment with only a handful of tracked features. The formulation in this paper also makes use of the special dynamic characteristics of the quadrotor MAV to further improve the VIF algorithm. To the best of our knowledge, this is the first time where such a formulation, specifically designed for quadrotor MAVs, has been presented and evaluated using experiments. Our algorithm also makes use of the "Inverse-Depth" (ID) parametrization for environmental features. We highlight the advantages of using ID parametrization in this context where accurate and undelayed initialization is the key to a effective SLAM algorithm with smaller number of features. Challenges associated with incorporating the inertial measurements and MAV dynamics to the ID parametrization of VSLAM are discussed and it is demonstrated how these challenges can be overcome in practice.

2 Related Work

Loosely-coupled VIF has received a significant attention in the recent past with the advent of "black-box" VSLAM algorithms such as PTAM [5]. In [6], images from a camera were processed using a variation of PTAM and then combined with accelerometer and gyroscope measurements in an EKF to estimate the pose of a quadrotor MAV. A similar setup is presented in [7] but with the addition of an air pressure sensor. Neither of the above approaches make use of the specific dynamic characteristics of the quadrotor to aid the estimation process. The approach followed by [8] is similar to the above in that it also makes use of a PTAM to estimate camera pose. They however, make use of the quadrotor motion equations to derive metric scale velocity estimates using inertial measurements, and then proceed to derive the scale of the PTAM velocity estimates by employing a maximum likelihood estimator. The PTAM position estimates, in metric scale, are then fused again with inertial measurements within an EKF. A similar approach is followed in [9], where measurements from a downward facing sonar is combined with PTAM altitude estimates to derive metric scale motion estimates. In our previous work on loosely coupled VIF [10], we unified this approach by estimating the scale within the same EKF that fuses VSLAM estimates with inertial sensor measurements while also incorporating the quadrotor specific motion equations. Instead of PTAM, we made use of a more generic bearing only SLAM formulation based on the inverse-depth feature parametrization. While both [8] and [9] claim accuracies in the order of a few centimetres, it is important to note that their experiments only consider a hovering quadrotor MAV, which is the ideal operating scenario for PTAM. Instead, here we analyse the estimation results for a flight sequence more closer to real indoor flights with frequent linear accelerations on the order of a few gs and rotational rates on the order of few tens of degrees per second. Indeed in section 6 we show that PTAM like VSLAM algorithms which rely on constant velocity motion models will fail catastrophically when attempting to produce pose estimate using the image sequence used for the experiments detailed here.

As mentioned before, tightly-coupled VIF implementations are less common in literature because of their increased complexity. In [11], an iterated EKF is used to fuse visual and inertial measurements in an attempt to estimate the inter-sensor calibration parameters and camera pose. A similar approach that makes use of an unscented Kalman filter can be found in [12]. However, in both these approaches, the focus is on estimating the calibration parameters. Therefore, the motion of the camera is restricted such that it observes a specific set of environmental features in all images. A similar formulation can be found in [13], but with experimental results that span long indoor and outdoor trajectories. However, there the algorithms were not developed with the resource constraints of MAVs in mind. A tightly-coupled VIF employing the ID parametrization for environmental features is presented in [14]. In the experimental results included there, the camera motion is largely one dimensional thus limiting the usefulness of the analysis. In addition, the lack of accurate ground truth in the last two approaches mentioned above inhibits a thorough evaluation of the estimation accuracy. In [4], a tightly-coupled implementation of bearing only inertial SLAM is used to estimate the pose of fixed wing aerial vehicle along with the position of a set of artificial landmarks placed on the ground. This work is close in spirit to the work presented here although it uses delayed feature initialisation.

The tightly-coupled VIF implementation presented here makes use of the ID feature parametrization for monocular SLAM, introduced in [15]. In this parametrization, the 3D position of environmental features are represented by a six parameter vector as opposed to the more conventional three parameters. These six parameters encode the feature location by a vector, originating from the camera optical centre and terminating in the feature position. This vector is represented by the position of the camera optical centre, the elevation and bearing to the feature and the inverse of the length of the vector. The key idea here is that this parametrization reduces the non-linearities in measurement equations of an EKF based monocular SLAM algorithm and thus improves the consistency and accuracy of the filter. Another the main advantage of this strategy is that information from tracked environmental features can be exploited without waiting for a sufficient baseline. To conserve space, we refrain from discussing the ID parametrization in detail here. More information about an open source implementation of an EKF based monocular SLAM algorithm using ID feature parametrization can be found in [16].

3 Visual-Inertial Fusion: Methodology

3.1 System Description

The platform under consideration is a quadrotor MAV affixed with an IMU (consisting of a triad of orthogonal accelerometers and gyroscopes) and a monocular camera. Without loss of generality, we assume that the IMU is located at the centre of mass of the quadrotor and that the camera and IMU coordinate frames are aligned. We name this the body coordinate frame $\{B\}$ and also define an earth fixed inertial coordinate frame $\{E\}$, which is defined by the position and heading of $\{B\}$ at the start of the VSLAM estimator but with it's vertical axis aligned with gravity (see Fig. 1). Throughout the paper we use boldface letters to denote vectors and leading superscripts to denote coordinate frame in which the vector is expressed. A trailing subscript denotes individual components of the vector.

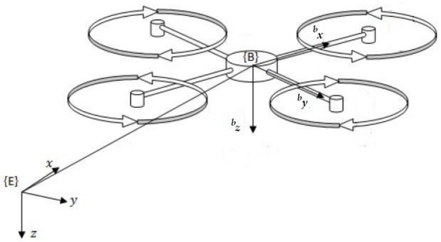


Fig. 1 Quadrotor and the coordinate systems

3.2 Process Model

The states we wish to estimate are: the position of the origin of $\{B\}$ expressed in $\{E\}$, $^e\boldsymbol{p}$; velocity of the origin of $\{B\}$ measured in $\{E\}$, but expressed in $\{B\}$, $^b\boldsymbol{v}$; orientation of $\{B\}$ with respect to $\{E\}$ expressed as a quaternion, $\boldsymbol{q}$; angular velocity of $\{B\}$ with respect to $\{E\}$, expressed in $\{B\}$, $^b\boldsymbol{\omega}$; gyroscope bias $\boldsymbol{\beta_g}$ and accelerometer bias $\boldsymbol{\beta_a}$. The states evolve according to:

$$\begin{bmatrix} ^e\dot{\boldsymbol{p}} \\ \\ ^b\dot{\boldsymbol{v}} \\ \\ \dot{\boldsymbol{q}} \\ ^b\dot{\boldsymbol{\omega}} \\ \dot{\boldsymbol{\beta}}_a \\ \dot{\boldsymbol{\beta}}_g \end{bmatrix} = \begin{bmatrix} R\,^b\boldsymbol{v} \\ R^T\boldsymbol{g} - \begin{bmatrix} k_1\,^bv_x + \eta_{vx} \\ k_1\,^bv_y + \eta_{vy} \\ \beta_{az} + \eta_{az} \end{bmatrix} \\ -0.5\boldsymbol{q} \otimes \,^b\boldsymbol{\omega} \\ \boldsymbol{\eta}_\omega \\ \boldsymbol{\eta}_{\beta a} \\ \boldsymbol{\eta}_{\beta g} \end{bmatrix} + \begin{bmatrix} \boldsymbol{0}_{3\times1} \\ 0 \\ 0 \\ 1 \\ \boldsymbol{0}_{4\times1} \\ \boldsymbol{0}_{3\times1} \\ \boldsymbol{0}_{3\times1} \\ \boldsymbol{0}_{3\times1} \end{bmatrix} {}^ba_z \tag{1}$$

where k_1 is a positive constant specific to a given quadrotor MAV[1], $\boldsymbol{g}$ is the gravity vector in $\{E\}$, R is the rotation matrix that transforms a vector from $\{B\}$ to $\{E\}$, $\otimes$ denotes the quaternion multiplication and $^b\boldsymbol{a} = \{^ba_x, ^ba_y, ^ba_z\}$ are the accelerometer measurements. Also, $\eta_{vx}, \eta_{vy}, \eta_{az}, \boldsymbol{\eta}_\omega, \boldsymbol{\eta}_{\beta a}$ and $\boldsymbol{\eta}_{\beta g}$ are zero mean White Gaussian Noise (WGN) terms denoting uncertainties in process equations. The accelerometer measurement along ^bz axis is considered as a control input to the system. The equation for $^b\dot{\boldsymbol{v}}$ of equation (1) which describes the relationship between the orientation and the translational motion is unique to a quadrotor MAV. More details on this equation along with experimental validation can be found in [17].

In addition to the above mentioned ego-motion states, the VIF algorithm also estimates the location of a number of environmental features. All features are initialized in the filter using the ID parametrization and later converted to Euclidean when sufficient information about feature depth has been acquired through repeated observations. A Euclidean feature is introduced to the filter by augmenting the state

[1] In the current formulation k_1 has to be estimated offline. Refer to [17] for details.

vector with the 3-D position of the feature $\{E\}$.

$$\boldsymbol{x}_i = \begin{bmatrix} X_i \; Y_i \; Z_i \end{bmatrix} \tag{2}$$

In contrast, an ID feature is introduced by augmenting the state vector with a 6-D vector:

$$\boldsymbol{y}_i = \begin{bmatrix} x_i \; y_i \; z_i \; \theta_i \; \phi_i \; \rho_i \end{bmatrix} \tag{3}$$

which models a 3-D point located at $\begin{bmatrix} X_i \; Y_i \; Z_i \end{bmatrix}$ as:

$$\begin{aligned} \boldsymbol{x}_i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} &= \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} + \frac{1}{\rho}\boldsymbol{m}(\theta_i, \phi_i) \\ \boldsymbol{m}(\theta_i, \phi_i) &= \begin{bmatrix} \cos\phi_i \sin\theta_i & -\sin\phi_i & \cos\phi_i \cos\theta_i \end{bmatrix}^T \end{aligned} \tag{4}$$

where $\begin{bmatrix} x_i \; y_i \; z_i \end{bmatrix}^T$ is the location of the camera when the feature was first observed, $\boldsymbol{m}(\theta_i, \phi_i)$ denotes the unit vector from that location to the feature and ρ_i is the inverse of the depth to the feature along that unit vector. Further details on the ID feature parametrization for monocular SLAM can be found in [15].

3.3 Measurement Model

Body mounted gyroscopes measure the instantaneous rotational rate of $\{B\}$ with respect to $\{E\}$. They are assumed to be corrupted by a slow-varying bias and zero-mean WGN.

$$\boldsymbol{h}_g = \; {}^b\boldsymbol{\omega} + \boldsymbol{\beta}_{\boldsymbol{g}} + \boldsymbol{\eta}_g \tag{5}$$

Accelerometers measure a combination of static and dynamic acceleration projected onto their sensing axes and therefore their measurements depend on the dynamics of the platform to which they are attached. For the case of a quadrotor MAV, accelerometer measurements can be easily derived from equation (1).

$$\boldsymbol{h}_a = \begin{bmatrix} {}^b a_x \\ {}^b a_y \end{bmatrix} = \begin{bmatrix} -k_1 \, {}^b v_x + \beta_{ax} + \eta_{ax} \\ -k_1 \, {}^b v_y + \beta_{ay} + \eta_{ay} \end{bmatrix} \tag{6}$$

Measurements to the currently tracked set of features are made in each camera image. To express these measurements mathematically, the position of the feature first needs to be expressed in $\{B\}$: $\boldsymbol{h}_c = \begin{bmatrix} h_x \; h_y \; h_z \end{bmatrix}$. For a Euclidean features at $\begin{bmatrix} X_i \; Y_i \; Z_i \end{bmatrix}$:

$$\boldsymbol{h}_c = R^T \left(\begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} - {}^e\boldsymbol{p} \right) \tag{7}$$

For ID features:

$$\boldsymbol{h}_c = R^T \left(\rho_i \left(\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} - {}^e\boldsymbol{p} \right) + \boldsymbol{m}(\theta_i, \phi_i) \right) \tag{8}$$

For both Euclidean and ID features, the actual measurements are the projection of the features onto the image plane assuming a pin hole camera. After the application of camera calibration, we arrive at:

$$\boldsymbol{h} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} u_0 - \frac{fh_y}{h_x} \\ v_0 - \frac{fh_z}{h_x} \end{bmatrix} \tag{9}$$

where f is the camera focal length and u_0, v_0 is the camera's principal point.

4 Filter Implementation

The open source implementation of the EKF monocular SLAM with inverse depth parametrization [16] was used as the basis for implementing the tightly coupled VIF algorithm detailed in the previous section. The inclusion of quadrotor specific dynamic model described by equation (1), the addition of accelerometer and gyroscope bias states and the incorporation of inertial measurement equations (5) and (6) are the main contributions of the work proposed in this paper. This section discusses some of the important aspects of that implementation.

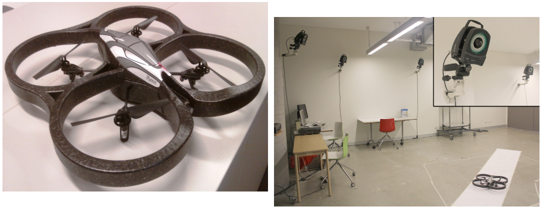
In an EKF based monocular SLAM implementation, the initialization and removal of features from the state vector needs special attention. Parameters involved in these processes had to be tuned somewhat to achieve a proper balance between the estimation accuracy and computational time. For the experiments presented here, the filter attempts to track at least 10 environmental features in each image. A feature active in the filter is only deleted when it fails a certain number of data association attempts. An upper limit of thirty was placed on the total number of active features in the filter at any point of time. For ease of implementation, when this limit is exceeded, the features are deleted in a FIFO manner although more sophisticated strategies exist. Also of importance is the method used for feature extraction and matching. The original implementation of the ID monocular SLAM detailed in [16] extracted FAST features [18] from each image along with a small image patch around the feature location as the descriptor. For each feature, a Region of Interest (RoI) is defined in each subsequent image based on the current best estimate of the camera pose and feature location. A correlation based matching within that RoI using the stored image patch establishes correspondences. While other more sophisticated feature extraction algorithms such as SIFT [19] are attractive for their robustness, we decided to proceed with FAST features and RoI based search for the computational simplicity of that approach.

One major advantage of the ID parametrization for monocular SLAM is the ability to perform un-delayed initialization of environmental features. At the first observation, each feature is incorporated into the state vector using the 6 parameter vector y_i with an mean inverse depth of ρ_{i0} and a variance in inverse depth σ_{i0}. Several rule of thumb guidelines on how to select values for both ρ_{i0} and σ_{i0} can be found in [16]. The idea is to set ρ_{i0} such that the range of depths from the minimum expected depth to infinite depth is covered within the $2\sigma_{i0}$ bound. While these rules generally provided good results for vision only experiments, some difficulties were faced when fusion with inertial information was attempted. In that case, we observed that the features that are considerably further away than the closest feature tend to end up behind the camera after the first EKF update step, especially when the camera is moving toward the features. Other authors have noted the same behaviour when attempting to incorporate metric scale information to EKF based monocular SLAM algorithms using ID parametrization [20]. To overcome this issue, we resorted to initializing the features at approximately the mean depth of features in the image rather than at the depth of the closest feature. With just a simple visual inspection of the environment, we were able to set an approximate value to the initial feature depth that would perform satisfactorily. We have also realized that, when considering the consistency of the filter, it is safer to initialize features further away than close by.

Another difficulty faced during the implementation of the tightly-coupled VIF algorithm was the tuning of filter noise parameters. The final EKF implementation had nineteen states and three different types of measurements resulting in a total of twenty two different noise parameters that had to be tuned. This proved an increasingly difficult task, unless a methodical approach to filter tuning was followed. The method followed in this work is to first focus on a reduced number of filter states and measurements. In our previous work [17], we have shown that it is possible to design a drift free velocity and attitude estimator for quadrotor MAVs using only accelerometer and gyroscope measurements. With that experience, we initiated the filter tuning process by turning off vision measurement updates and concentrating on only obtaining drift free velocity and attitude estimates. For this however, we had to provide the filter with an estimate of accelerometer biases, as it is not possible to estimate that using only inertial sensors. This tuning step involved only four states and four measurements, and was rather easy to achieve. In the next step vision measurements was turned on and we focused on obtaining accelerometer bias estimates and on reducing the drift in position estimates. Even though this still required a number of trial and error steps, it proved to be easier and more reliable than attempting to tune all noise parameters at once.

5 Experiments

Experiments presented here were performed using a real world flight dataset consisting rapid linear and angular manoeuvres, obtained with a quadrotor MAV flying in a Vicon motion capture environment approximately of size $4 \times 4 \times 3(m)$. The quadrotor platform used was the Parrot AR Drone I with a total flight weight of 420g (see Fig. 2). The AR Drone comes pre-equipped with one front facing and one downward facing cameras, along with a triad of accelerometers and gyroscopes. The front facing camera is equipped with a 93^0 wide-angle lens and has a resolution of



(a)

(b)

Fig. 2 (a) AR Drone I, (b) Vicon motion capture environment. Inset - one of the Vicon IR cameras.

320×240 pixels. Images from this camera, captured at approximately 7Hz along with accelerometer and gyroscope measurements (at 200Hz) were timestamped onboard the MAV and wirelessly transmitted to a ground station computer. All estimation tasks were performed off-board and off-line. Vicon real-time state estimates captured at 120Hz were treated as ground truth in evaluating the VIF estimates.

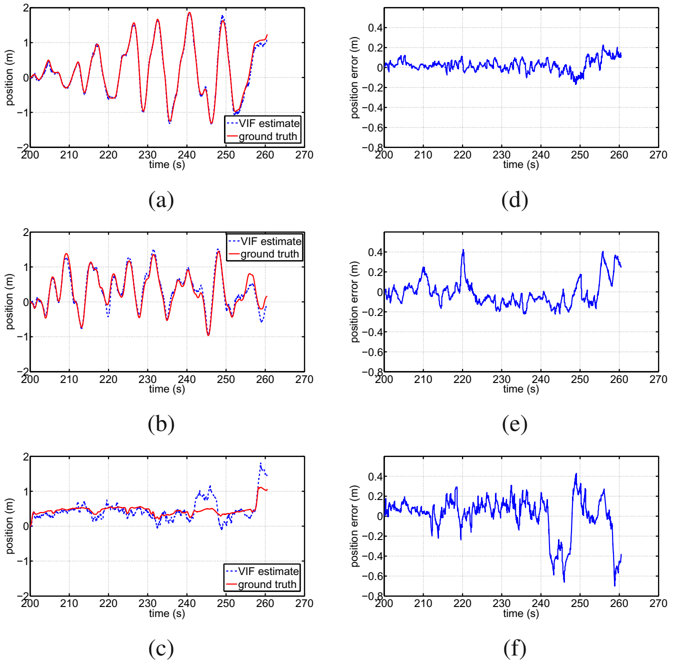
5.1 Experimental Results

Position, velocity and orientation estimates along with ground truth for the collected dataset are presented respectively in Fig. 3, Fig. 4 and Fig. 5. Note that the quaternion attitude estimate of the VIF was converted to the familiar Z-Y-X Euler angles for the purpose of illustration.

These figures clearly illustrates that the filter is producing consistent, metric scale estimates. Note how the errors in ${}^{e}p_z$ axis for position and ${}^{b}v_z$ axis for velocity are considerably higher than the other axes. This increased error stems from the lack of a dynamic relationship in the filter constraining the quadrotor motion in ${}^{b}v_z$ axis, as is the case for ${}^{b}v_x$ and ${}^{b}v_y$ axes. A similar trend can be seen in yaw angle estimate when compared to the roll and pitch estimates. Thus it is reasonable to assume that the position, velocity and attitude estimation errors would increase considerably, if not for the incorporation of the quadrotor dynamic model in the VIF algorithm.

6 A Comparison

To highlight the importance of the tightly-coupled formulation detailed in this paper, we compare its estimation accuracy and execution time with two other state estimators using the same dataset. First is a vision only formulation and we consider two variations of a typical VSLAM estimator; a vanilla ID based monocular SLAM formulation that make use of a constant velocity motion model to track features across images and an SIFT based extension of the same that does not rely on a motion model for feature tracking. For ease of reference we name the first NID (native ID) and the second SbID (SIFT based ID). The underlying filtering mechanics for both these variations are detailed in [16]. Second type of estimator that we wish to



(a) (d)

(b) (e)

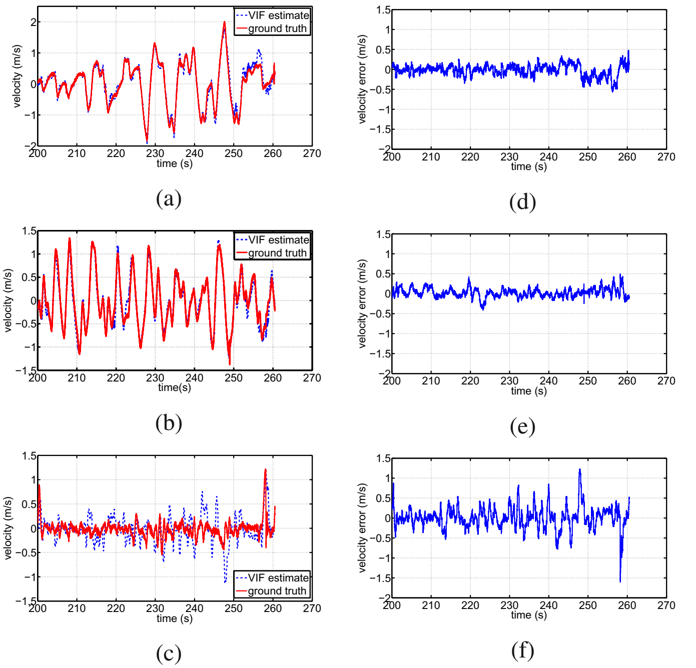
(c) (f)

Fig. 3 VIF position estimates: x - (a), y - (b), z - (c) and respective position estimation errors (d) - (f)

compare against is a loosely-coupled VIF formulation where the loose coupling of the vision and inertial information is the only difference between that and the present work. Again for ease of reference we name the these as LC-VIF(loosely-coupled VIF) and TC-VIF (tightly-coupled VIF). Note that the vision estimates required for LC-VIF were derived from SbID. More details of LC-VIF can be found in [10].

6.1 Vision Only Estimation

Fast camera movements and the close proximity of features make the gathered dataset an extremely difficult one for the conventional VSLAM algorithms that are based on constant angular and linear velocity models. Attempts to process the dataset using the open source implementation of the EKF based ID monocular SLAM algorithm [16] failed repeatedly despite meticulous tuning of the filter noise parameters, because of the inability of the process model to predict camera motion with sufficient accuracy that would enable successful feature tracking between images. Thus, we conclude that NID estimators are unable to produce meaningful results when the camera motion exhibits fast dynamics. We also remark that native PTAM algorithms fall into the same category as the NID in the sense that they also rely heavily on constant velocity motion model for feature tracking.



(a)

(d)

(b)

(e)

(c)

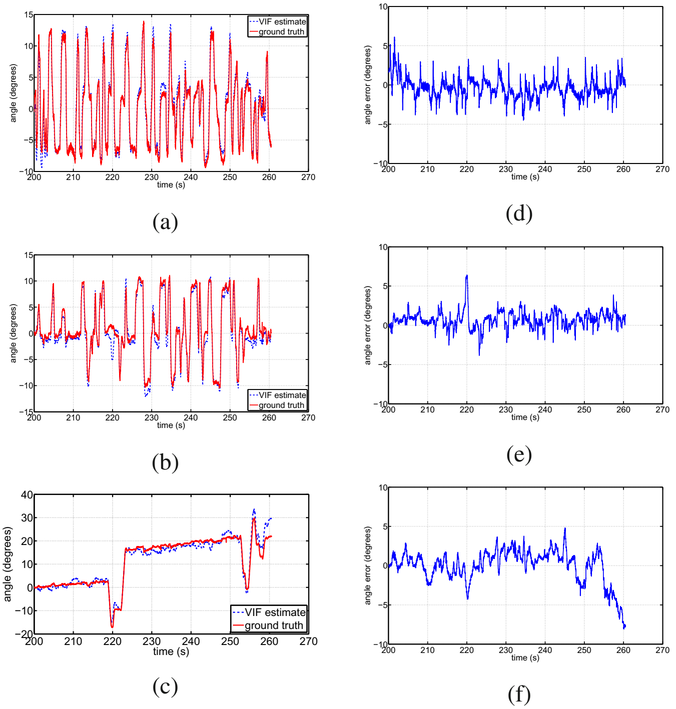
(f)

Fig. 4 VIF velocity estimates: x - (a), y - (b), z - (c) and respective velocity estimation errors (d) - (f)

It is possibly to forego the tracking step altogether by employing a high quality feature descriptor. In such an implementation, salient features and their descriptors are extracted throughout each new image and matched with the descriptors of the features currently in the map to establish initial correspondences. We modified the ID monocular SLAM implementation by Civera et al. so that it extracts SIFT features in each image, descriptors of which were then used for data association. Again due to the fast camera motions in the present dataset, we had to extract a large number of SIFT features in each image to ensure sufficient feature overlap between images. This (and the fact that the algorithms were implemented in Matlab) resulted in extremely slow processing of images - about four seconds for each image. The resulting VSLAM position estimation errors, calculated after manually scaling the position estimates, are presented in Fig. 6a.

6.2 Loosely-Coupled VIF

As detailed in [10], the goal of the LC-VIF design was to produce metric scale estimates by combining VSLAM estimates of arbitrary scale and biased inertial measurements. To achieve this the SbID estimates were fused with the accelerometer and gyroscope measurement in an EKF that was based on the dynamic motion

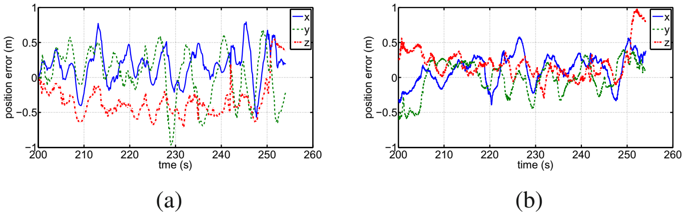


(a) (d)

(b) (e)

(c) (f)

Fig. 5 VIF attitude estimates: roll - (a), pitch - (b), yaw - (c) and respective attitude estimation errors (d) - (f)



(a) (b)

Fig. 6 (a) VSLAM only position estimation error from SbID, (b) LC-VIF position estimation error.

equations of the quadrotor MAV. Position estimation errors of the LC-VIF presented in Fig. 6b. Comparing Fig.6b with Fig. 6a, it is possible to see that the even a loose coupling of visual and inertial information help reduce the estimation errors. Also note how the estimation errors in LC-VIF change in proportion to the errors in SbID estimates. Thus we remark that the improvement of accuracy due to VIF is minimal due to the presence of considerable errors in the SbID estimate.

A summary of the performance of each estimator is presented in Table 1. Note the improvement brought about by the tightly-coupled VIF, both in accuracy and

Table 1 Performance of different estimators

	RMS position error(m/s)			Execution time (s)
Structure	x	y	z	per image
NID	NA	NA	NA	NA
SbID	0.29	0.35	0.41	4
LC-VIF	0.24	0.24	0.30	4
TC-VIF	0.06	0.13	0.19	0.3

in execution time. This clearly illustrates the advantages of the filter formulation presented here.

7 Conclusion

This paper presents the details on the implementation of a tightly-coupled VIF for the state estimation of quadrotor MAVs that also makes use of the platform specific dynamics to aid the estimation process. The experiments conducted clearly demonstrate the ability of the proposed design to derive reasonably accurate state estimates while attempting to minimize the computational expense by tracking only a handful of environmental features. Even though the current implementation is off-line and off-board, we believe it to be an important step toward real-time on-board MAV state estimation. Future extensions of this work will attempt to bridge this gap by further refining the proposed design.

References

1. Mahony, R., Kumar, V., Corke, P.: Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. IEEE Robot. Autom. Mag. 19(3), 20–32 (2012)
2. Bachrach, A., Prentice, S., He, R., Roy, N.: RANGE - robust autonomous navigation in GPS-denied environments. Journal of Field Robotics 28(5), 644–666 (2011)
3. Bachrach, A., Prentice, S., He, R., Henry, P., Huang, A.S., Krainin, M., Maturana, D., Fox, D., Roy, N.: Estimation, planning and mapping for autonomous flight using an rgb-d camera in GPS-denied environments. Int. J. Robot. Res. 31(11), 1320–1343 (2012)
4. Bryson, M., Sukkarieh, S.: Building a robust implementation of bearing-only inertial SLAM for a UAV. Journal of Field Robotics, Special Issue on SLAM in the Field 24(1-2), 113–143 (2007)
5. Klein, G., Murray, D.: Parallel tracking and mapping for small ar workspaces. In: Proc. 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 225–234 (November 2007)
6. Weiss, S., Achtelik, M., Lynen, S., Chli, M., Siegwart, R.: Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments. In: Proc. IEEE Int. Conf. Robot. Autom., pp. 957–964 (May 2012)
7. Nützi, G., Weiss, S., Scaramuzza, D., Siegwart, R.: Fusion of IMU and vision for absolute scale estimation in monocular SLAM. Journal of Intelligent and Robotic Systems 61, 287–299 (2011)

8. Engel, J., Sturm, J., Cremers, D.: Camera-based navigation of a low-cost quadrocopter. In: Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., pp. 2815–2821 (October 2012)
9. Sa, I., He, H., Huynh, V., Corke, P.: Monocular vision based autonomous navigation for a cost-effective mav in gps-denied environments. In: 2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), pp. 1355–1360. IEEE (2013)
10. Abeywardena, D., Wang, Z., Kodagoda, S., Dissanayake, G.: Visual-inertial fusion for quadrotor Micro Air Vehicles with improved scale observability. In: Proc. IEEE Int. Conf. Robot. Autom., pp. 3133–3138 (May 2013)
11. Mirzaei, F., Roumeliotis, S.: A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation. IEEE Trans. Robot. 24(5), 1143–1156 (2008)
12. Kelly, J., Sukhatme, G.S.: Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. Int. J. Robot. Res. 30(1), 56–79 (2011)
13. Jones, E.S., Soatto, S.: Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. The International Journal of Robotics Research 30(4), 407–430 (2011)
14. Piniés, P., Lupton, T., Sukkarieh, S., Tardós, J.D.: Inertial aiding of inverse depth slam using a monocular camera. In: 2007 IEEE International Conference on Robotics and Automation, pp. 2797–2802. IEEE (2007)
15. Civera, J., Davison, A., Montiel, J.: Inverse depth parametrization for monocular SLAM. IEEE Trans. Robot. 24(5), 932–945 (2008)
16. Civera, J., Grasa, O.G., Davison, A.J., Montiel, J.M.M.: 1-point RANSAC for extended Kalman filtering: Application to real-time structure from motion and visual odometry. Journal of Field Robotics 27(5), 609–631 (2010)
17. Abeywardena, D., Kodagoda, S., Dissanayake, G., Munasinghe, R.: Improved state estimation in quadrotor MAVs: A novel drift-free velocity estimator. IEEE Robot. Autom. Mag. PP(99), 1 (2013)
18. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part I. LNCS, vol. 3951, pp. 430–443. Springer, Heidelberg (2006)
19. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
20. Munguia, R., Grau, A.: Delayed features initialization for inverse depth monocular slam. In: European Conference on Mobile Robotics (2007)

Enabling Aircraft Emergency Landings Using Active Visual Site Detection

Michael Warren, Luis Mejias, Xilin Yang, Bilal Arain,
Felipe Gonzalez, and Ben Upcroft

Abstract. The ability to automate forced landings in an emergency such as engine failure is an essential ability to improve the safety of Unmanned Aerial Vehicles operating in General Aviation airspace. By using active vision to detect safe landing zones below the aircraft, the reliability and safety of such systems is vastly improved by gathering up-to-the-minute information about the ground environment. This paper presents the Site Detection System, a methodology utilising a downward facing camera to analyse the ground environment in both 2D and 3D, detect safe landing sites and characterise them according to size, shape, slope and nearby obstacles. A methodology is presented showing the fusion of landing site detection from 2D imagery with a coarse Digital Elevation Map and dense 3D reconstructions using INS-aided Structure-from-Motion to improve accuracy. Results are presented from an experimental flight showing the precision/recall of landing sites in comparison to a hand-classified ground truth, and improved performance with the integration of 3D analysis from visual Structure-from-Motion.

1 Introduction

Everyday operations of Unmanned Aerial Vehicles (UAVs) are fast becoming a reality as automation technology improves and regulations change to allow civilian applications in commercial airspace. However, there are a number of opportunities to improve the safety of these vehicles from a regulatory and operational point of view. Critical to these operations is the ability to perform a safe emergency landing in the case of engine or control surface failure. While General Aviation (GA) aircraft pilots are highly trained in the detection and safe navigation of an aircraft to an emergency landing site, we automate this process for application in small, fixed-wing UAVs, in addition to using the technology to assist pilots in full-size aircraft.

Michael Warren · Luis Mejias · Xilin Yang · Bilal Arain · Felipe Gonzalez · Ben Upcroft
Australian Research Centre for Aerospace Automation (ARCAA),
Queensland University of Technology, Brisbane, Australia
e-mail: michael.warren@qut.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_12, © Springer International Publishing Switzerland 2015

Through the use of an on-board, self-contained system that uses a downward facing camera, we apply visual classification and 3D reconstruction of the environment below the aircraft to automatically detect suitable landing sites under the flight path. Using this information, the system will plan suitable approach trajectories to land the aircraft safely without significantly damaging the system, and more importantly, not adversely impact on people or property on the ground.

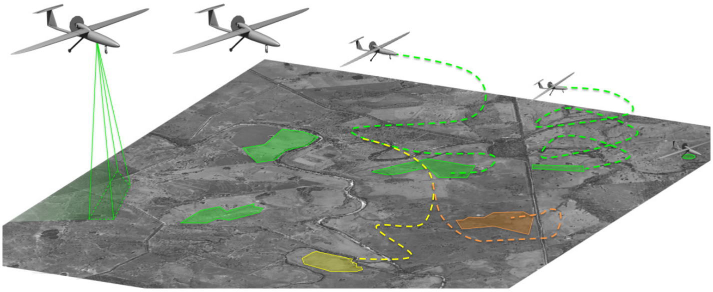


Fig. 1 Overview of the Automated Emergency Landing System: 1) normal flight, 2) site segmentation, 3) decision processing, 4) path planning, 5) landing

This paper presents an overview of the Automated Emergency Landing System (AELS), the first iteration of a fully-automated landing site detection and navigation system for fixed-wing aircraft (Fig. 1), but here we focus on the task of automated landing site detection. The Site Detection System (SDS) uses a downward facing camera and both 2D texture and 3D point clouds to analyse the suitability of sites for a forced-landing. We present results showing the accuracy of the landing site classification system to a mapped ground truth and compare both precision and recall against this hand-classified data.

The rest of this paper is outlined as follows: Section 2 reviews related literature on the topic of automated forced landings for Unmanned Aerial Vehicles. Section 3 gives an overview of the AELS, while Sections 4 and 5 details the implementation of the Site Detection System. Section 6 presents the experimental platform and dataset used for this research and demonstrates results of the implementation on the gathered data.

2 Related Work

The implementation of an automated landing site detection and navigation system for unmanned aircraft is relatively recent in the literature. The first fully integrated system for detecting, selecting and navigating to an unprepared site for a full-scale helicopter is presented by Scherer *et al.* [15]. By utilising a nodding 3-D LiDAR scanner, the system generates a high accuracy 3D point cloud of the terrain underneath a full-scale Little Bird 2-seat helicopter. By fitting a simulated aircraft footprint at regular intervals in the 3D point map, obstacle-free and relatively smooth sites are extracted as candidates for an automated landing.

This solution highlights the various challenges and design decisions for the implementation of such a system. The use of a LiDAR-based system is well suited to a GA aircraft due to their carrying capacity, relative cost of the platform and power availability, but such systems are not suited to smaller Unmanned Aircraft for these same reasons. Instead, we focus on the use of downward facing vision to perform 3D reconstruction and classification. In this context, vision is not range limited (in normal Visual Meteorological Conditions [VMC]) and is suitably cheap and low-mass for application in UAVs as small as 5kg. Since many of these low-mass UAVs are restricted to the same VMC restrictions as many GA pilots, there is no significant loss of ability in normal operations through the choice of a visual sensor. However, the use of vision from such high altitudes presents a number of challenges, particularly in terms of 3D reconstruction using Structure-from-Motion (SfM) and accurate ground classification to assess suitability.

Johnson *et al.* [6] present an alternative implementation that uses SfM to reconstruct the area under a small, unmanned rotorcraft to assess suitability and execute a safe landing of the vehicle. Yu *et al.* perform a similar task [21]. However, the aircraft in these experiments are flying at a relatively low altitude (a few tens of metres) and only use image pairs for the reconstruction task. Use of this stereo triangulation method is prone to degeneracy and unreliable at the high altitudes (500 – 5000 ft Above Ground Level [AGL]) at which we intend to operate the system due to planarity of the ground and the extremely large temporal baseline required.

As the first stage of detection of potential landing sites, the SDS includes a 2D candidate identifier based on canny-edge detection, which forms a significant component of a 2D binary landing site classifier already presented in the literature [11, 10]. The use of texture analysis and contrast descriptors to identify suitable landing areas has also been used by Garcia-Pardo *et al.* [4]. However, this approach cannot enable decision making due to its inability to detect multiple landing sites. Recently, a similar detection approach intended to aid pilots in decision making has been proposed by Shen *et al.* [16]. This approach has limited applicability at this point since it has only been tested using synthetic images from Google Earth®.

The lack of 3D information in these algorithms partially motivates the approach developed here. We extend the previous 2D classification methodology to integrate potentially noisy observations into a probabilistic representation of the ground DEM and identify strong landing site candidates. This representation is then used as a prior to trigger evaluation of the visual data in 3D (a potentially costly exercise). This allows the system to corroborate or challenge the 2D classifier's output as well as gain a finer resolution that can determine obstacles such as trees and individual buildings from high altitudes.

SfM is a well studied area of research, with a large sum of literature on the topic in recent years [14, 3]. By tracking features between camera frames with overlapping views, a 3D model of the scene structure and camera pose can be extracted. As computational speed improves and algorithms become more efficient, near real-time implementations of SfM and Visual Odometry (VO) are now in the literature [7, 12] and can perform online on commodity hardware for small workspaces.

Typically, SfM is applied on ground-vehicles as a form of VO [8, 2], and the point clouds generated are often used in obstacle avoidance and navigation tasks [13]. VO has also been applied in selected airborne applications[20]. More specialised, high altitude applications of VO, however, presents additional challenges. Accurate VO relies on triangulation from spatially separated cameras to achieve robust estimation of 3D structure, meaning a reliance on aircraft movement to achieve accuracy, and triangulation performance decreases quadratically with distance, a difficulty at the altitudes in which we intend the system to operate. Additionally, degeneracies occur when looking at planar structure, an oft-encountered scenario when using vision from high altitude. However, VO has been successfully applied in airborne applications [19, 18, 17] by taking into account many of these factors.

For the SDS, our interest is the mapping output of a VO system: a dense 3D point cloud capable of discriminating obstacles and flat surfaces that are otherwise ambiguous or indiscernible from a 2D image. While SfM provides a sparse 3D cloud, at the altitude range we intend for the system to operate, we employ denser mapping using high-resolution depth maps and the integration of this data into a 3D mesh using Poisson reconstruction [5].

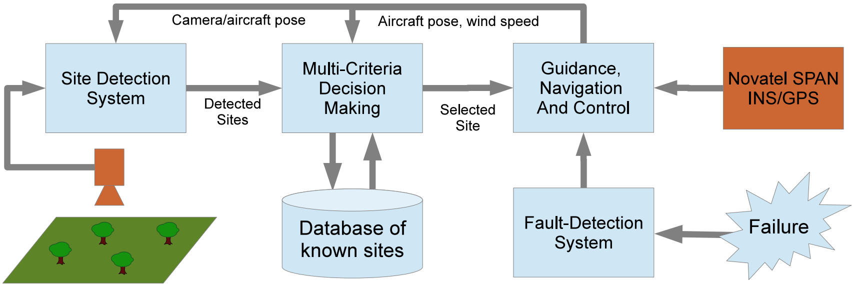


Fig. 2 An overview of the AELS and its components

3 System Overview

The AELS (Fig. 2) has four major components:

1. The Fault Detection System (FDS), an automated system for detection of in flight failure modes.
2. The Site Detection System (SDS), a system for detecting and characterising feasible landing sites below the aircraft, the focus of this paper.
3. The Multi-Criteria Decision Maker (MCDM), a continuous estimator that chooses feasible landing sites based on certain criteria such as terrain ruggedness, slope and obstacles as well as wind direction/speed in preparation for an emergency.
4. The Guidance Navigation and Control (GNC) system, for planning and navigating the aircraft to a safe aim point in preparation for a final landing manoeuvre.

Each operates in a loose hierarchy between sensing a failure and navigating to a safe landing location.

This paper focuses on the Site Detection System (SDS) as the primary method of providing landing site candidates to the higher level decision making and guidance algorithms. Using on-board sensing and prior information such as satellite and DEM data, the SDS is required to detect landing sites on the ground that meet minimum criteria of size, slope, variance and proximity to obstacles such as trees and buildings.

The SDS currently consists of three major components:

1. A binary classification algorithm that operates on purely 2D data, classifying the pixels of each image into *safe* or *not-safe* based on derivative and intensity measures, and
2. A world model that incorporates a terrain ruggedness prior and 2D image observations into a Bayesian model by projecting the observed classification of *safe*/*not-safe* from the image onto the world plane, accounting for altitude differences in the terrain.
3. A dense 3D reconstruction algorithm that leverages Poisson reconstruction to assess potential landing sites for suitability in relation to ground variance, obstacles and slope.

The major components of the SDS can be seen in Figs. 4 and 6. The map is initially split into small segments of $100m^2$ and the initial probability of *safe* established from a-priori data. The 2D classifier detects candidate landing sites in the image before projecting these observations into the world plane (Fig. 4). Once a contiguous landing site is established that meets minimum size and probability requirements, an SfM routine uses the imagery to construct a dense 3D surface model to refine and check the estimate (Fig. 6). From this model, the surface normals are used to determine relatively flat and non-flat areas and segment potential obstacles from the world plane (Fig 11). We describe these modules in more detail in the following sections.

4 2D Landing Site Pre-classifier

The 2D landing site classifier has already been described in the literature [11, 10]. It operates purely on 2D imagery, without any temporal information, to classify the pixels in an image into a binary *safe*/*not-safe* classification by detecting Canny Edges in the camera image and performing a dilation to expand the local *unsafe* region. Forests, streets, buildings and cultivated land will all likely have a strong response to the Canny edge detector with a high incidence of edges, while grassed areas and water-bodies such as lakes and rivers will be highly uniform. The binary classification process is repeated for each image as it is captured. An example of this output is shown in Fig. 3

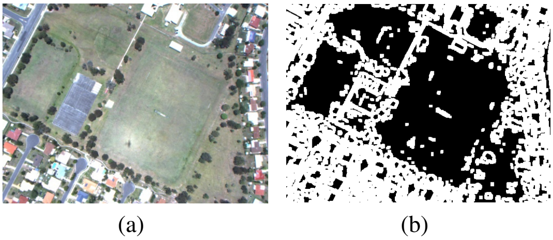


Fig. 3 a) Example image and, b) canny edge detection and expansion

4.1 Bayesian World Model

In order to determine contiguous classified landing sites, the 2D observations are projected into a 3D world model (Fig. 4) using the known camera pose and the camera intrinsics model. Camera poses are transformed from the data extracted from a Novatel SPAN INS/GPS system that gives a highly accurate 6DOF pose of the camera in the world frame, and the world is represented in a local Universal Transverse Mercator (UTM) co-ordinate system. This world plane L is divided into a large grid with 10×10 metre segments $L_{i,j}$ that contain the necessary information about the grid point, including altitude, relative classification probabilities, slope and structure variance, where i,j represent the grid index in the world plane. Inherently, each grid point has a corresponding probability of *safe*, $P(S)$ (and inverse *not-safe* $P(NS) = 1 - P(S)$), depending on the properties of the site. We seek to determine the true binary classification from the fusion of the noisy observations from the 2D classifier, prior from a DEM and a 3D reconstruction from the on-board imagery and camera poses. For each image gathered by the on-board camera, the pixels

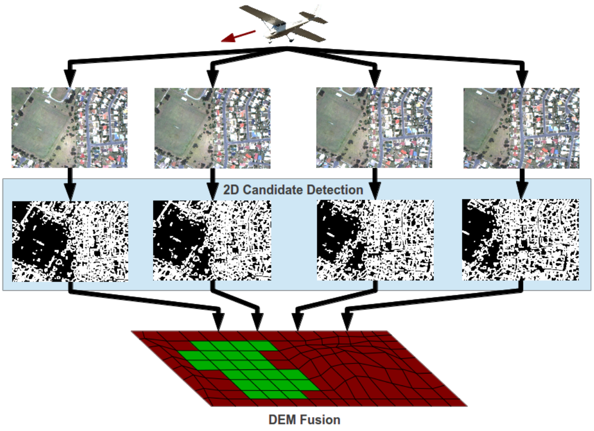


Fig. 4 An Overview of the 2D Pre-classifier Fusion

$x_C = \begin{bmatrix} u & v & 1 \end{bmatrix}^T$ are warped from the image plane to the world plane $x_L = \begin{bmatrix} x & y & 1 \end{bmatrix}^T$ (with local height extracted from the known DEM) via a plane-to-plane homography:

$$x_L = HK^{-1}x_C \tag{1}$$

where K is the (known) camera intrinsics matrix. Each pixel from the 2D classified image is then binned into the closest corresponding grid cell of the height-modified world plane and a winner-takes-all strategy determines the winner observation z_i at each grid point from this set of most recent pixels, where $z_1 = \mathit{safe}$ and $z_2 = \mathit{not\text{-}safe}$. Due to the inherently noisy observations from the 2D classifier, due to perspective change and shadowing of the aircraft, a pure integration of the positive classification will yield poor results. To counteract this, a sensor model is derived from the observations and fused into the map via a recursive Bayesian update for the cell:

$$P(S)_{k+1} = \alpha P(z_i|x_j)P(S)_k \tag{2}$$

$$P(NS)_{k+1} = \alpha(1 - P(z_i|x_j))P(NS)_k \tag{3}$$

where we define α as a normalising constant. Each grid-cell or site can be given a uniform prior $P(S) = 0.5$ or, alternatively, a prior from an external set of data. Since each grid-point that is observed is typically observed up to 20 times due to aircraft speed, altitude and frame-rate, a close to uniform sensor model describing $P(z_i|x_j)$ is used to determine the probability of *safe* to counteract the potentially noisy output of the 2D classifier. From empirical evaluation, we derive the sensor observation model as $P(z_1|x_1) = 0.52$ and $P(z_2|x_2) = 0.51$. Once a grid-point leaves the set of visible points, it can then be finally classified as binary *safe/not-safe* depending on whether the probability exceeds an empirically chosen threshold ($P(S) > T$). We explore the selection of this minimum threshold in the results.

4.2 Terrain Ruggedness: Generating a Classification Prior

While a uniform prior may be suitable for the binary classification of the world environment, a more representative prior can be generated from extensive a-priori knowledge about the environment. In many areas, a DEM of varying resolution is often available that allows the calculation of properties such as slope and terrain ruggedness.

While slope is calculated in degrees from the plane, the Terrain Ruggedness Index (TRI) is calculated as the mean difference in altitude from its neighbours:

$$\mathrm{TRI}(L_{i,j}) = \frac{\sum_{p=i-1}^{i+1} \sum_{q=j-1}^{j+1} |Z_{p,q} - Z_{i,j}|}{8} \tag{4}$$

Using a hyperbolic function to map TRI (0 → ∞) to a probability (0 → 1), a prior is established that helps to eliminate areas that may look uniform, but have a high degree of terrain variance:

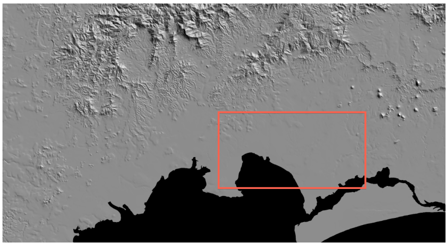


Fig. 5 A section of the prior DEM, shaded according to slope, with the approximate region flown highlighted in red

$$P(S)_0 = \frac{1}{0.1\mathrm{TRI}(L_{i,j}) + 1.8} \tag{5}$$

The coefficients of the hyperbolic function are chosen empirically to meet a maximum safe probability $P(S) \approx 0.55$ at a TRI = 0.0 (flat) and $P(S) = 0.5$ at a TRI of 4.0, corresponding to a mean variation of 0.5m on a 25m resolution DEM. We implement this prior to take into account that, in some cases, heavily forested areas may look uniform from a 2D perspective but contain a high degree of ruggedness. Alternatively, farmed land may also look uniform, but subject to a high degree of slope. A non-uniform prior helps to down-weight these particular observations and establish a better model of *safe* terrain.

In addition, knowledge of areas covered by water bodies is a useful output of such a prior, and can be easily included in the world model. Given that coastlines do not change significantly over long periods, this is an extremely strong prior that assists where an on-board sensor will likely fail to successfully classify a site due to the non-static nature of the scene. Hence, areas known to be water in the DEM are given a *safe* classification with probability $P(S) = 1.0$.

However, despite the assistance of such a set of priors, many agricultural areas can change: forests can be cleared and fields replanted with trees. While a DEM can give broad scale knowledge of terrain, it does not give up-to-the-minute knowledge about land-use changes that is available from using an on-board sensor. It is for this reason we include an active sensor, in addition to it's ability to increase model precision.

5 3D Landing Site Classification

While the 2D classifier can infer the suitability of a landing site to a large degree, a significant amount of fine information is lost, and many areas of land that are suitable for a forced landing do not necessarily respond appropriately to a canny edge classifier. By performing 3D analysis on a candidate site, local obstacles, terrain smoothness and other data about the 3D environment can be better determined to a higher resolution than both the DEM and 2D classifier. This, however, can come at high cost: a fully-featured SfM routine is computationally expensive on limited hardware suited to deployment on a small UAV, particularly in relation to the dense 3D reconstructions required for 3D analysis. For these reasons, we trigger the 3D analysis only when certain criteria are met: the *safe* classification for a set of grid points must exceed a minimum probability of 80%, and they must lie in a contiguous area of minimum size. While this area may vary depending on the aircraft, we set a minimum contiguous area of 2000m^2 to suit the test aircraft, a full-scale Cessna 172.

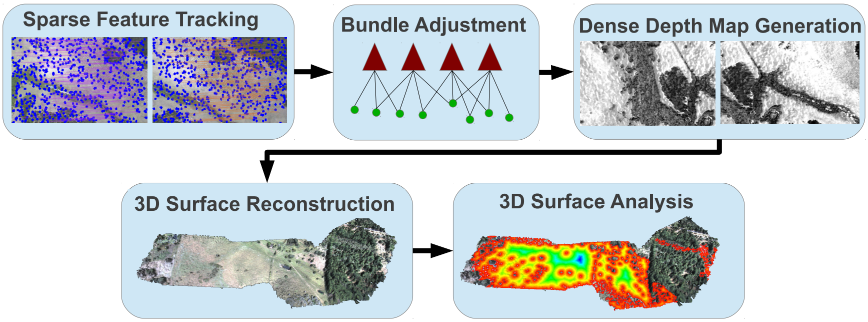


Fig. 6 The 3D Analysis pipeline

The 3D reconstruction routine follows a standard Structure-from-Motion pipeline, with some modifications. Once a contiguous area is recognised from the 2D classifier, those camera frames that observe the candidate site are flagged for the SfM pipeline (as poses and views are already known to a high degree of accuracy). To reduce complexity, frames are subsampled from the incoming stream at about 5Hz. Additionally, instead of a structure based pose update, poses are extracted directly from the INS solution to give an accurate estimate of pose in the world frame without requiring scale or other transforms to align the poses. SURF [1] features are tracked between frames and initial structure triangulated between matched features. To account for any triangulation and pose errors, a monocular bundle adjustment is applied to the set of flagged frames.

From this optimised pose and scene structure, dense depth maps are generated from those images that observe a potentially safe grid point, using a Semi-Local Method for iterative refinement [9] in a GPU-based architecture to achieve depthmap generation at about $2Hz$ on a consumer NVIDIA GPU. A Poisson mesh [5] surface estimation is then applied to merge and filter out erroneous depths from the dense maps.

5.1 3D Analysis

Once the meshed surface has been estimated, a local plane is fitted to the data covering the candidate region via a 3-point plane estimator, utilising RANSAC to remove

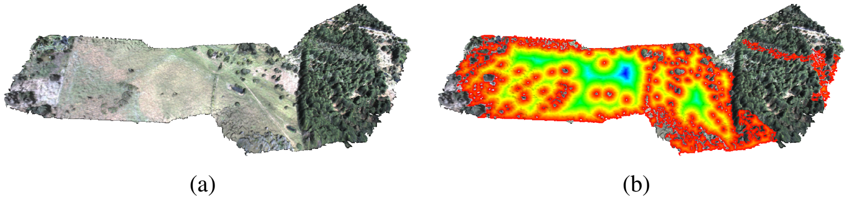

(a) (b)

Fig. 7 (a) Reconstructed dense surface model generated from 83 images (b) Corresponding cost-map of proximity to high-angle normals from a fitted ground-plane)

outliers that correspond to objects off the plane. From this plane, slope relative to the world plane and variance of the structure are easily extracted as properties.

In order to find flat contiguous areas that correspond to safe landing zones, surface normals are extracted at regular intervals by fitting local planes via Principle Component Analysis (PCA) of the points in a 5m radius neighbourhood and determining the corresponding normal. The angle between this normal and the corresponding local ground plane is then determined linearly:

$$\theta = \arcsin \frac{|n_1 u_1 + n_2 u_2 + n_3 n_3|}{\sqrt{n_1^2 + n_2^2 + n_3^2}\sqrt{u_1^2 + u_2^2 + u_3^2}} \tag{6}$$

where $\mathbf{u} = [u_1, u_2, u_3, 1]$, $\mathbf{n} = [n_1, n_2, n_3, 1]$ correspond to the plane coefficients of the local normal and fitted ground plane respectively, and θ is the relative angle between the normal and the fitted ground plane.

The surface is then classified into *safe* and *not-safe* via the gathered properties: if the surface point diverges from the plane by more than 5m or the surface normal angle θ is less than $80°$, the point is considered as not corresponding to the local plane and flagged as *not-safe*. From this analysis, contiguous regions classified as safe are extracted by a nearest-neighbour search expansion. These contiguous areas are then mapped into the 10×10m world grid and classified as *safe*.

For fixed-wing aircraft, landing sites must meet certain minimum criteria related to their width and breadth. In most cases, their length must far exceed their width. For this reason, a 2 dimensional mask that corresponds to the minimum safe landing footprint for the aircraft is applied to the classified world grid to find these zones.

In addition, for an upstream control algorithm, the planner must plan a path that approaches the landing site with the maximal length. Principle Component Analysis (PCA) is again used to determine the dominant angle from which to approach the site by calculating the eigenvectors of the 2-variable covariance matrix corresponding to X and Y directions. This is easily converted to a compass direction and passed as an additional property of the site to the MCDM for path planning. Additionally, the ratio of the eigenvalues can be used to determine the relative weight applied to selection of a dominant final approach angle.

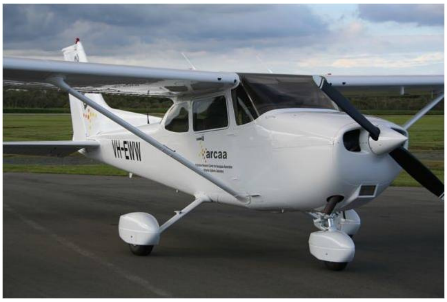
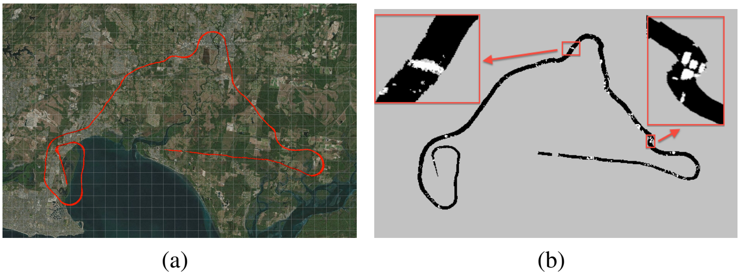


Fig. 8 The data-gathering aircraft, fitted with a Novatel SPAN INS/GPS Navigation system and downward facing camera for site detection

6 Experiments

A set of experiments was designed to test the efficacy and robustness of both the 2D and 3D site detection system, focusing on both recall ability and precision. Using ARCAA's Airborne Systems Lab (ASL) (Fig. 8), a dataset was gathered from a flight over the South-East region of Queensland, Australia. The imaging component includes 10Hz imagery from a downward facing 1024 × 768 pixels Flea2 camera with 4mm lens. 200Hz 6DOF pose-estimates were also recorded from the on-board Novatel SPAN INS/GPS system. The aircraft was flown for a distance of 67km at altitudes from 100-1000ft AGL, covering a range of terrain types including water, beach, townships, mangrove swamp, farmland, forest and crop. For this trajectory, a high resolution satellite map was ground-truth classified into a broad set of categories including grass, trees, water, crop, road and buildings. The classifications were then split into binary classes based on their suitability: *safe* or *not-safe*. Water, roads, grass and crop-land were classified as *safe* due to their relatively flat surface away from civilisation, while trees and buildings were classified as *not-safe* due to

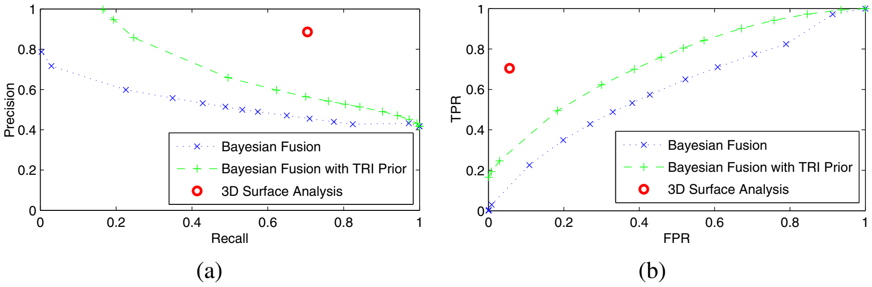

(a)

(b)

Fig. 9 (a) Overview of the flight path of the aircraft, showing overview satellite map and trajectory (Map attribute: Nearmaps.com), grid at 1km resolution. (b) Corresponding binary classification from Bayesian Fusion with 80% minimum classification threshold, with inset highlights. (grey: unknown, black: unsafe, white: safe) the likelihood of interaction with persons or damage to the aircraft. These classifications could change depending on aircraft size and whether the vehicle is manned. Each classification was binned into a world grid corresponding to the 2D Bayesian world model derived from the observation data.

6.1 Results

Figure 10 shows the relative performance of the Bayesian pre-classifier, both with and without a terrain ruggedness prior, and the 3D classifier in detecting and correctly classifying a safe landing site. For fairness in the 3D classification, and to demonstrate accuracy of precision, we evaluate precision and recall for the 3D classifier only over those grid points at which a 3D reconstruction was triggered, but evaluate the 2D classifier over the whole set of observed grid points. For this analysis, the minimum threshold probability required for the Bayesian classification to successfully classify a site was varied from $P(S) = 0.1$ to $P(S) = 0.999$. As can be seen, with an extremely strict minimum threshold probability of *safe* at $P(S) = 0.999$, precision of the 2D classifier approaches 80%, at the expense of recall.



(a) (b)

Fig. 10 a) Precision/Recall for Bayesian fusion with (blue) a uniform prior and (cyan) a terrain-ruggedness derived prior. b) ROC curve of the same data, showing similar improvements.

Utilising a terrain ruggedness prior increases both recall and precision by accounting for terrain that is not relatively smooth. Additionally, some of the gain in precision is from successfully classifying water bodies such as lakes and ocean.

Using a minimum threshold of 80% from the TRI-derived classification, 3D analysis was triggered to cover approximately 32% of the covered area, significantly reducing analysis time. Using the 3D classifier, precision and recall improve dramatically due to the inclusion of strong 3D information that is independent of the 2D classifier As there is no Bayesian probability associated with the 3D classifier, only a single data point is available (Note the red circle in Fig. 10). For the 3D classifier, checking of boundary points is dilated by 1 grid-point unit to account for mis-registration caused by errors in the ground-truth map and projection of the 3D model. An example of the 3D classification and comparison to ground-truth is

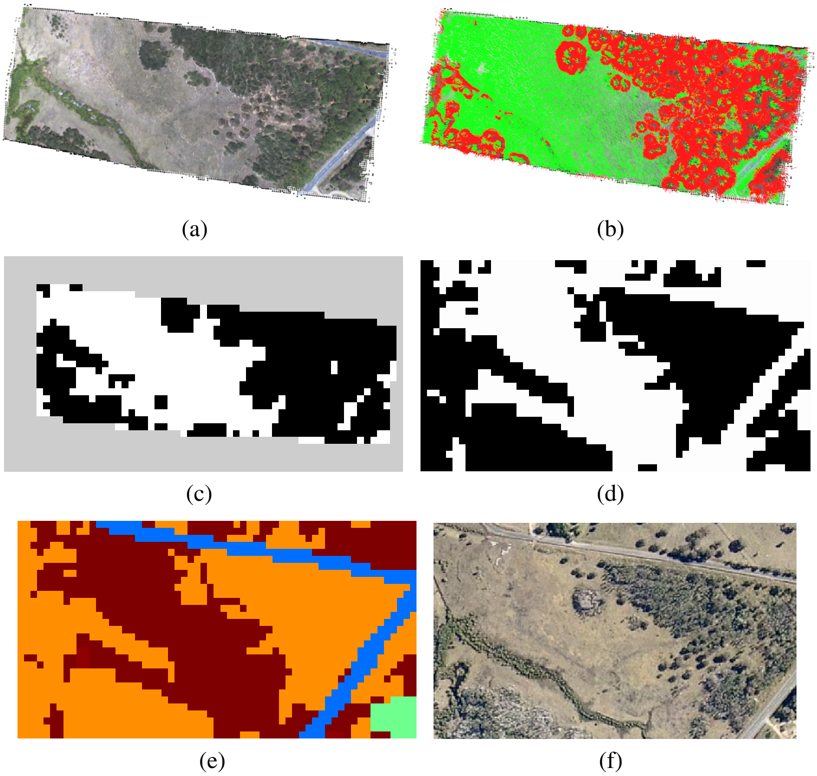


Fig. 11 Example output classification from dataset covering an area of approximately 530×210m a) 3D Reconstruction, b) 3D Surface Analysis, c) Binary classified map (black: not safe, white: safe, grey: unknown), d) Corresponding ground truth binary classified map, e) Classes (orange: trees, red: farmland, light blue: road, green: structure), f) Corresponding satellite map.

shown in Fig. 11. Here it must be noted that for a forced landing, precision has far greater importance than recall, as the ability to land safely is the key requirement rather than detecting all candidate sites. Also note that land-use changes between the capturing of satellite data and the presented dataset account for some of the reduced precision. In these experiments, the 2D Canny edge detection and bayesian integration work in online time, updating at the 10Hz rate of the incoming imagery. For this paper, the 3D scene reconstruction and analysis remains an offline process, operating in a batch scheme for a set of frames that takes approximately 50-60 seconds per set of 30 frames. With algorithmic improvements, including the leverage of a GPU and other hardware speed improvements, the speed of 3D reconstruction will improve.

7 Conclusion

An automated landing site detection system has been presented that is capable of finding safe landing sites under a fixed-wing aircraft using purely visual data. The algorithm leverages Structure-from-Motion and dense 3D point cloud analysis to determine safe landing sites on a high resolution grid, incorporates a 2D pre-classifier and Bayesian fusion in a world model to reduce the required 3D processing. Results are presented showing the performance of both the 2D pre-classifier and the the performance of the 3D surface analysis, showing a precision of 96% at a recall of 76%, showing the method is capable of performing consistently on field-gathered data.

Future work includes improving performance of the algorithm to perform near-online and incorporating the system into a demonstration aircraft capable of performing an automated forced landing from failure to final approach.

Acknowledgements. The research presented in this paper forms part of Project ResQu led by the Australian Research Centre for Aerospace Automation (ARCAA). The authors gratefully acknowledge the support of ARCAA and the project partners, Queensland University of Technology (QUT); Commonwealth Scientific and Industrial Research Organisation (CSIRO); Queensland State Government Department of Science, Information Technology, Innovation and the Arts; Boeing and Insitu Pacific. Thanks to Duncan Greer and David Robinson for flight testing and data analysis.

References

[1] Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded Up Robust Features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part I. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)

[2] Comport, A., Malis, E., Rives, P.: Real-time Quadrifocal Visual Odometry. The International Journal of Robotics Research 1 (2010)

[3] Davison, A.J., Reid, I.D., Molton, N.D., Stasse, O.: MonoSLAM: Real-Time Single Camera SLAM. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(6) (June 2007)

[4] Garcia-Pardo, P.J., Sukhatme, G.S., Montgomery, J.F.: Towards Vision-Based Safe Landing for an Autonomous Helicopter. Robotics and Autonomous Systems 38 (January 2002)

[5] Hoppe, H.: Poisson Surface Reconstruction and its Applications. In: ACM Symposium on Solid and Physical Modelling (2008)

[6] Johnson, A., Montgomery, J., Matthies, L.: Vision Guided Landing of an Autonomous Helicopter in Hazardous Terrain. In: International Conference on Robotics and Automation. IEEE (2005)

[7] Klein, G., Murray, D.: Parallel Tracking and Mapping for Small AR Workspaces. 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality (November 2007)

[8] Konolige, K., Agrawal, M., Sola, J.: Large Scale Visual Odometry for Rough Terrain. In: International Symposium on Robotics Research (2007)

[9] McKinnon, D., Smith, R.N., Upcroft, B.: A Semi-Local Method for Iterative Depth-Map Refinement. In: International Conference on Robotics and Automation, ICRA (2012)

[10] Mejias, L., Fitzgerald, D.: A Multi-Layered Approach for Site Detection in UAS Emergency Landing Scenarios Using Geometry-Based Image Segmentation. In: International Conference on Unmanned Aircraft Systems (ICUAS). IEEE (May 2013)

[11] Mejias, L., Fitzgerald, D., Eng, P., Liu, X.: Forced Landing Technologies for Unmanned Aerial Vehicles: Towards Safer Operations. Aerial Vehicles (2009)

[12] Newcombe, R., Lovegrove, S., Davison, A.: DTAM: Dense Tracking and Mapping in Real-Time. In: International Conference on Computer Vision (November 2011)

[13] Pollefeys, M., Nistér, D., Frahm, J.-M., Akbarzadeh, A., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Kim, S.-J., Merrell, P., Salmi, C., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewénius, H., Yang, R., Welch, G., Towles, H.: Detailed Real-Time Urban 3D Reconstruction from Video. International Journal of Computer Vision 78(2-3) (October 2007)

[14] Scaramuzza, D., Fraundorfer, F., Siegwart, R.: Real-Time Monocular Visual Odometry for On-Road Vehicles with 1-Point RANSAC. In: International Conference on Robotics and Automation, ICRA (2009)

[15] Scherer, S., Chamberlain, L., Singh, S.: Autonomous Landing at Unprepared Sites by a Full-Scale Helicopter. Robotics and Autonomous Systems 60(12) (December 2012)

[16] Shen, Y.-F., Rahman, Z.-U., Krusienski, D., Li, J.: A Vision-Based Automatic Safe Landing-Site Detection System. IEEE Transactions on Aerospace and Electronic Systems 49(1) (January 2013)

[17] Warren, M., McKinnon, D., He, H., Glover, A., Shiel, M.: Large Scale Monocular Vision-only Mapping from a Fixed-Wing sUAS. In: Yoshida, K., Tadokoro, S. (eds.) Field and Service Robotics. STAR, vol. 92, pp. 495–509. Springer, Heidelberg (2014)

[18] Warren, M., Upcroft, B.: High Altitude Stereo Visual Odometry. In: Proceedings of Robotics: Science and Systems (2013)

[19] Warren, M., Upcroft, B.: Robust Scale Initialization for Long-Range Stereo Visual Odometry. In: IROS (2013)

[20] Weiss, S.M.: Vision Based Navigation for Micro Helicopters. PhD thesis, Swiss Federal Institute of Technology (2012)

[21] Yu, Z., Nonami, K.: 3D Vision Based Landing Control of a Small Scale Autonomous Helicopter. International Journal of Advanced Robotics Systems (2007)

INS Assisted Monocular Visual Odometry for Aerial Vehicles

Ji Zhang and Sanjiv Singh

Abstract. The requirement to operate aircrafts in GPS denied environments can be met by use of visual odometry. We study the case that the height of the aircraft above the ground can be measured by an altimeter. Even with a high quality INS that the orientation drift is neglectable, random noise exists in the INS orientation. The noise can lead to the error of position estimate, which accumulates over time. Here, we solve the visual odometry problem by tightly coupling the INS and camera. During state estimation, we virtually rotate the camera by reprojecting features with their depth direction perpendicular to the ground. This allows us to partially eliminate the error accumulation in state estimation, resulting in a slow position drift. The method is tested with data collected on a full-scale helicopter for approximately 16km of travel. The estimation error is less than 1% of the flying distance.

1 Introduction

This paper addresses the problem of vision-based state estimation for an aerial vehicle. Typically, vision-based method is useful in the cases where GPS is unavailable or insufficiently accurate. On aerial vehicles, continuously accurate GPS positioning can be hard to ensure, especially when the vehicle flies at a high speed. Visual odometry [1, 2] becomes a supplemental method. Multiple cameras fixed on the aircraft can be used to recover 6DOF motion, but this requires that the baseline between the cameras to be at least a non-trivial fraction of the vehicle elevation above the ground. That is, if a small baseline is used, the cameras reduce to a monocular camera when the vehicle flies at a high altitude. If the cameras are separated significantly, camera calibration becomes hard and accuracy can be uninsured.

This paper uses a monocular camera looking downward toward the ground. The scale of translation is solved by the distance of the vehicle above the ground measured by an altimeter. We model the imaged ground as a locally flat patch with

Ji Zhang · Sanjiv Singh
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA
e-mail: {zhangji,ssingh}@cmu.edu

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_13, © Springer International Publishing Switzerland 2015

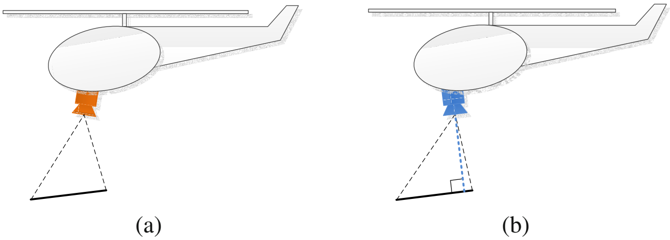


Fig. 1 Illustration of the proposed method. (a) shows the real position of the camera. During state estimation, features are reprojected with their depth direction perpendicular to the ground. This equals to rotating the camera to a virtual position pointing perpendicularly to the ground, as (b). We will show in this paper that using features associated with the virtual camera for the state estimation can decelerate the accumulation of the motion estimation error.

two rotation DOFs. The proposed method estimates the translation and the inclination angles of the ground patch concurrently.

To deal with the noise in INS orientation, we propose to reproject features with their depth direction perpendicular to the ground patch. This equals to rotating the camera virtually to be perpendicular to the ground, as shown in Fig. 1. By doing this, we can partially eliminate the accumulation of translation estimation error–the accumulated translation error introduced by roll and pitch angle noise from the INS largely cancels itself overtime, especially when the vehicle flies at a constant altitude. Also, we find that it is hard to prevent propagation of the yaw angle noise and the altimeter noise in the state estimation, but we can only reduce the noise amount from the error sources. Correspondingly, we implement a Kalman filter [3] to reduce the yaw angle noise. We also adopt a high quality laser altimeter on the aircraft to obtain accurate elevation measurement. The result is state estimation with relative error less than 1% of the flying distance.

The rest of this paper is organized as follows. In section 2, we present related work. In section 3, we define assumptions and coordinate systems. The method is overviewed in Section 4, and solved in detail in Section 5. Analysis of error propagation is given in Section 6. Experimental results are presented in Section 7 and conclusion is made in Section 8.

2 Related Work

Vision based methods are now common for vehicle state estimation [4, 5]. Typically, the problem solves 6DOF camera motion in an arbitrary environment. When stereo cameras are used [6], the relative poses of the two cameras function as a constraint that helps solve the motion estimation problem. For example, Konolige, at al's stereo visual odometry recovers the camera motion from bundle adjustment [7]. The method is integrated with an IMU which handles orientation drift of the visual odometry in long distance navigation. For a monocular camera, if the camera motion is unconstrained and no prior knowledge is assumed about the environment, the scale ambiguity is generally unsolvable. Klein and Murray develop a visual SLAM method by parallel tracking and mapping of monocular imagery [8]. The method is improved by Weiss and modified to be visual odometry [9].

When using a monocular camera, if the camera motion follows certain constraint, the scale ambiguity can be solved in constrained cases. On the other hand, if certain a prior knowledge or constraint about the environment is available, it can also assist to solve the motion estimation problem. For example, Artieda, et al's visual SLAM uses a front looking camera mounted on an aerial vehicle [10]. The scale ambiguity is solved by assuming some of the feature points with known 3D coordinates. Conte and Doherty's visual navigation system works for flights at a relatively high altitude such that the ground is considered as flat and level [11]. The vehicle motion is solved by planar homography between images taken from the ground. The method also uses geo-referenced aerial images to fix the visual odometry drift. Caballero, et al's visual odometry also assumes flat ground and uses planar homography [12]. However, the method does not require the ground to be level and online recovers its orientation with respect to the vehicle. The scale is solved by the vehicle elevation above the ground measured by a range sensor.

Our method is similar to [12] in the sense that both assume the imaged ground to be locally flat but not necessary level. However, our method does not rely on planar homography. The orientation readings from the INS are directly used in solving the translation in a tightly coupled fashion. The result is that our method solves a problem with less DOFs. Further, the method is designed to be insensitive to the INS orientation errors, and therefore has a slow position drift.

3 Assumptions and Coordinate Systems

The visual odometry problem addressed in this paper is to estimate the state of an aerial vehicle using a monocular vision system, an INS and an altimeter. We assume that the camera is well modeled as a pinhole camera [13]. The camera intrinsic parameters are known from pre-calibration, and the lens distortion is removed. As a convention in this paper, we use left uppercase superscription to indicate coordinate systems, and right superscription k, $k \in Z^+$ to indicate image frames. We use $\mathscr{I}$ to denote the set of feature points. We define two coordinate systems.

- Image coordinate system $\{I\}$ is a 2D coordinate system with its origin at the left upper corner of the image. The u- and v- axes in $\{I\}$ are pointing to the right and downward directions of the image. A point i, $i \in \mathscr{I}$, in $\{I^k\}$ is denoted as ${}^I\boldsymbol{x}_i^k$.
- Camera coordinate system $\{C\}$ is a 3D coordinate system. As shown in Fig. 2, the origin of $\{C\}$ is at the camera optical center with the z-axis coinciding with the camera principal axis. The $x-y$ plane is parallel to the camera image sensor with the x-axis pointing to the forward direction of the vehicle. A point i, $i \in \mathscr{I}$, in $\{C^k\}$ is denoted as ${}^C\boldsymbol{X}_i^k$.

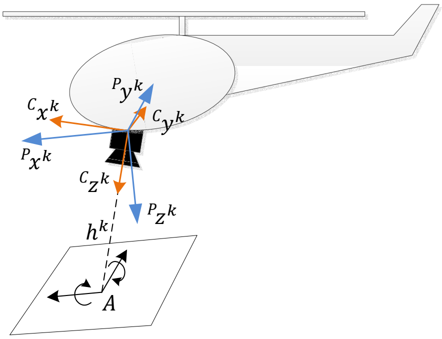


Fig. 2 Illustration of the coordinate systems and ground model. $\{C^k\}$ is the camera coordinate system at frame k. $\{P^k\}$ is a coordinate system with its $x-y$ plane parallel to the ground patch. A is the intersection of the z-axis of $\{C^k\}$ with the ground. The distance from A to the origin of $\{C^k\}$, h^k, is measured by an altimeter. The ground patch is modeled with roll and pitch DOFs around A.

We model the imaged ground as a locally flat patch, as shown in Fig. 2. Let A be the intersection of the z-axis of $\{C^k\}$ with the ground patch. The distance between A and the origin of $\{C^k\}$ is measured by an altimeter, denoted as h^k. The ground patch is modeled to have roll and pitch DOFs around A. Here, we define another coordinate system.

- Parallel to ground coordinate system $\{P\}$ is a 3D coordinate system. The origin of $\{P\}$ is coinciding with the origin of $\{C\}$, the $x-y$ plane is parallel to the ground with the x-axis pointing to the forward direction of the vehicle. The z-axis is pointing downward perpendicularly to the ground patch. A point i, $i \in \mathscr{I}$, in $\{P^k\}$ is denoted as ${}^{P}\mathbf{X}_i^k$.

4 Software System Diagram

The system diagram of the visual odometry software is shown in Fig. 3. The system takes the camera images, altimeter reading, orientation from the INS, and computes the translation and inclination of the ground. We will show that the translation estimation is insensitive to the noise in roll and pitch angles, but sensitive to yaw noise. Hence the visual odometry is particulary designed to take only the roll and pitch

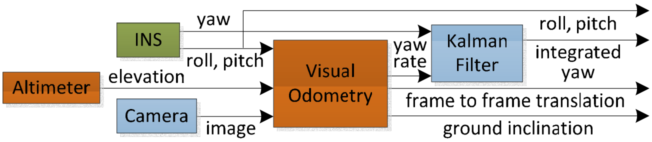


Fig. 3 Visual odometry software system diagram

angles from the INS, and estimate the yaw angle by itself. Behind the visual odometry block, we implement a Kalman filter that integrates the yaw angle. The Kalman filter helps reduce the noise amount by taking the visual odometry estimate in the prediction steps and the INS measurement in the update steps. The integrated yaw angle can be used to register the translation in the world.

5 Visual Odometry Method

5.1 *Method Intuition*

Fig. 4 presents the key idea for the visual odometry method. We use two parallel coordinate systems at frames $k-1$ and k, respectively. Let $\{V^{k-1}\}$ be a coordinate system with its origin coinciding with the origin of $\{C^{k-1}\}$, and $\{V^k\}$ a coordinate system with its origin coinciding with that of $\{C^k\}$. Initially, $\{V^{k-1}\}$ and $\{V^k\}$ are rotated to the horizontal position as shown in Fig. 4(a), using orientation from the INS. Then, through nonlinear iterations, $\{V^{k-1}\}$ and $\{V^k\}$ are rotated to be parallel to $\{P^k\}$, the coordinate system parallel to the ground patch at frame k, as shown in Fig. 4(b). During the nonlinear iterations, $\{V^{k-1}\}$ and $\{V^k\}$ are kept parallel, and the features at both frames are projected into $\{V^{k-1}\}$ and $\{V^k\}$. The projected features are used to compute the vehicle frame to frame motion.

5.2 *Mathematical Derivation*

In this section, we present the mathematical derivation of the proposed method. The complete visual odometry algorithm is presented in the next section. From the pin-hole camera model, we have the following relationship between $\{I^l\}$ and $\{C^l\}$, $l \in \{k-1, k\}$,

$$\alpha \, {}^{I}\boldsymbol{X}_i^l = \mathbf{K} \, {}^{C}\boldsymbol{X}_i^l, \tag{1}$$

where α is a scale factor, and $\mathbf{K}$ is the camera intrinsic matrix, which is known from pre-calibration [13].

Let ${}^{C}_{V}\theta^l$ and ${}^{C}_{V}\psi^l$, $l \in \{k-1, k\}$, be the roll and pitch angles from $\{V^l\}$ to $\{C^l\}$. The relationship between $\{C^l\}$ and $\{V^l\}$ is expressed as

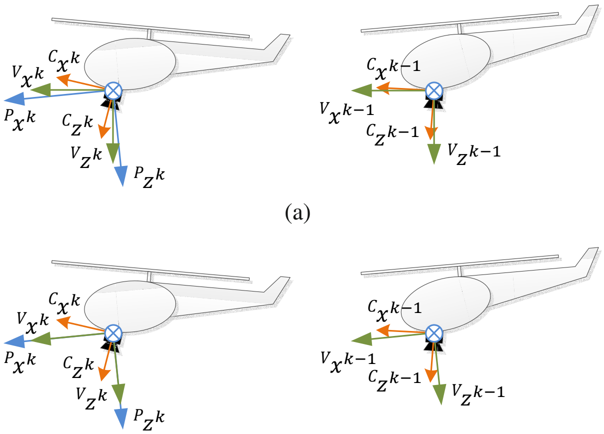
$${}^{C}\boldsymbol{X}_i^l = \mathbf{R}_x({}^{C}_{V}\theta^l)\mathbf{R}_y({}^{C}_{V}\psi^l) \, {}^{V}\boldsymbol{X}_i^l, \tag{2}$$

where $\mathbf{R}_x(\cdot)$ and $\mathbf{R}_y(\cdot)$ are rotation matrices around the x- and y- axes, respectively.

Let ${}^{V}\tilde{\boldsymbol{X}}_i^l$, $l \in \{k-1, k\}$, be the normalized term of ${}^{V}\boldsymbol{X}_i^l$, such that

$${}^{V}\tilde{\boldsymbol{X}}_i^l = {}^{V}\boldsymbol{X}_i^l / {}^{V}z_i^l, \tag{3}$$

where ${}^{V}z_i^l$ is the 3rd entry of ${}^{V}\boldsymbol{X}_i^l$. ${}^{V}\tilde{\boldsymbol{X}}_i^l$ can be computed by substituting (2) into (1) and scaling ${}^{V}\boldsymbol{X}_i^l$ such that the 3rd entry becomes one.



(a)

(b)

Fig. 4 Illustration of coordinate systems $\{V_{k-1}\}$ and $\{V_k\}$. As indicated by the green colored arrows, $\{V_{k-1}\}$ (in the right column) and $\{V_k\}$ (in the left column) are two parallel coordinate systems at frames $k-1$ and k, respectively. $\{V_{k-1}\}$ and $\{V_k\}$ are initialized at the horizontal position as illustrated in (a), using orientation from the INS. Then, through a nonlinear optimization, $\{V_{k-1}\}$ and $\{V_k\}$ are rotated to be parallel to $\{P_k\}$, as shown in (b). The figure only represents a planar case, while $\{V_{k-1}\}$ and $\{V_k\}$ have roll and pitch DOFs with respect to $\{C_{k-1}\}$ and $\{C_k\}$.

Let Δ_x^k, Δ_y^k, and Δ_z^k be the vehicle translation in the x-, y- and z- directions between frames $k-1$ and k, and let Δ_ϕ^k be the corresponding yaw rotation between the two frames. From the vehicle motion, we can establish a relationship between $\{V^{k-1}\}$ and $\{V^k\}$,

$$ {}^{V}\boldsymbol{X}_i^k = \mathbf{R}_z(\Delta_\phi^k)({}^{V}\boldsymbol{X}_i^{k-1} - \left[\Delta_x^k,\ \Delta_y^k,\ \Delta_z^k\right]^T), \tag{4}$$

where $\mathbf{R}_z(\cdot)$ is the rotation matrix around the z- axis.

Substituting (3) into (4) for both frames $k-1$ and k, and since Δ_ϕ^k is a small angle in practice, we perform linearization to obtain the following equations,

$$ s\,{}^{V}\tilde{x}_i^{k-1} = {}^{V}\tilde{x}_i^{k} - {}^{V}\tilde{y}_i^{k}\Delta_\phi^k + \Delta_x^k / {}^{V}z_i^k, \tag{5}$$

$$ s\,{}^{V}\tilde{y}_i^{k-1} = {}^{V}\tilde{y}_i^{k} + {}^{V}\tilde{x}_i^{k}\Delta_\phi^k + \Delta_y^k / {}^{V}z_i^k, \tag{6}$$

$$ s = 1 - \Delta_z^k / {}^{V}z_i^k, \tag{7}$$

where ${}^{V}\tilde{x}_i^l$ and ${}^{V}\tilde{y}_i^l$, $l \in \{k-1, k\}$, are the 1st and the 2nd entries of ${}^{V}\tilde{\boldsymbol{X}}_i^l$, respectively, ${}^{V}z_i^k$ is the 3rd entry of ${}^{V}\boldsymbol{X}_i^k$, and s is a scale factor.

Recall that h^k is the distance from the vehicle to the ground patch, and the ground patch has roll and pitch DOFs round point A in Fig. 2. Let ${}^{V}_{P}\theta^k$ and ${}^{V}_{P}\psi^k$ be the roll and pitch angles from $\{P^k\}$ to $\{V^k\}$. In (5)–(7), the feature depth ${}^{V}z_i^k$ can be computed from a simple geometry relationship,

$$ {}^{V}z_i^k = h^k(1-({}^{V}\tilde{x}_i^k - {}^{C}_{V}\psi^k){}^{V}_{P}\psi^k - ({}^{V}\tilde{y}_i^k - {}^{C}_{V}\theta^k){}^{C}_{V}\theta^k). \tag{8} $$

Combining (5), (6) and (8), we have,

$$ \frac{a\Delta_x^k + b\Delta_y^k}{c + d\,{}^{V}_{P}\psi^k + e\,{}^{V}_{P}\theta^k} + f\Delta_\psi^k + g = 0. \tag{9} $$

where

$$ a = {}^{V}\tilde{y}_i^{k-1},\ b = -{}^{V}\tilde{x}_i^{k-1},\ c = h^k,\ d = -h^k({}^{V}\tilde{x}_i^k - {}^{C}_{V}\psi^k), \tag{10} $$

$$ e = -h_k(\tilde{y}^V_{(k,i)} - {}^{C}_{V}\theta^k),\ f = -{}^{V}\tilde{x}_i^k\,{}^{V}\tilde{x}_i^{k-1} - {}^{V}\tilde{y}_i^k\,{}^{V}\tilde{y}_i^{k-1}, \tag{11} $$

$$ g = {}^{V}\tilde{x}_i^k\,{}^{V}\tilde{y}_i^{k-1} - {}^{V}\tilde{y}_i^k\,{}^{V}\tilde{x}_i^{k-1}. \tag{12} $$

In (9), we have totally five unknowns, Δ_x^k, Δ_y^k, Δ_ϕ^k, ${}^{V}_{P}\theta^k$, ${}^{V}_{P}\psi^k$. The function can be solved using five or more feature points with a nonlinear method. However, in certain cases, we can consider ${}^{V}_{P}\theta^k$ and ${}^{V}_{P}\psi^k$ as known variables such that (9) can be solved linearly with three or more feature points. Next, we will provide a linear and a nonlinear way to solve the function. Both methods will be useful for the visual odometry algorithm presented in the next section.

5.2.1 Linear Method

Set ${}^{V}_{P}\theta^k$ and ${}^{V}_{P}\psi^k$ in (9) as known variables and treat Δ_x^k, Δ_y^k, Δ_ϕ^k as unknowns. For m, $m \geq 3$, feature points, stack (9) for each feature. This will give us a linear function in the form of

$$ \mathbf{A}\boldsymbol{X}_L = \boldsymbol{b}, \tag{13} $$

where $\mathbf{A}$ is a $m \times 3$ matrix, $\boldsymbol{b}$ is a $m \times 1$ vector, and $\boldsymbol{X}_L$ contains the unknowns, $\boldsymbol{X}_L = [\Delta_x^k,\ \Delta_y^k,\ \Delta_\phi^k]^T$. Solving (13) with the singular value decomposition method [13], we can recover $\boldsymbol{X}_L$.

5.2.2 Nonlinear Method

For m, $m \geq 5$, feature points, stack (9) for each feature and reorganize the function into the following form,

$$ \boldsymbol{f}(\boldsymbol{X}_N) = \boldsymbol{b}, \tag{14} $$

where $\boldsymbol{f}$ is a nonlinear function with 5 inputs and m outputs. $\boldsymbol{b}$ is a $m \times 1$ vector, and $\boldsymbol{X}_N$ contains the unknowns, $\boldsymbol{X}_N = [\Delta_x^k,\ \Delta_y^k,\ \Delta_\phi^k,\ {}^{V}_{P}\theta^k,\ {}^{V}_{P}\psi^k]^T$. Compute the Jacobian matrix of $\boldsymbol{f}$ with respect to $\boldsymbol{X}_N$, denoted as $\mathbf{J}$, where $\mathbf{J} = \partial f / \partial \boldsymbol{X}_N$. (14) can be solved through nonlinear iterations using the Levenberg-Marquardt method [13],

$$ \boldsymbol{X}_N \leftarrow \boldsymbol{X}_N + (\mathbf{J}^T\mathbf{J} + \lambda \mathrm{diag}(\mathbf{J}^{\mathrm{T}}\mathbf{J}))^{-1}\mathbf{J}^{\mathrm{T}}(\boldsymbol{b} - \boldsymbol{f}(\boldsymbol{X}_{\mathrm{N}})), \tag{15} $$

where λ is a scale factor.

5.3 Algorithm

Algorithm 1 presents the proposed visual odometry algorithm. The algorithm first initializes using readings from the INS. Let θ_{INS}^{k-1} and ψ_{INS}^{k-1} be the roll and pitch angles of the vehicle at frame $k-1$, measured by the INS, and let θ_{INS}^{k} and ψ_{INS}^{k} be the corresponding angles at frame k. On lines 4-5, we rotate $\{V^{k-1}\}$ and $\{V^{k}\}$ to the horizontal position using the INS orientation, and we project the feature points from $\{C^{k-1}\}$ and $\{C^{k}\}$ to $\{V^{k-1}\}$ and $\{V^{k}\}$, respectively. From now, $\{V^{k-1}\}$ and $\{V^{k}\}$ become parallel coordinate systems. Then, on line 6, we set ${}^{V}_{P}\theta_l, {}^{V}_{P}\psi_l \leftarrow 0$ and compute Δ_x^k, Δ_y^k, Δ_ϕ^k linearly. The result is used as initialization for the nonlinear optimization between lines 7-20. The 5 unknowns Δ_x^k, Δ_y^k, Δ_ϕ^k, ${}^{C}_{V}\psi^k$, ${}^{C}_{V}\theta^k$ are updated on line 10. On lines 11-12, $\{V^{k-1}\}$ and $\{V^{k}\}$ are rotated to the newly updated orientation with the features reprojected into $\{V^{k-1}\}$ and $\{V^{k}\}$. The iterations finish if convergence is found or the maximum iteration number is met.

Algorithm 1. Translation Estimation

```
1  input : ${}^{I}\boldsymbol{x}_i^{k-1}$, ${}^{I}\boldsymbol{x}_i^{k}$, $i \in \mathscr{I}$, $\theta_{INS}^{k-1}$, $\psi_{INS}^{k-1}$, $\theta_{INS}^{k}$, $\psi_{INS}^{k}$, $h^k$
2  output : $\Delta_x^k$, $\Delta_y^k$, $\Delta_z^k$
3  begin
4      Rotate $\{V^l\}$ to the horizontal position by ${}^{C}_{V}\theta^l \leftarrow \theta_{INS}^{l}$, ${}^{C}_{V}\psi^l \leftarrow \psi_{INS}^{l}$, $l \in \{k-1, k\}$;
5      Compute ${}^{V}\tilde{\boldsymbol{X}}_i^{k-1}$, ${}^{V}\tilde{\boldsymbol{X}}_i^{k}$ for $i \in \mathscr{I}$ based on (1-3);
6      Use $i \in \mathscr{I}$ to compute $\Delta_x^k$, $\Delta_y^k$, $\Delta_\phi^k$ linearly by setting ${}^{V}_{P}\theta^k, {}^{V}_{P}\psi^k \leftarrow 0$ based on (13);
7      for a number of iterations do
8          Compute image reprojection error (IRE) for $i \in \mathscr{I}$, then compute a weight for
           $i \in \mathscr{I}$ using the IREs;
9          for a number of iterations do
10             Use $i \in \mathscr{I}$ to update $\Delta_x^k$, $\Delta_y^k$, $\Delta_\phi^k$, ${}^{V}_{P}\theta^k$, ${}^{V}_{P}\psi^k$ for one iteration based on (15);
11             Rotate $\{V^l\}$ by ${}^{C}_{V}\theta^l \leftarrow {}^{C}_{V}\theta^l + {}^{V}_{P}\theta^k$ and ${}^{C}_{V}\psi^l \leftarrow {}^{C}_{V}\psi^l + {}^{V}_{P}\psi^k$, $l \in \{k-1, k\}$,
               then ${}^{V}_{P}\theta^k, {}^{V}_{P}\psi^k \leftarrow 0$;
12             Project ${}^{V}\tilde{\boldsymbol{X}}_i^{k-1}$, ${}^{V}\tilde{\boldsymbol{X}}_i^{k}$, $i \in \mathscr{I}$, $\Delta_x^k$, $\Delta_y^k$, $\Delta_\phi^k$ to the newly rotated $\{V^{k-1}\}$ and
               $\{V^k\}$;
13             if the nonlinear optimization converges then
14                 Break;
15             end
16         end
17         if the robust fitting converges then
18             Break;
19         end
20     end
21     Compute $\Delta_z^k$ based on (7) as the weighted average of the features;
22     Return $\Delta_x^k$, $\Delta_y^k$, $\Delta_z^k$;
23 end
```

The algorithm is adapted to a robust fitting [14] to ensure robustness against features with large tracking errors. The algorithm assigns a weight for each feature (line 8), based on the image reprojection error (IRE). The features with larger IREs are assigned with smaller weights, while features with the IREs larger than a threshold are considered as outliers and assigned with zero weights. Note that the robust fitting only solves the x- and y- translation of the vehicle, Δ_x^k, Δ_y^k. To obtain the z-translation, Δ_z^k, we use (7) with the selected inlier features from the robust fitting (line 21). Δ_z^k is computed as the weighted average of the inlier features using the same weights generated by the robust fitting on line 8.

6 Analysis of Error Propagation

Here we show how the errors are propagated onto the vehicle motion estimation. We care about how the errors accumulate in the horizontal position estimate because vertical position drift can be largely corrected by reading of the altimeter. We will derive the upper bound of the accumulated position drift.

We start with the INS roll and pitch angles. Recall that θ_{INS}^l, ψ_{INS}^l, $l \in \{k-1,k\}$ are the roll and pitch inclination angles of the vehicle measured by the INS at fame l. Let us define $\hat{\theta}_{INS}^l$ and $\hat{\psi}_{INS}^l$ as their measurement values containing errors. Let e_θ^l and e_ψ^l be the corresponding errors, we have $e_\theta^l = \hat{\theta}_{INS}^l - \theta_{INS}^l$ and $e_\psi^l = \hat{\psi}_{INS}^l - \psi_{INS}^l$. By examining each step in Algorithm 1, we find that e_θ^l and e_ψ^l are introduced into the algorithm at the initialization step (line 4). With the INS measurements, the coordinate systems $\{V^{k-1}\}$ and $\{V^k\}$ are intended to be rotated to the horizontal position. However, because of e_ψ^l and e_θ^l, $\{V^{k-1}\}$ and $\{V^k\}$ are not exactly aligned with the horizontal position. The roll and pitch difference between $\{V^{k-1}\}$ and $\{V^k\}$ are $e_\theta^{k-1} - e_\theta^k$ and $e_\psi^{k-1} - e_\psi^k$, respectively. This angle difference is kept through the algorithm since the two coordinate systems are rotated simultaneously by the same angle. In the end, $\{V^k\}$ is rotated to be parallel to $\{P^k\}$, or the ground patch at frame k, and $\{V^{k-1}\}$ keeps an angular error to $\{P^k\}$. Let ${}_V^C\hat{\theta}^{k-1}$ and ${}_V^C\hat{\psi}^{k-1}$ be the measurement values of the roll and pitch angles from $\{V^{k-1}\}$ to $\{C^{k-1}\}$, ${}_V^C\theta^{k-1}$ and ${}_V^C\psi^{k-1}$, we can compute

$$ {}_V^C\hat{\theta}^{k-1} = {}_V^C\theta^{k-1} + e_\theta^{k-1} - e_\theta^k,\ {}_V^C\hat{\psi}^{k-1} = {}_V^C\psi^{k-1} + e_\psi^{k-1} - e_\psi^k. \tag{16} $$

The errors in ${}_V^C\hat{\theta}^{k-1}$ and ${}_V^C\hat{\psi}^{k-1}$ propagate through (2). With the errors introduced, we rewrite the equation as follows,

$$ {}^C\boldsymbol{X}_i^{k-1} = \mathbf{R}_x({}_V^C\theta^{k-1} + e_\theta^{k-1} - e_\theta^k)\mathbf{R}_y({}_V^C\psi^{k-1} + e_\psi^{k-1} - e_\psi^k)\ {}^V\boldsymbol{X}_i^{k-1}. \tag{17} $$

Correspondingly, we derive (9) again containing the errors,

$$a\left(\frac{\Delta_x^k}{c+d\,{}^V_P\psi^k+e\,{}^V_P\theta^k}+e_\psi^{k-1}-e_\psi^k\right)+b\left(\frac{\Delta_y^k}{c+d\,{}^V_P\psi^k+e\,{}^V_P\theta^k}+e_\theta^{k-1}-e_\theta^k\right)$$
$$+f\Delta_\phi^k+g=0, \qquad (18)$$

where a, b, c, d, e, f, and g are defined in (10)-(12).

Now, we compare (18) with (9). Note that after the nonlinear optimization in Algorithm 1 converges, we have ${}^V_P\psi^k, t_k^{PV} \to 0$. Under this condition, if we define $\hat{\Delta}_x^k = \Delta_x^k + (e_\psi^{k-1} - e_\psi^k)h^k$ and $\hat{\Delta}_y^k = \Delta_y^k + (e_\theta^{k-1} - e_\theta^k)h^k$, and substitute the terms into (18), (18) becomes essentially the same as (9) except that Δ_x^k and Δ_y^k are replaced by $\hat{\Delta}_x^k$ and $\hat{\Delta}_y^k$. Examining the expressions of $\hat{\Delta}_x^k$ and $\hat{\Delta}_y^k$, we find that the terms are invariant with respect to different features. This indicates that if we use $\hat{\Delta}_x^k$ and $\hat{\Delta}_y^k$ as the measurement values of Δ_x^k and Δ_y^k for the case that contains the errors, (14) is satisfied for each of its m rows. Define e_x^k and e_y^k as the estimation errors corresponding to Δ_x^k and Δ_y^k, we have

$$e_x^k = \hat{\Delta}_x^k - \Delta_x^k = (e_\psi^{k-1} - e_\psi^k)h^k,\ e_y^k = \hat{\Delta}_y^k - \Delta_y^k = (e_\theta^{k-1} - e_\theta^k)h^k. \qquad (19)$$

We want to analyze how the errors accumulate over time. Let us define e_x and e_y as the accumulated errors of e_x^k and e_y^k respectively, from frames 1 to n, $n \in \mathbb{Z}^+$,

$$e_x = \sum_{k=1}^{n} e_x^k,\ e_y = \sum_{k=1}^{n} e_y^k. \qquad (20)$$

We want to find the upper bounds of $|e_x|$ and $|e_y|$. Let us define E_θ and E_ψ as the upper bounds of the roll and pitch errors from the INS, where $|e_\theta^k| \leq E_\theta$ and $|e_\psi^k| \leq E_\psi$, $k \in \{1, 2, ..., n\}$. Substituting (19) into (20), we can derive

$$\begin{aligned}
e_x &= \sum_{k=2}^{n}(e_\psi^{k-1} - e_\psi^k)h^k = \sum_{k=2}^{n}(e_\psi^{k-1} - e_\psi^k)h^{(2)} + \sum_{k=3}^{n}(e_\psi^{k-1} - e_\psi^k)(h^k - h^{(2)}) \\
&= \sum_{k=2}^{n}(e_\psi^{k-1} - e_\psi^k)h^{(2)} + \sum_{j=3}^{n}\sum_{k=j}^{n}(e_\psi^{k-1} - e_\psi^k)(h^j - h^{j-1}) \\
&= (e_\psi^1 - e_\psi^n)h^{(2)} + \sum_{j=3}^{n}(e_\psi^{j-1} - e_\psi^n)(h^j - h^{j-1}). \qquad (21)
\end{aligned}$$

Here, since $|e_j^p - e_n^p| \leq |e_j^p| + |e_n^p| \leq 2E_\psi$, $j \in \{1, 2, ..., n\}$, we can find the upper bound of $|e_x|$ as

$$|e_x| \leq 2E_\psi(h^{(2)} + \sum_{j=3}^{n}|h^j - h^{j-1}|). \qquad (22)$$

Similarly, we can derive the upper bound of $|e^y|$ as

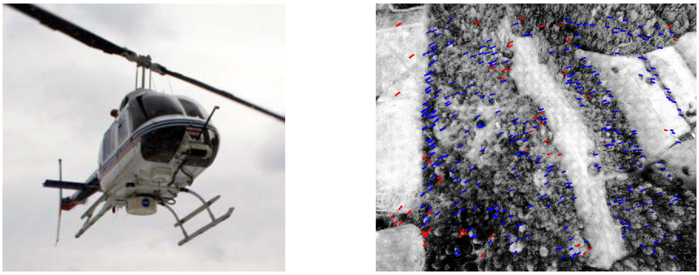
$$|e_y| \leq 2E_\theta(h^{(2)} + \sum_{j=3}^{n}|h^j - h^{j-1}|). \qquad (23)$$

Eq. (22) and (23) indicate that the accumulated translation error introduced by the roll and pitch noise from the INS is only related to the altitude change of the vehicle, regardless of the flying distance. In a special case that the vehicle keeps a constant height above the ground during a flight, $|e_x|$ and $|e_y|$ are bounded by two constants, $|e_x| < 2E_\psi h$ and $|e_y| < 2E_\theta h$, where h is the constant height of the flight. In another case that the vehicle takes off from the ground, h^k starts from zero. The upper bounds of $|e_x|$ and $|e_y|$ are proportional to the accumulated altitude change during the flight, $|e_x| < 2E_\psi \sum_{j=3}^{n} |h^j - h^{j-1}|$ and $|e_y| < 2E_\theta \sum_{j=3}^{n} |h^j - h^{j-1}|$.

Further, we find that the upper bound of the position drift introduced by the yaw angle and altimeter noise is proportional to the flying distance. For space issue, we eliminate the proof. The conclusion can be explained intuitively that if the yaw angle is off, the position estimate will constantly drift to the left or right side. Similarly, noise in the altimeter reading will result in under or overly estimated translation scale. Note that the proposed visual odometry also estimates the yaw angle of the aircraft. This is particularly proposed and allows us to integrate the yaw angle from the INS and visual odometry in a Kalman filter. The integrated yaw angle has a lower amount of noise and is used to register the translation in the world. Also, we use a high quality laser altimeter to reduce the drift in scale.

7 Experiments

We obtain image sequences from a downward pointing camera mounted to a full-scale helicopter (Fig. 5(a)). The camera resolution is 612 × 512 pixels with the horizontal field of view of 75°. The camera frame rate is set at 14Hz. The helicopter is also equipped with a laser altimeter and a GPS/INS. The orientation measurement from the GPS/INS is used by the visual odometry, while the position reading is used as the ground truth for comparison purposes.



(a) (b)

Fig. 5 (a) Helicopter used in the visual odometry tests. A downward pointing camera is mounted to the front of the helicopter. (b) Tracked features. A number of 450 feature points are tracked between image frames. The red colored segments are outlier features assigned with zero weights in Algorithm 1. The blue colored segments are inlier features used in the motion estimation.

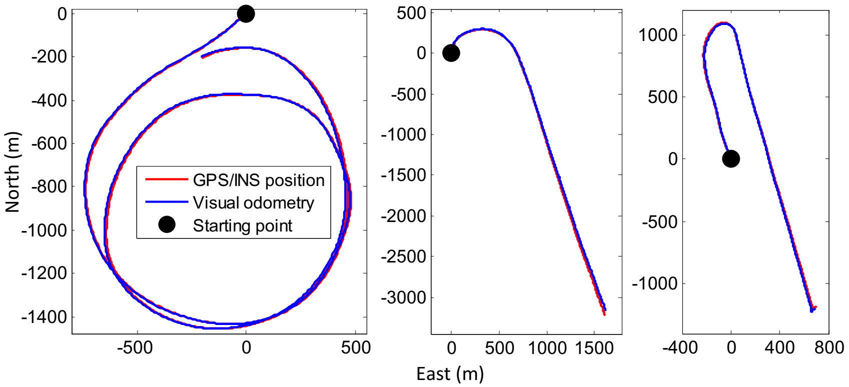


Fig. 6 Visual odometry outputs (blue) compared to GPS/INS ground truth (red) for three tests. The subfigures from left to right correspond to Test 1-3 in Table 1. The overall distance of the three tests is 16km, and the average error at the end is 0.57% of the flying distance.

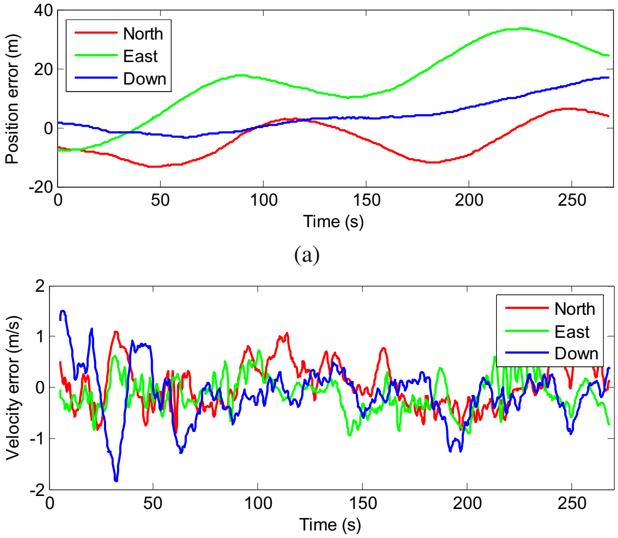
Table 1 Configuration and accuracy of the three tests in Fig. 6 (from left to right)

Test No.	Flying Distance	Altitude	Flying Speed	Accuracy
1	7800m	300m	30m/s	0.39%
2	3700m	150m	20m/s	0.73%
3	4500m	200m	20m/s	0.78%

The algorithm selects a number of 450 Harris corners [13] using the openCV library, and tracks the feature points between image frames using the Kanade Lucas Tomasi method [15]. To evenly distribute the feature points in the images, we separate the images into 9 (3×3) identical subregions. Each subregion provides 50 features. Fig. 5 shows an example of the tracked features. The red colored segments are outliers assigned with zero weights in Algorithm 1, and the blue colored segments are inliers used in the motion estimation.

Fig. 6 shows results of the proposed method in three flight tests. The blue colored curves are visual odometry outputs, the red colored curves are ground truth provided by the GPS/INS, and the black colored dots are starting points. More detailed configurations and accuracy comparison of the three tests are in Table 1. Tests 1-3 correspond to the subfigures in Fig. 6 from left to right. The overall flying distance is 16km, and the average error at the end is 0.57% of the flying distance.

Fig. 7 presents position and velocity errors for Test 1 (the left subfigure in Fig. 6). Fig. 7(a) shows the accumulated position drift through the test. Fig. 7(b) gives the



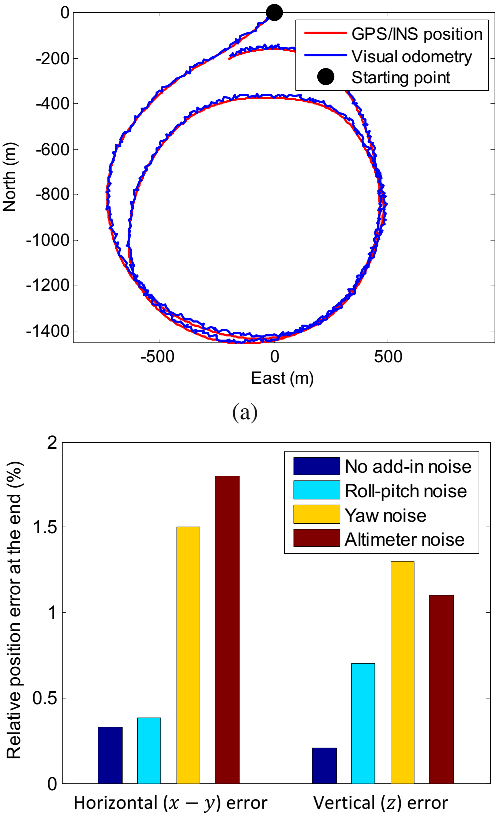
(a)

(b)

Fig. 7 (a) Accumulated position drift and (b) velocity errors in Test 1 (the left subfigure in Fig. 6)

absolute velocity errors. Most of the velocity errors are smaller than 1m/s, while the average speed of the helicopter is 30m/s during the test.

To inspect how sensor noise affects the motion estimation, we add artificial noise to the INS and altimeter readings. In Fig. 8(a), $\sigma = 3^\circ$ Gaussian noise is added to the roll and pitch angles from the INS. The corresponding visual odometry output becomes locally noisy but little drift happens in global scale. This confirms to the theory proposed in this paper that the motion estimation is insensitive to the roll and pitch angle noise. Fig. 8(b) presents a more complete comparison with respect to different add-in noise. Note that with roll and pitch angle noise, we only prove upper bound of the position drift on horizontal plane but not in vertical direction. The light blue colored bars indicate that position drift does accumulate in vertical direction. A possible solution of fixing the drift is using elevation of the vehicle measured by the altimeter. As expected, the position drift from yaw angle noise and altimeter noise accumulates overtime (yellow and brown colored bars).



(a)

(b)

Fig. 8 (a) Blue: visual odometry output with artificial add-in noise. The noise is added to the roll and pitch angles from the INS. Red: GPS/INS ground truth (b) Relative errors with respect to different add-in noise. The noise is added to the roll and pitch angles from the INS, yaw angle, and altimeter reading, respectively. The angle noise follows $\sigma = 3°$ Gaussian distribution, and the noise for the altimeter is 3% (σ value) of the elevation with Gaussian distribution.

8 Conclusion and Future Work

When using INS orientation readings in solving a visual odometry problem, the noise contained in the INS measurements can affect the vehicle motion estimation, causing the position estimate to drift. The proposed method reduces the accumulation of the position estimation error in two ways. First, we assume the imaged ground is locally flat and online estimate the inclination angles, and second, we re-project features with their depth direction perpendicular to the ground. This way, the translation error from the INS orientation noise cancels itself partially, resulting in a slow position drift. The method is tested on a full-scale helicopter for 16km of flying experiments. The results indicate a relative error of less then 1%.

References

1. Nister, D., Naroditsky, O., Bergen, J.: Visual odometry for ground vechicle applications. Journal of Field Robotics 23(1), 3–20 (2006)
2. Maimone, M., Cheng, Y., Matthies, L.: Two years of visual odometry on the mars exploration rovers. Journal of Field Robotics 24(2), 169–186 (2007)
3. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. The MIT Press, Cambridge (2005)
4. Amidi, O., Kanade, T., Fujita, K.: A visual odometer for autonomous helicopter flight. Robotics and Autonomous Systems 28(2-3), 185–193 (1999)
5. Nuske, S., Roberts, J., Wyeth, G.: Robust outdoor visual localization using a three-dimensional-edge map. Journal of Field Robotics 26(9), 728–756 (2009)
6. Kelly, J., Saripalli, S., Kukhatme, G.: Combined visual and inertial navigation for an unmanned aerial vehicle. In: Laugier, C., Siegwart, R. (eds.) Field and Service Robotics. STAR, vol. 42, pp. 255–264. Springer, Heidelberg (2008)
7. Konolige, K., Agrawal, M., Sol, J.: Large-scale visual odometry for rough terrain. Robotics Research 66, 201–212 (2011)
8. Klein, G., Murray, D.: Parallel tracking amd mapping for small AR workspaces. In: Proc. of the International Symposium on Mixed and Augmented Reality, Nara, Japan, pp. 1–10 (November 2007)
9. Weiss, S.: Vision based navigation for micro helicopters. Ph.D. dissertation, ETH Zurich (2012)
10. Artieda, J., Sebastian, J., Campoy, P., et al.: Viusal 3-d SLAM from UAVs. Journal of Intelligent and Robotic Systems 55 (2009)
11. Conte, G., Doherty, P.: Vision-based unmanned aerial vehicle navigation using geo-referenced information. EURASIP Journal on Advances in Signal Processing 2009, 10 (2009)
12. Caballero, F., Merino, L., Ferruz, J., Ollero, A.: Vision-based odometry and SLAM for medium and high altitude flying UAVs. Journal of Intelligent and Robotic Systems 54(1-3), 137–161 (2009)
13. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, New York (2004)
14. Andersen, R.: Modern methods for robust regression. Sage University Paper Series on Quantitative Applications in the Social Sciences (2008)
15. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of Imaging Understanding Workshop, pp. 121–130 (1981)

Part IV
Control

An Attitude Controller for Small Scale Rockets

Florian Kehl, Ankur M. Mehta*, and Kristofer S.J. Pister

Abstract. As technology has advanced, electronic components and systems have become smaller and more powerful. A similar trend holds for space systems, and satellites are no exception. As payloads become smaller, so too can the launch vehicles designed to carry them into orbital trajectories. An energy analysis shows that a rocket system with as low as tens of kg of fuel can be sufficient to deliver a 10g payload into orbit given a sufficiently low mass autonomous rocket flight control system. To develop this, the GINA board, a 2g sensor-laden wireless-enabled microprocessor system, was mounted on a custom actuated rocket system and programmed for inertial flight control. Ground and flight tests demonstrated accurate dead reckoning state estimation along with successful open loop actuator control. Further experiments showed the capabilities of the control system at closed loop feedback control. The results presented in this paper demonstrate the feasibility of a sufficiently low mass flight controller, paving the way for a small scale rocket system to deliver a 10g attosatellite into low Earth orbit (LEO).

1 Introduction

The environment above Earth's atmosphere and beyond is primarily dominated by large one-off spacecraft. Only recently has there been analysis of potential deployments of distributed networks in space (such as in [1–3]). Distributed satellite systems comprising femto- or attosatellites (10-100 grams and 1-10 grams respectively [4]) can enable new atmospheric and astronomical scientific research [5] as well as address wireless sensor network (WSN) research in the absence of notable interference from ground based sources and physical obstacles. As the availability and functionality of electronics go up and the cost goes down, the required hardware

Florian Kehl · Ankur M. Mehta · Kristofer S.J. Pister
UC Berkeley, Berkeley, CA, USA
e-mail: {kehl,mehtank,pister}@eecs.berkeley.edu

* Corresponding author.

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_14,

becomes smaller, cheaper, and more accessible. While previous small satellite research has focused on systems on the order of kilograms, sensor nodes have shrunk to where a 10 gram system is sufficiently powerful for many purposes.

This paper addresses the issue of deploying such sensor nodes into a low Earth orbit (LEO) for applications in space-based WSNs. In particular, this paper begins to examine a small scale rocket-based solution for delivering a 10 gram attosatellite payload to a desired orbital trajectory. The additional difficulty in miniaturizing a satellite deployment system is offset by the drastically lower cost and risk factors compared to current large-scale launch options.

Ultimately, a launch vehicle (LV) should be of comparable scale and cost to the payload mote being deployed [6]. A full launch solution will also require careful rocket and propellant design; this paper focuses on miniaturizing a control system as described in [7] to be used to guide the LV into a desired trajectory. The hardware developed here is applicable as a final stage in orbital insertion – a rocket was designed and built using low cost, off-the-shelf components to estimate and control system attitude.

An overview of rocket systems and the difficulties in their miniaturization is presented in section 2. The specific hardware designed in this work in described in section 3, with an explanation of the experimental setup and some testing results in section 4. Finally, section 5 offers some conclusions and avenues for future research.

2 Background

2.1 Energy to Low Earth Orbit

Energy considerations drive the mechanical rocket design, constrained by physical properties of available materials.

2.1.1 General Rocket Equations

A common metric quantifying the energy required to implement an orbital maneuver is the scalar Δv or delta-vee. This energy must be provided by the propulsion system. In the case of a LV moving from rest on the surface of the Earth to LEO, the required Δv_{leo} can be decomposed as follows [8,9]:

$$\Delta v_{leo} = v_o + \Delta v_d + \Delta v_g + \Delta v_c + \Delta v_{atm} - v_{rot}, \tag{1}$$

where

- v_o is the orbital velocity,
- Δv_d represents the energy lost to drag,
- Δv_g represents the additional gravitational potential energy,
- Δv_c represents energy needed to effect trajectory control,
- Δv_{atm} represents the energy lost due to engine inefficiency in atmosphere,
- v_{rot} is the velocity of the launch platform due to earth's rotation.

The orbital velocity v_o is the speed required to maintain the orbital trajectory. For a circular orbit,

$$v_o = \sqrt{\frac{G \cdot M_e}{(R_e + h)}}, \tag{2}$$

with gravitational constant G, earth's mass M_e and radius R_e, and orbital altitude h. For a satellite in LEO at an altitude of 200 km, this gives $v_o = 7.8$ km/s. The other Δv losses in equation 1 total 1.5 - 2 km/s, yielding a total Δv_{leo} of 9.5 - 10 km/s for a ground launched LV to reach LEO [8].

The total Δv generated by a propulsion system over the duration of a maneuver can be calculated from the time history of its instantaneous thrust ($|F|$) and LV total mass (m):

$$\Delta v = \int_t \frac{|F|}{m} dt. \tag{3}$$

Evaluating this integral for a basic combustion chamber rocket design yields the ideal rocket equation

$$\Delta v = v_e \cdot \ln\left(\frac{m_i}{m_f}\right), \tag{4}$$

where the exhaust velocity v_e is a property of the specific fuel/rocket system.

For a given v_e, then, the Δv of a rocket stage can simply be calculated from that stage's mass ratio m_i/m_f. In practice, the final mass m_f after burnout consists of the structural mass of the rocket along with the payload (which includes higher stages), while the initial mass m_i also includes the mass of the fuel. The total Δv of a multi-stage rocket is the sum of of the Δv's of each stage calculated independently.

2.1.2 Adaptation for Minirockets

The drag force experienced by a rocket moving through the atmosphere is given by:

$$F_d = \frac{1}{2}\rho C_d v^2 A, \tag{5}$$

where ρ is the density of the surrounding air, C_d is the drag coefficient, v is the velocity of the rocket relative to the air, and A is the rocket's cross sectional area. This results in a penalty:

$$\Delta v_d = \int \frac{F_d}{m} dt, \tag{6}$$

that scales inversely with length.

As a result, small scale rockets must provide a higher total Δv. Given the ideal rocket equation (4), this means that it is especially important to minimize the mass of the LV, including the electronic flight controller. Thus, the primary specification for the rocket system design becomes the weight metric [10].

The designed rocket system itself need not deliver the total Δv required for orbit. Though a minirocket must still be used to deliver the payload to the specific desired orbital trajectory, some portion of the path can be achieved by bootstrapping on existing aircraft launches. A conventional rocket LV, ultra-high altitude balloon (UHAB), or gun launch system can be used to deploy a system at altitude in the upper atmosphere [11, 12].

There is precedent for such piggyback style systems. The CubeSat program [13] uses spare payload capacity between stages of large scale commercial rocket launches to deploy academic 1 L, 1.33 kg "nano-satellites" into orbit. A UHAB system has been demonstrated to lift a 690 kg payload to a peak altitude of 49.4 km [14], and J.A. Van Allen had used balloons to launch rockets into the upper atmosphere extensively during the 1950s [15].

There are a number of energy advantages for high-altitude launches:

- The gravitational potential energy Δv_g need not be supplied by the rocket.
- The control effort Δv_c is reduced for a shorter trajectory. Not needing to compensate for the wind gusts present at lower altitudes further reduces Δv_c.
- The air drag losses Δv_d experienced by a LV launched above 98.5% of the atmosphere are less than 3% that of a similar system launched from the ground [16].
- The engine operates at peak performance when exhausting into near vacuum, reducing the Δv_{atm} loss caused by to a lower ratio between combustion chamber and ambient pressures [17].

2.1.3 Minirocket Design

The design of a complete miniature rocket LEO LV requires a complicated interplay of materials science, aerodynamics, and mechanical engineering [18] – it is a significant undertaking, and beyond the scope of this work. Nonetheless, first order analysis of such a system is necessary to establish the feasibility of such a system.

To keep costs, complexity and structural mass at a minimum, a solid propellant seems to be favorable for a small-scale LV. The need for pipes, valves, tanks, and insulation in liquid propellant engines would contribute to a high overall structural mass. The main disadvantages of solid-fuel propellant are the lower specific impulse I_{sp} and the lack of active throttling, though the latter could potentially be overcome by a combination of intelligent propellant grain design and control system tuning, allowing specific thrust-time characteristics [19].

A solid fuel rocket is little more than an open-ended cylinder packed with fuel (e.g. an ammonium perchlorate oxidizer in a hydroxy-terminated poly-butadiene (HTPB) binder) forming the combustion chamber with a nozzle for exhaust emission, upon which the control system and actuators are mounted. To minimize structural weight, high tensile strength carbon fiber can be used to form the combustion chamber. The physical characteristics of such a system are summarized in table 1.

An single stage to orbit (SSTO) rocket would require that at least 98% of the mass of the rocket be fuel. With required structural mass going to the rocket body, control actuators and electronics, and payload, this requirement proves impossible

Table 1 Physical parameters of a proposed solid fuel minirocket system

Parameter	Subsystem	Value	Units
Fuel exhaust velocity	Fuel	2.6	km/s
Fuel density	Fuel	1.763	g/cc
Chamber pressure	Rocket body	100	atm
Carbon fiber tensile strength	Rocket body	3	GPa
Carbon fiber density	Rocket body	1.75	g/cc
Payload mass	Payload	10	g

or infeasible to satisfy. Instead, staging can be used to distribute the total $\Delta v = 10$ km/s necessary to hit orbit between several stages. This requires a much smaller mass fraction of fuel per stage, bringing the full system down to a manageable size.

2.2 Guidance Control System

A rocket LV requires active control for both stability and guidance in order to reach LEO. A typical solid rocket motor is often dynamically unstable, as shown in figure 1. Above atmosphere, passive aerodynamic surfaces cannot provide restoring moments, and so the flight controller must enforce stability. Furthermore, there is no ballistic path to LEO ("what goes up must come down"), so active guidance is necessary to steer the LV along a specified trajectory to reach LEO. Though the lower stages of a multi-stage rocket can be replaced by an alternate carrier as described above, the final, smallest, stage of the minirocket will still need to perform the final orbital trajectory insertion. Thus, it is the design of this controller that is critical to system performance.

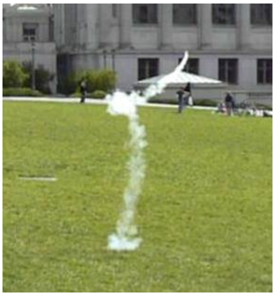


Fig. 1 In the absence of aerodynamic forces, a bare solid rocket engine spins out of control. A stabilizing controller is therefore necessary on a minirocket LV

A schematic of a rocket flight control system is diagrammed in figure 2. In order to implement feedback control, the system must first calculate the state of the LV. In

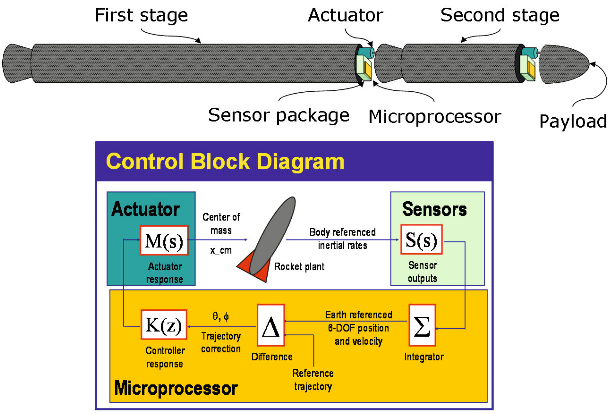


Fig. 2 A feedback control system can be used to stabilize flight and follow a trajectory. The 6DOF state is sensed by a 6 axis IMU and input to a microprocessor. The current heading and trajectory error is calculated, and a control signal is output to an actuator. The actuator positions a mass or aims the nozzle to induce the appropriate torque on the rocket.

principle, an inertial measurement unit (IMU) can measure body-referenced inertial rates, which can be integrated to provide a six degree of freedom (6DOF) state identifying the earth-referenced position and orientation [20]. In practice, additional sensors must also be filtered in to give an accurate state estimate.

Given the state estimate, then, the controller can calculate the deviation from a desired trajectory and command actuators to generate the appropriate corrections. With the final orbital insertion needing to happen above earth's atmosphere, aerodynamic control surfaces such as fins cannot be used. Instead, actuators must direct the body-referenced rocket thrust. These could take the form of a gimbaled nozzle, vanes in the exhaust stream, or an offset mass to generate the required torques for attitude control.

3 Hardware Setup

3.1 Rocket

In order to test the guidance hardware designed for small scale LVs, a model rocket based test system was developed. Miniaturization of this rocket wasn't attempted, as the focus was on the guidance subsystem comprising sensors and actuators.

Off-the-shelf model rocket components were used for the basic rocket structure, namely cardboard tubes, polystyrene and balsa wood. As usual in model rocketry, the rocket contained a parachute for recovery and held disposable off-the-shelf solid-fuel engines. Depending on the experimental setup, different engines with characteristic performances could be mounted. The rocket was designed to carry the sensors and controller in its body, and incorporated actuators for active control. A camera was also mounted for in-flight video recording for post-flight analysis.

The dimension of the final rocket, shown in figure 3 was 1.25m with a diameter of 0.06m and overall mass of 0.57kg.

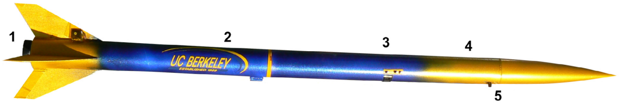


Fig. 3 Final rocket: 1) gimbaled nozzle, 2) parachute bay, 3) camera, 4) IMU, motor and payload section, 5) antenna

3.2 Actuators

To control the longitudinal roll axis (Fig. 4a) , the rocket body (1) contained two concentrically mounted discs (3), driven by two counter-rotating brushless DC motors (2). Controlled acceleration and deceleration of these discs was used to counteract external torques on the rocket's roll axis by compensating angular momentum.

For yaw and pitch control, a gimbaled nozzle was developed to vector the thrust along both axes (Fig. 4b) similar to existing large scale and sounding rockets. An inner engine mount tube (4) was gimbaled on a spherical bearing (7), driven by two high-torque servos (5,6). Controlling the position of the servos steered the rocket engine to point in any direction within a ±4.5° cone.

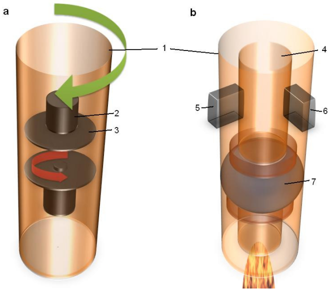


Fig. 4 Control principles: a) spinning discs for roll, b) gimbaled nozzle for pitch and yaw

3.3 Sensors

An on-board inertial measurement unit (IMU) was used to measure the body referenced 6DOF inertial rates. A MEMS accelerometer measured 3 axis linear motion while MEMS gyros measure the 3 axis angular rates. These sensors can be integrated to calculate the 6DOF position.

However, the outputs of these sensors suffer from additive Gaussian noise as well as zero bias drift. The additive noise often integrates out, but the random offset in the rates integrates over time into nontrivial errors. Determining attitude from angular rates requires one integration, and so diverges from true rather slowly; position however is the double integral of acceleration and can accumulate errors rather quickly. To compensate for these errors, additional sensors would be necessary.

Though beyond the scope of this paper, sensors which directly measure position and attitude can be filtered together with the measured rates to generate a more accurate state estimate. Typically, magnetometers, GPS, and cameras complement inertial sensors for localization of robotic systems [21–23]. In the case of minirockets, however, weight is at a premium, and so a minimum of additional sensors are desired. A camera looking at defined features such as the curvature of the earth or celestial bodies can resolve a full 6DOF state estimate, so it may be a good addition to the sensor suite [24].

3.4 Controller

The flight controller for the rocket was a custom designed circuit board for use in small robotic applications, based on the WARPWING project [25]. The Guidance and Inertial Navigation Assistant (GINA) board shown in figure 5 comprises the MEMS inertial sensors, a microcontroller, a 2.4GHz wireless radio, and headers to a daughter card to drive the actuators. The 2g system is the size of a US quarter at half the mass.

Though the microcontroller is capable of implementing control laws itself, for ease of development the system was set up to use a laptop as a command station. The microcontroller polls the sensors and transmits the data wirelessly to a basestation connected to the laptop, which processes the data to generate control outputs. The control signals are send back over the wireless link to the GINA microcontroller, which then drives the actuators via the daughter board. This introduces a slight delay of 6ms in the feedback loop due to time-scheduled communications, as well as a source of errors due to potential packet loss. This cost was tolerated to streamline the development cycle; however a final implementation of an autonomous controller would necessarily incorporate on-board processing to eliminate such problems.

Orbital trajectories are more robust to altitude errors than they are to attitude errors, and so the focus of the controller was on attitude control. The basestation received angular rates from the gyros on the GINA board and integrated them into an attitude estimate. This estimate was demonstrated to track the actual orientation of the rocket quite closely over several minutes, and so this state was fed into various feedback loops to control the rocket's roll, pitch, and yaw over the duration of a flight. These loops generated the control signals which were relayed to the GINA board to set the motor speeds and servo positions.

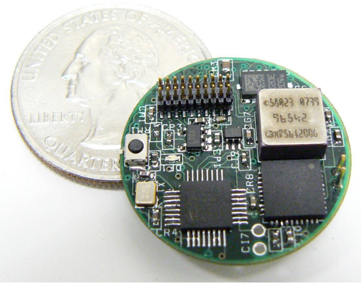


Fig. 5 The 2g GINA controller board incorporates inertial sensing, processing, communications, and actuation. A MEMS accelerometer and gyros, a 2.4 GHz radio, and a connector to an actuator driver board are visible; the processor is on the backside of the board.

4 Hardware Testing and Experimental Results

4.1 Sensor Validation

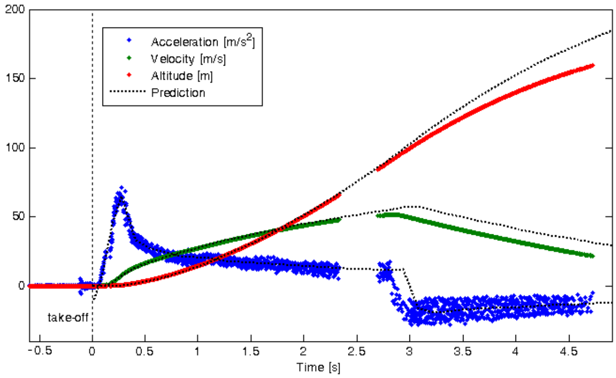


Fig. 6 Measured versus predicted flight profile

The body-referenced angular rates measured by the IMU can be integrated into a full earth-referenced 6DOF state estimate. Using those measurements, the position of a sample uncontrolled rocket flight can be calculated and compared with theoretical predictions. This comparison is shown in figure 6.

The predictions account for aerodynamic drag and gravity and are based on thrust-time data sheets from the National Association of Rocketry (NAR) for off-the-shelf model rocket engines [26] given the dimensions of the rocket and its time dependent mass. For the latter, we assumed the expelled mass to be proportional to the thrust. Looking at the acceleration, we can clearly see the characteristic thrust-time behavior of the engine (here a *Estes^{TM} E9-4*), with a initial peak thrust due to the engines core burning for the lift-off boost, a subsequent steady burning followed by the engines burnout and deceleration of the rocket during the coasting phase. Predicted and measured data match perfectly until the engine's burnout. A significant shorter burning time of the engine than predicted by the NAR (due to manufacturing tolerances) led therefore to the deviations in velocity and altitude after this event. Around T+2.5s, connection to the basestation was lost for a short period of time due to unknown reason but occurred during a few flights. Tracking the rocket with the antenna decreased the chance of losing data packets.

4.2 Open Loop Actuator Demonstration

4.2.1 Attitude

For safety reasons, the attitude (yaw + pitch) controller was first tested on the ground. The rocket was fixed at its center of gravity, allowing free rotation about the pitch axis (Fig. 7c). Commanding the servo to vector the engine to maximal deflection, +4.5°, led to an angular acceleration of approximately 6rad/s^2 during the peak thrust of a *Estes^{TM} C6-0* engine. This matched quite well with the expected behavior, as seen in figure 7a. The prediction was based on the NAR engine data sheet [26], the thrusting angle and the measured moment of inertia of the rocket. In Fig. 7b, the thrust vector was switched from one endpoint of +4.5° to the opposite endpoint at -4.5° at the time indicated by the vertical dotted line. After about a 0.2s delay caused by accumulated mechanical hysteresis and stiction, the nozzle vectored its thrust in the opposite direction, decelerating the pitching of the rocket, causing it to stop and reverse its rotation.

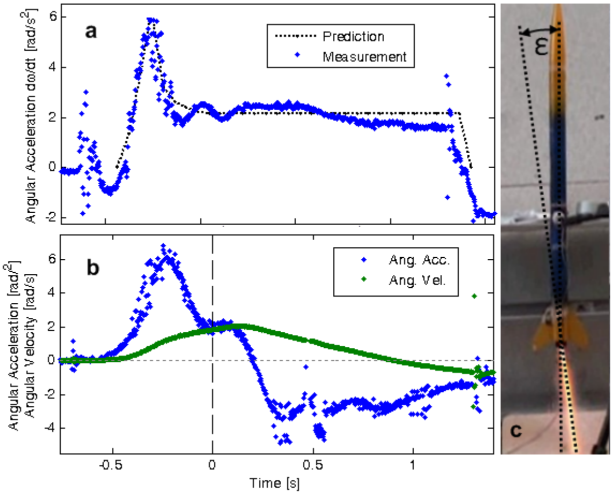


Fig. 7 Angular acceleration depending on thrusting vector ε

For the actual open loop controlled flight with the vectored thrust, the rocket was programmed to sinusoidally swing the nozzle in the pitch axis back and forth at a 3 Hz frequency, starting the wiggle when the rocket reached an altitude of 7m. The goal was to directly control the rocket's pitch and hence its trajectory by gimbaling the nozzle. Figure 8b shows the resulting pitch angular velocity. Around T+1s, when the controller's state estimate indicated a 7m altitude, the nozzle started its sinusoidal movement which led to the same wave like pitch movement, revealed by its angular velocity. As seen before, the rocket's response was time shifted by around 0.2s due to stiction and hysteresis. The initial pitching of the rocket is visible on all previous launches, and occurs due to unstable low-velocity behavior. As the relative speed of the rocket increases, the fins add stability and the swinging disappears. For this experiment, the rocket was powered by a *Estes^{TM} E9-4* engine with a 3s burning time for additional time for thrust vectoring. An evidently wavelike trajectory can be observed in the rocket's smoke trail (Fig. 8a), visually supporting the measured data.

4.2.2 Roll

For roll control, accelerating and decelerating the discs during the flight caused the rocket to rotate back and forth along its longitudinal axis. One big advantage of this system is that it is independent of thrust, being still capable to control during the coasting phase. The major drawback is its tendency for saturation once the motors are spinning at full speed, since the discs act just as a reservoir for angular momentum but are not capable to get rid of it. Knowing expected torques on the rocket body, appropriate disc and motor selection can help to overcome this problem, but might also increase the overall mass.

4.3 Closed Loop Feedback

To keep a rocket on a desired trajectory, closed loop feedback control is necessary. Since the rocket presented in this paper steers by vectoring its thrust in a particular direction, holding the roll axis constant is important to prevent a corkscrew like flight path. Depending on the deviation from the initial roll angle at the launch pad, a PID controller drives the rotation speed of the two discs, compensating any external torque, e.g. wind gusts or engine nonuniformities. For testing and simulation purposes, the rocket was suspended by attaching a thread to its nose cone for unhindered longitudinal rotation. Once the PID gains were manually tuned, two sets of experiments were conducted: first, a fan blowing asymmetrically on the rocket fins induced a constant torque; second, a table tennis ball hitting one of the fins generated a torque impulse. In both cases, the PID controller tried to hold the rocket in its initial orientation.

As mentioned above, the actuator can only compensate for a certain amount of angular momentum, limited by the motor speed and the combined moment of inertia of the rotor and disc. Therefore, application of a constant torque will finally end in

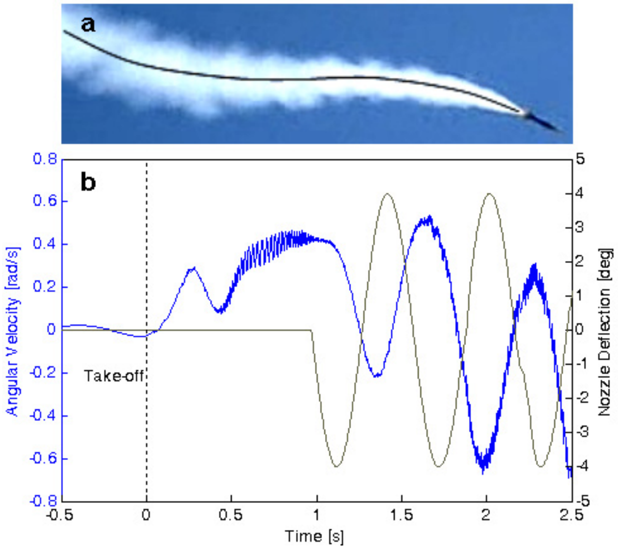


Fig. 8 a) The sinusoidal smoke trail is generated by thrust vectoring, b) Measured angular velocity demonstrates the rocket's response to pitch control. The oscillation immediately after the take-off is due to unstable flight at low velocities

rotation, but it can be delayed by a significant amount of time as shown on the left in figure 9 compared to the uncontrolled case.

In the case of an abrupt event like the table tennis ball hitting a fin (simulating a short duration gust or the like) the system reacts fast to stop the rotation. In the uncontrolled case (Fig. 9a), the rocket keeps turning steadily after the impact, slightly decelerated by the air resistance of the fins. By contrast, in the PID controlled scenario, the rotation stops suddenly and the roll controller drives the rocket back towards its initial position, as seen in figure 9b. The observed oscillations are vibrations generated by off-center mounting of the reaction masses on the flywheels. Tighter tolerances during manufacturing could improve the response.

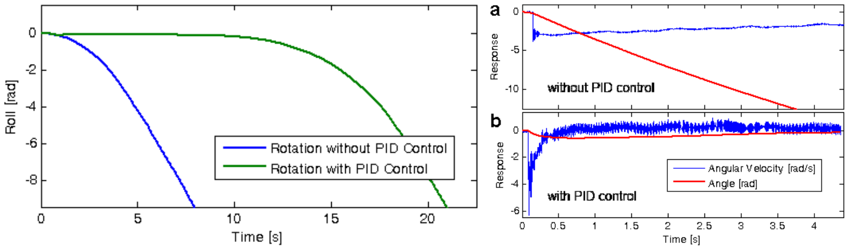


Fig. 9 Roll control with spinning flywheels given constant torque (left) and an impulse (right)

5 Conclusions and Future Work

The experiments presented in this paper demonstrated the validity of delivering small scale satellites into low Earth orbit using minirockets. The extremely small 2g GINA controller was accurately able to estimate a portion of the state required for trajectory control and command actuators to control that state. In the end, an extremely low cost, off the shelf rocket system was demonstrated with the capability for attitude controlled flight.

However, attitude alone is insufficient for orbital insertion, and for longer duration flights additional sensing and sensor fusion are required to compensate for IMU sensor drift. For example, direct state estimates can be generated with a millimeter/milligram scale horizon or star camera, measuring the relative position of the sensor with respect to the earth and/or astronomical bodies. Future work will address these concerns to develop a more robust controller to accurately guide the rocket along longer and more precise trajectories.

Alternate actuation schemes must be also investigated. Though the system presented in this work was able to achieve full attitude control, the actuators and structure required to do so proved to be quite heavy. Reducing the mass of the controllers directly lowers the final system size, and so more efficient ways of effecting stability and guidance control will be necessary. Such actuation could include inserting controllable vanes to deflect the exhaust stream or shifting the position of the payload to adjust the relative thrust vector of the engine.

With a robust, lightweight, and accurate stability and guidance solution for small rocket control, personal scale satellite launches become possible, opening up new avenues of research for scientists and engineers across a wide range of fields.

Supplemental Media

A video showing key features of this work can be seen at:
`http://people.csail.mit.edu/mehtank/minirocketry.mp4`

References

1. Hitt, D.L., et al.: Mems-based satellite micropropulsion via catalyzed hydrogen peroxide decomposition. Smart Materials and Structures 10(6), 1163–1175 (2001)
2. Vladimirova, T., et al.: Characterising wireless sensor motes for space applications. In: Second NASA/ESA Conference on Adaptive Hardware and Systems, pp. 43–50 (August 2007)
3. Barnhart, D.J., et al.: A low-cost femtosatellite to enable distributed space missions. Acta Astronautica 64(11), 1123–1143 (2009)
4. Janson, S.: Micro/Nanotechnology for Micro/Nano/Picosatellites. In: SPACE Conferences & Exposition. American Institute of Aeronautics and Astronautics (September 2003)
5. Kramer, H.J., Cracknell, A.P.: An overview of small satellites in remote sensing. International Journal of Remote Sensing 29(15), 4285–4337 (2008)

6. Santacreu, E.B.: Study of a low cost inertial platfom for a femto-satellite deployed by a mini-launcher (2010)
7. Wertz, J.R.: Spacecraft attitude determination and control, vol. 73. Springer (1978)
8. Sarigul-Klijn, N., et al.: Air launching earth-to-orbit vehicles: Delta V gains from launch conditions and vehicle aerodynamics. In: 42nd AIAA Aero. Sci. Mtg. and Exhibit (January 2004)
9. Malina, F.J.: The problem of escape from the earth by rocket. Journal of the Aeronautical Sciences (Institute of the Aeronautical Sciences) 14(8) (2012)
10. Tsohas, J., et al.: Sounding rocket technology demonstration for small satellite launch vehicle project. In: AIAA 4th Responsive Space Conference, Los Angeles, CA, pp. 1–15 (2006)
11. Ransone, E.D., Gregory, D.D.: An overview of the NASA sounding rockets and balloon programs. In: 17th ESA Symposium on European Rocket and Balloon Programmes and Related Research, vol. 590, pp. 19–24 (2005)
12. Kanevsky, V., Ventskovsky, O.: Gun launch system for small satellites: A fresh look. In: Small Satellites for Earth Observation: Selected Proc. of the 5th Int'l. Symp. of the Int'l. Academy of Astronautics, Berlin, April 4-8, p. 214 (2005)
13. Toorian, A., Diaz, K., Lee, S.: The cubesat approach to space access. In: 2008 IEEE Aerospace Conference, pp. 1–14 (2008)
14. Rainwater, E.L., Smith, M.S.: Ultra high altitude balloons for medium-to-large payloads. Advances in Space Research 33, 1648–1652 (2004)
15. Van Allen, J.A.: Balloon-Launched Rockets for High-Altitude Research, ch. 9 (1959)
16. Gizinski, S.J., Wanagas, J.D.: Small satellite delivery using a balloon-based launch system. In: International Communication Satellite Systems Conference and Exhibit (March 1992)
17. Nakka, R.A.: Solid Propellant Rocket Motor Design and Testing. U. Manitoba (1984)
18. Jove-Casulleras, R., Tristancho, J., Vera-Flores, C.: Technical Constraints for a Low-Cost Femto-Satellite Launcher. AIAA Aero. Sci. Mtg. (January 2011)
19. Tsien, H.S.: Optimum thrust programming for a sounding rocket. Journal of the American Rocket Society 21(5), 99–107 (1951)
20. Ghersin, A.S., Pena, R.S.S.: Lpv control of a 6-dof vehicle. IEEE Transactions on Control Systems Technology 10(6), 883–887 (2002)
21. Eldridge, A.M.: Improved State Estimation for Miniature Air Vehicles. Master's thesis, Brigham Young University (2006)
22. Gemeiner, P., Einramhof, P., Vincze, M.: Simultaneous Motion and Structure Estimation by Fusion of Inertial and Vision Data. The International Journal of Robotics Research 26(6), 591–605 (2007)
23. Montenbruck, O., Markgraf, M.: A gps tracking system with onboard iip prediction for sounding rockets. Journal of Spacecraft and Rockets 41, 644–650 (2003)
24. Izquierdo, L., Tristancho, J.: Next generation of sensors for femto-satellites based on commercial-of-the-shelf. In: 2011 IEEE/AIAA 30th Digital Avionics Systems Conference (DASC), p. 8A4-1. IEEE (2011)
25. Mehta, A.M., Pister, K.S.J.: Warpwing: A complete open source control platform for miniature robots. In: IROS, pp. 5169–5174. IEEE (2010)
26. National association of rocketry: NAR certified motors, `http://www.nar.org/SandT/NARenglist.shtml` (updated June 03, 2009, accessed September 15, 2009)

Posture Reconfiguration and Navigation Maneuvers on a Wheel-Legged Hydraulic Robot

Christopher Yee Wong, Korhan Turker, Inna Sharf, and Blake Beckman

Abstract. Wheel-legged hybrid robots are known to be extremely capable in negotiating different types of terrain as they combine the efficiency of conventional wheeled platforms and the rough terrain capabilities of legged platforms. The Micro-Hydraulic Toolkit (MHT), developed by Defense Research and Development Canada at the Suffield Research Centre, is one such quadruped hybrid robot. MHT's relatively small size, mobility, actuation and locomotion types fill a gap in military unmanned ground vehicles (UGVs). Previously, a velocity-level closed loop inverse kinematics controller had been developed and tested in simulation on a detailed physics-based model of the MHT in LMS Virtual.Lab Motion. The controller was employed to generate a variety of posture reconfiguration maneuvers, such as achieving minimum or maximum chassis height at specific wheel separations. In this paper, the aforementioned inverse kinematics controller was adapted to function on the physical MHT. Several test maneuvers, including chassis height and pitch reconfiguration and uneven terrain navigation maneuvers, were implemented on the MHT and the robot's performance was evaluated.

1 Introduction

Unmanned ground vehicles (UGVs) are playing an increasingly large role in replacing or aiding humans in a multitude of tasks, ranging from large-scale construction [10], search and rescue missions [5], everyday housework [7], to extraterrestrial ex-

Christopher Yee Wong · Korhan Turker · Inna Sharf
McGill University, Department of Mechanical Engineering, Montreal, Canada
e-mail: christopher.wong3@mail.mcgill.ca,
korhan.turker@mail.mcgill.ca, inna.sharf@mcgill.ca

Blake Beckman
Defense Research and Development Canada Suffield Research Centre,
Autonomous Systems Operations-Ground Group, Suffield, Canada
e-mail: blake.beckman@drdc-rddc.gc.ca

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_15, © Springer International Publishing Switzerland 2015

ploration [13, 22]. The employment of such robots relieves human beings from the risk and monotony associated with these tasks. Unfortunately, the mobility, dexterity, perception, and intelligence of human beings are removed from the task as well. Thus, mobile robotics research attempts to bridge the gap between human and robots.

The most important aspect of mobile robots, common to all types of mobile systems, is locomotion and mobility. A mobile robot without efficient and effective locomotion cannot complete its mission. Particular locomotion types are advantageous in certain terrains, but can be catastrophic in others, and there is often a balance between efficiency and mobility. Typically, wheeled robots are known for efficiently traversing long distances on continuous terrain, and legged robots excel when negotiating rough terrain. Wheel-legged hybrid locomotion, a type of locomotion combining legs with wheel end-effectors, not only retains the advantages native to both, but expands the robot's capabilities with new locomotion modes [15, 6, 16, 20], active suspension [3], improved stability [2] or new step negotiation methods [21, 17, 12].

The focus of this paper is on UGVs under development by defense organizations for reconnaissance or load-carrying purposes. These sorts of robots come in many different sizes depending on the application, ranging from small throw-able surveillance robots such as iRobot's FirstLook [11], to larger robots such as Boston Dynamic's BigDog [14] or the US Army's car-sized Multi-Mission Unmanned Ground Vehicle (previously known as Multifunctional Utility/Logistics and Equipment Vehicle) [4].

The Micro-Hydraulic Toolkit (MHT) is an advanced prototype UGV developed by DRDC at the Suffield Research Centre for the purposes of investigating new mobility and control algorithm development. The Toolkit's size and payload capacity were designed to produce a platform capable of autonomously navigating urban environments [1] while acting as a load-carrying mule for soldiers or as a reconnaissance vehicle. Within the author's knowledge, there is a lack of military robots with characteristics similar to the MHT in terms of its locomotion type, actuation method, size and payload. Implementation and evaluation of controllers for robots such as MHT is imperative. Thus, the MHT fills a gap in currently available service robots and this paper contributes to the advancement of control algorithms developed for electrically and hydraulically actuated ground vehicles.

2 Micro-Hydraulic Toolkit

The MHT is a quadruped with 12 controllable degrees of freedom. The chassis houses the self-contained hydraulic system and connects to 4 identically structured legs in a mirrored configuration (Fig. 1). Each leg assembly follows a biologically-inspired nomenclature mimicking human anatomy: starting at the chassis from proximal to distal end, the leg assembly is comprised of a hip joint assembly, a 0.315 m femur structural segment, a knee joint assembly, a 0.377 m tibia structural segment, and a wheel end effector assembly with a wheel diameter of 0.254 m.

The separation between the front and rear hip joints is 0.3 m. Both hip and knee joints are powered using non-continuous hydraulic rotary actuators with a range of motion of 100°, and the average maximum angular velocities of the hip and knee joints are 2.39 rad/s and 2.40 rad/s, respectively. The wheels are powered using geared continuous electric rotary motors, with a maximum unloaded rotational speed of 3.86 rad/s. The robot has a mass of approximately 150 kg and occupies approximately 1 m^3. More details of MHT's structure can be found in [23]. The hydraulic actuators are mounted such that when the femur is parallel to the chassis and the tibia is perpendicular to the femur, the actuators are in their zero position, also known as the *home configuration* (Fig. 2). The hip actuators allow motion of [-50°, 50°], whereas the knee actuator spans [-40°, 60°]. The positive direction for the hip and knee joints is defined as up or extending away from the center of the robot.

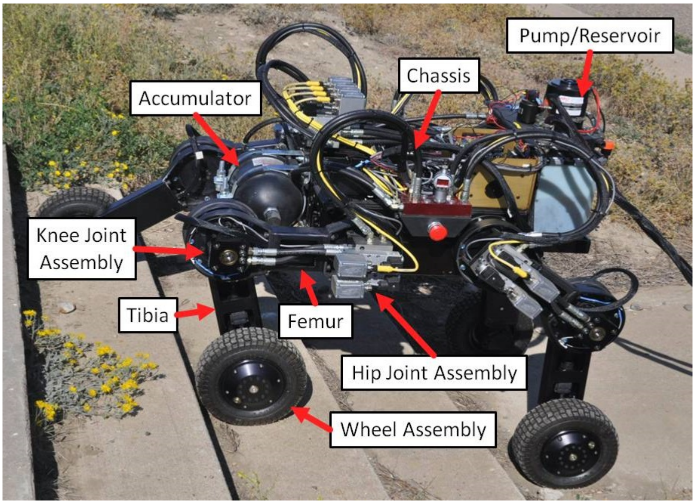


Fig. 1 Breakdown of MHT's anatomy

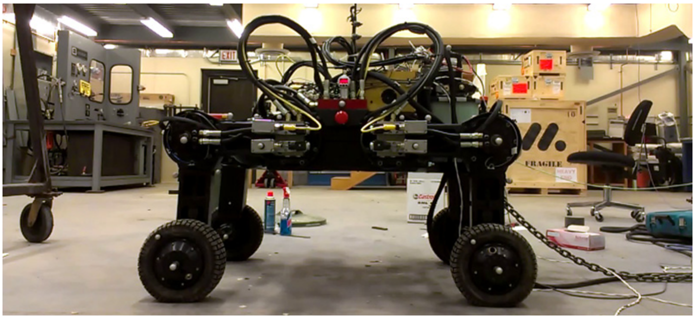


Fig. 2 MHT in *home configuration*, where all hip and knee joints of all legs are set to 0°

Despite having all electronics, including sensors (hip and knee joint potentiometers, wheel encoders, and an inertial measurement unit) and Phytec's phyCORE-MPC565 Rapid Development Kit on the robot, MHT is currently not power autonomous and is powered via an umbilical cord. Battery systems are currently being developed to render MHT fully autonomous. Although the system is estimated to have a payload capacity of approximately 75 kg, no actual tests have been performed on MHT yet, but each hydraulic actuator is capable of 412 N·m torque at the standard operating hydraulic fluid pressure of 17237 kPa (2500 psi). Compared to electric motors, especially with high torque requirements, hydraulic actuators have excellent power to weight ratios.

With 12 controllable degrees of freedom, the MHT is a redundantly-actuated robot. Thus, combined with MHT's hybrid nature, the redundancy allows the robot to readjust its posture to optimize certain criteria depending on the task at hand. For example, in small corridors or areas with low ceilings, the MHT is able to minimize its wheel separation or chassis height, effectively reducing the space occupied. If high stability is required, the MHT can expand its wheel base. The reconfigurable legs can be used as active suspension, maintaining a desired chassis pose. Examples of these maneuvers will be shown in Section 5.

3 Control Scheme

An inverse kinematics controller originally proposed in [9] and implemented on the HyLoS robot was adapted and implemented by Thomson [19] in a simulation model of MHT in MATLAB/Simulink and LMS Virtual.Lab Motion. The underlying control architecture used on the Micro-Hydraulic Toolkit is that of velocity control-based state machine. Fig. 3 shows the overall controller architecture.

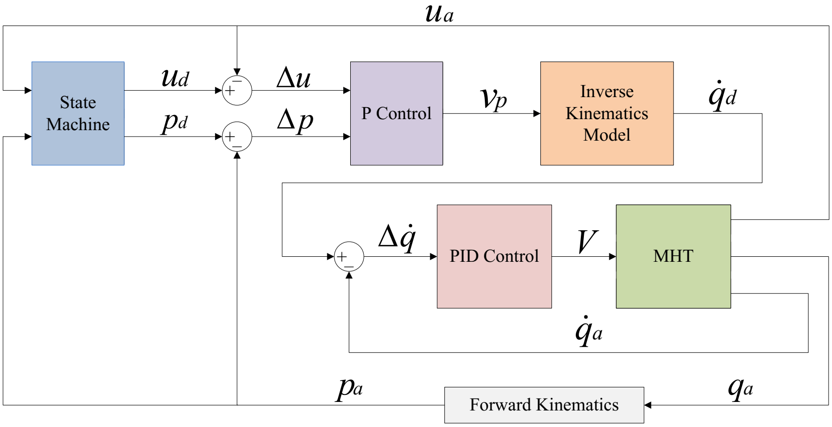


Fig. 3 Overview of MHT's controller architecture

The state machine block in Fig. 3 contains a limited set of possible states, as the robot does not have full autonomous capabilities yet. The controller inputs are separated into two types: desired posture inputs $\mathbf{p}_d$ and desired trajectory inputs $\mathbf{u}_d$. Complementing these inputs are the actual posture $\mathbf{p}_a$ and trajectory $\mathbf{u}_a$. The posture parameters are defined in (1) and trajectory parameters are defined in (2). The chassis reference frame is shown in Fig. 4, and is located at the bottom center of the chassis.

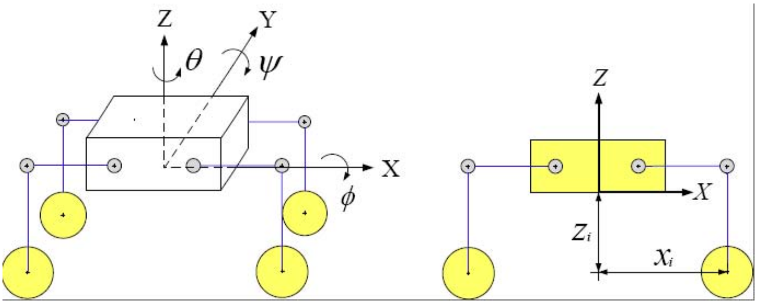


Fig. 4 MHT's reference frame and posture variables definition

$$\mathbf{p} = \begin{bmatrix}\phi \ \psi \ z \ x_1 \ x_2 \ x_3 \ x_4\end{bmatrix}^T \tag{1}$$

$$\mathbf{u} = \begin{bmatrix}x \ y \ \theta\end{bmatrix}^T \tag{2}$$

In the above, ϕ is the body roll, ψ is the body pitch, x_i and z_i are the x- and z-components of the position vector from the chassis center to the wheel joint centers of leg i in the body-fixed reference frame where $i = 1 \ldots 4$, z is the distance from the chassis to the contact surface as defined by (3), x and y are the x- and y-coordinates of the chassis in the global frame and θ is the yaw angle of the robot. All posture variables except for z are graphically illustrated in Fig. 4.

$$z = \frac{\sum_{i=1}^{4} z_i}{4} \tag{3}$$

The posture and trajectory errors, $\triangle\mathbf{u}$ and $\triangle\mathbf{p}$ respectively, are combined with selection matrices $\mathbf{C}_t$ and $\mathbf{C}_p$ and subjected to a proportional control law which transforms them into the desired parametrized platform velocity vector $\mathbf{v}_p = C_t\triangle\mathbf{u} + C_p\triangle\mathbf{p} = \begin{bmatrix}\dot{x} \ \dot{y} \ \dot{z} \ \dot{\phi} \ \dot{\psi} \ \dot{\theta}\end{bmatrix}^T$. The desired joint rates $\dot{\mathbf{q}}_{di} = \begin{bmatrix}\dot{\alpha}_{di} \ \dot{\beta}_{di} \ \omega_{di}\end{bmatrix}^T$, where $\dot{\alpha}_{di}$, $\dot{\beta}_{di}$ and ω_{di} are the joint angular velocities for the hip, knee and wheel, respectively, for leg i, are then determined by (4):

$$\dot{\mathbf{q}}_{di} = \mathbf{J}_i^{-1}\mathbf{L}_i\mathbf{v}_p,\ i = 1\ldots 4 \tag{4}$$

where, for leg i, $\mathbf{J}_i^{-1}$ is the inverse of the modified Jacobian matrix, and $\mathbf{L}_i$ is the modified locomotion matrix. Expanding upon [9], Thomson modified the Jacobian and locomotion matrices to include both the x- and z-positions of the wheels to allow direct control of the wheel base and end-effector height during stepping maneuvers [18]. The actual joint rates $\dot{\mathbf{q}}_{ai}$, derived from the robot's joint potentiometers and encoders, are coupled with the desired joint rates to form the joint rate error $\triangle\dot{\mathbf{q}}$. The joint rate error $\triangle\dot{\mathbf{q}}$ is used in a PID control law to determine the desired joint voltage V, which drive the hip, knee and wheel actuators.

Controller performance is observed through the robot's ability to reconfigure to new desired postures or maintain current desired posture variables in response to outside disturbances (such as terrain changes) while following a desired trajectory.

4 Workspace Analysis

The Toolkit's feasible workspace can be determined by combining the minimum energy stability margin (ESM) [8] requirement and the kinematic analysis of MHT's workspace, for the given limited range of motion of the hip and knee joint actuators. For example, some kinematically feasible postures position the robot's center of mass outside of the support polygon, rendering that posture statically unstable and hence unfeasible. A static stability analysis was performed on MHT comparing the lowest possible ESM values at different wheel separations (Fig. 5), and it was discovered that for wheel separations larger than 0.69 m, all possible postures are statically stable and hence feasible. Whereas with wheel separations below 0.69 m, there exist postures within the robot's workspace that are not statically stable. All test maneuvers (both shown and not shown) are restricted to the feasible set of postures.

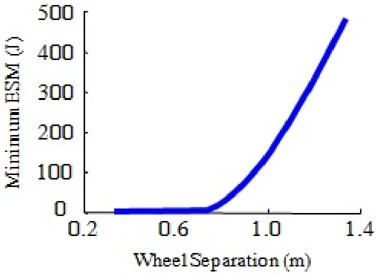


Fig. 5 Relationship between minimum ESM and wheel separation. Note that wheel separations can range from [0.254, 1.500] m

Furthermore, Figure 6 shows the minimum and maximum chassis height given the wheel separation. Any chassis height located within the area bounded by the two curves is feasible. The range of wheel separations is x_{ws} = [0.254, 1.580] m, which represents the separation values where both wheels are touching and the outer limit of the robot's workspace.

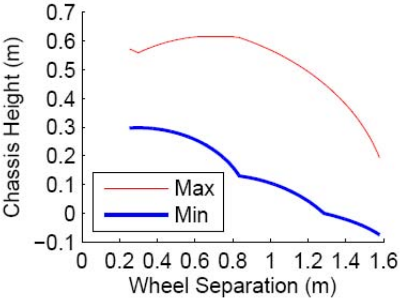


Fig. 6 Relationship between maximum chassis height z and wheel separation x_{ws}

Similarly, a relationship can be established between the maximum possible chassis pitch and chassis height (Fig. 7). The same limitations exist when pitching in both positive and negative directions, and thus only the positive direction is shown. Furthermore, it is important to note that any point below the curve is also kinematically feasible, but not necessarily statically stable.

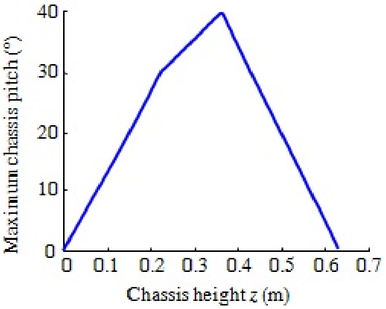


Fig. 7 Relationship between maximum chassis pitch ψ and chassis height z. Wheel separation x_{ws} is not specified in this case to allow unconstrained maximum chassis pitch within joint limits.

Workspace analysis is important as it determines prior to testing which postures are feasible or not without the possibility of damaging the robot. Only feasible postures are used to gauge controller performance in Section 5.1.

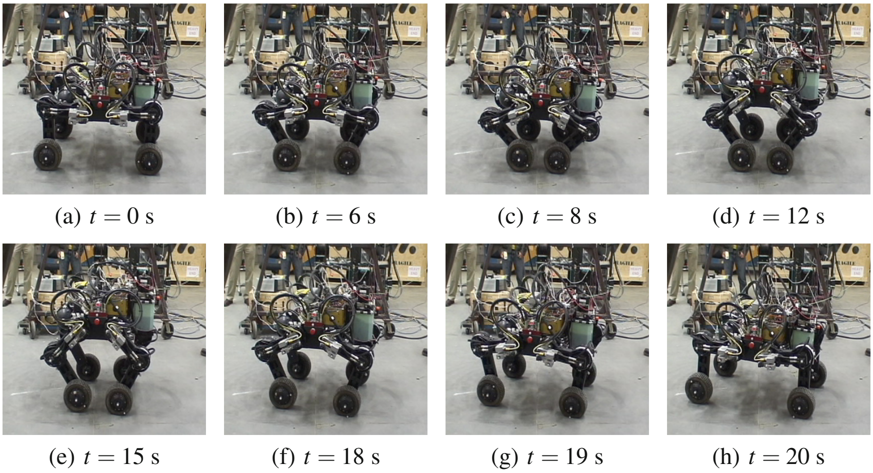
5 Experimental Results

In order to test MHT's capabilities as a hybrid vehicle, we devised a series of maneuvers and obstacles for the robot to perform and overcome. The maneuvers implemented and evaluated on MHT can be divided into two categories: posture reconfiguration maneuvers and navigation maneuvers.

5.1 Posture Reconfiguration Maneuvers

Posture reconfiguration maneuvers involve changing the leg $x-$ and $z-$positions to achieve specific robot chassis poses. The desired pose can be used for optimizing ground contact, maximizing stability, adjusting for ground or ceiling clearance, or repositioning the chassis to various roll/pitch angles to orient a tool mounted on the MHT. These posture reconfiguration maneuvers can be executed stationary or combined with a rolling motion. All maneuvers begin in the home configuration.

The posture reconfiguration maneuver described here involves reconfiguring the robot to a specified wheel separation $x_{ws} = 0.350$ m, while maintaining the current chassis height z. Then, the MHT positions itself to the maximum chassis height $z = 0.575$ m, as limited by the Toolkit's workspace at the desired x_{ws}. The robot will hold this position for several seconds before returning to the home configuration, changing both x_{ws} and z simultaneously. These maneuvers are performed while maintaining zero chassis pitch ψ and zero chassis roll ϕ. Snapshots of the maneuver are shown in Fig. 8.

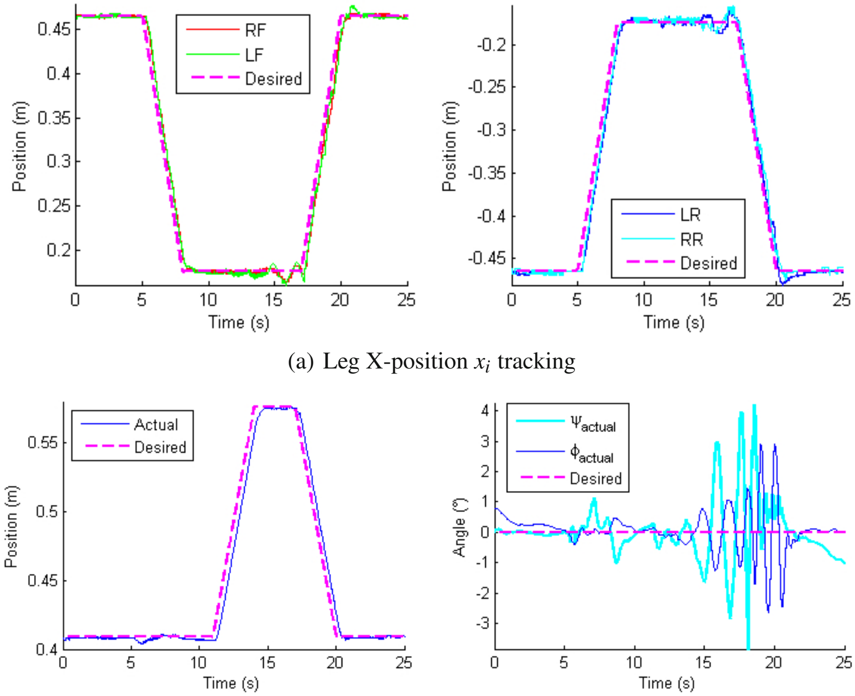


(a) $t = 0$ s (b) $t = 6$ s (c) $t = 8$ s (d) $t = 12$ s

(e) $t = 15$ s (f) $t = 18$ s (g) $t = 19$ s (h) $t = 20$ s

Fig. 8 Snapshots of the maximum chassis height posture reconfiguration maneuver with parameters $x_{ws} = 0.350$ m, $\phi = \psi = 0°$

The tracking responses of certain key variables are shown in Fig. 9. Each leg has been labelled according to its left (L)/right (R) and front (F)/rear (R) relative positioning to the chassis. This maneuver is potentially unstable, given the narrow wheel base and maximum chassis height—the oscillations in pitch tracking between $t = 15$ and 20 s in Fig. 9(c) confirm visible swaying of the robot back and forth. Leg positioning x_i is fairly accurate, as the largest error is less than 0.02 m. Although there is a slight lag in the chassis height z response, the robot achieves the maximum desired chassis height of 0.575 m.



(a) Leg X-position x_i tracking

(b) Chassis height z tracking

(c) Chassis pitch ψ, roll ϕ tracking

Fig. 9 MHT's tracking performance during a stationary maximum chassis height posture reconfiguration maneuver at $x_{ws} = 0.350$ m

5.2 Navigation Maneuvers

Navigation maneuvers conducted to date investigate the controller's ability to overcome different continuous obstacles while maintaining the desired posture parameters. One of the main obstacles is created using different combinations of straight ramps. In all tests performed, the controller attempts to maintain zero chassis roll ϕ and pitch ψ, a wheel separation of $x_{ws} = 0.930$ m and chassis height of $z = 0.409$ m—equivalent to the home configuration when on flat terrain. The MHT is capable of maneuvering up ramps with a maximum slope of 20°, but fails to climb a 22° ramp due to lack of wheel motor torque and loss of ground contact. Simulations of the ramp climbing maneuvers lead to similar conclusions as the robot is able to climb a slope of 20° but not 25° due to wheel slippage.

In the test maneuver presented here, MHT attempts to negotiate sagittally-asymmetric continuous ramps of differing slopes. The left side encounters ramps with slopes of 12° up and 15° down, whereas the right side maneuvers over ramps with slopes of 10° up and 10° down. Fig. 10 shows the layout and dimensions of this test ramp configuration. Figs. 11 and 12 show snapshots and parameter tracking respectively of the MHT negotiating sagittally-asymmetric offset ramps. The maneuver was repeated 4 times since the maneuver outcome varied between trials.

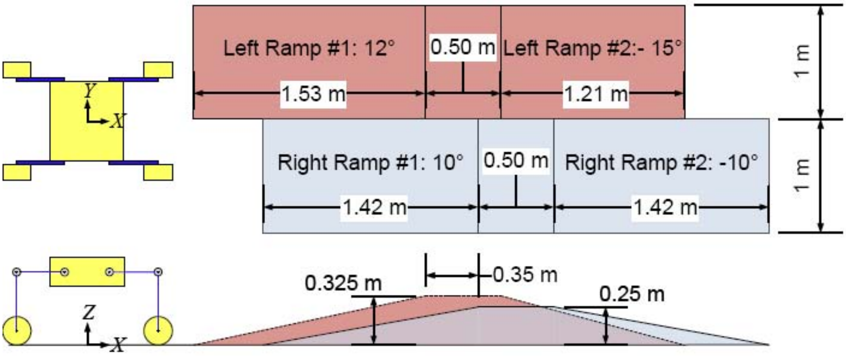
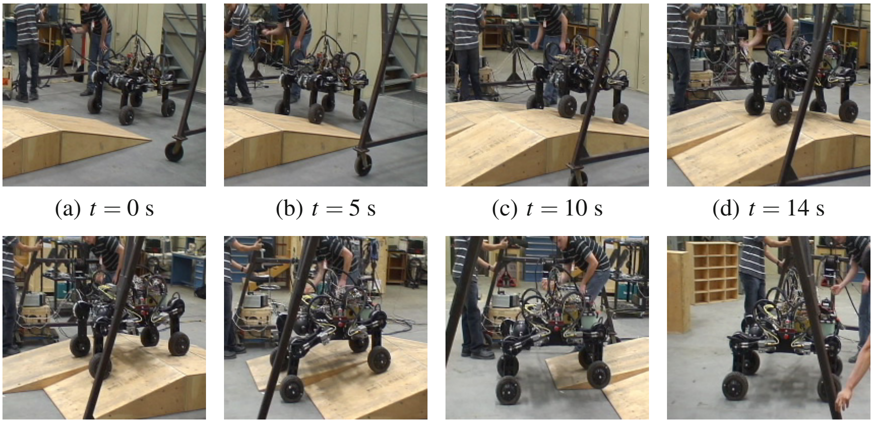


Fig. 10 Sagittally-asymmetric ramps layout, Left (in front): 12°/-15°, Right: 10°/-10°

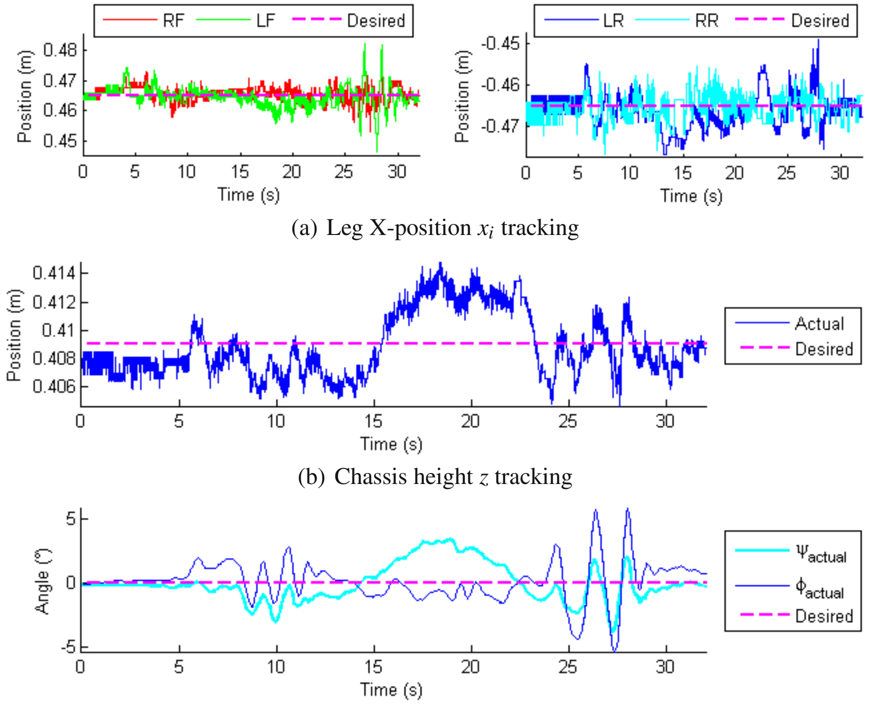
When the MHT negotiates sagittally-asymmetric terrain, the lack of ground contact detection becomes a liability that may render the robot unstable. For example, while the robot moves down the ramp in Fig. 11(e) to Fig. 11(g), the chassis will tend to pitch down, so both rear legs will raise to maintain $\psi = 0$. Combined with the difference in slope between the left and right sides, and the fact that the robot is stable in this configuration with only three supporting legs, the left rear leg loses ground contact, as can be seen in Fig. 13. The LR leg will mimic the RR leg motion and continue to rise to maintain $\psi = 0°$ as the robot moves further down the slope. If the robot's center of gravity shifts and the system becomes unstable, the MHT will fall towards the raised leg, causing sudden large errors in roll and pitch tracking. Attempts to correct the large roll and pitch errors lead to overcompensating motions, causing large oscillations in the roll and pitch angles ($t = 24 \ldots 28$ s, Fig. 12(c)), which leads to catastrophic failures in approximately 50% of the trials attempted. The results presented in Fig. 12 constitute what is classified as a successful trial. Although far from being a desirable response, the defining characteristic of



(a) $t = 0$ s (b) $t = 5$ s (c) $t = 10$ s (d) $t = 14$ s

(e) $t = 17$ s (f) $t = 20$ s (g) $t = 27$ s (h) $t = 30$ s

Fig. 11 Snapshots of the MHT negotiating sagittally-asymmetric offset ramps (Left: 12°/-15°, Right: 10°/-10°)



(a) Leg X-position x_i tracking

(b) Chassis height z tracking

(c) Chassis pitch ψ and roll ϕ tracking

Fig. 12 Tracking performance when negotiating sagittally-asymmetric offset ramps (Left: 12°/-15°, Right: 10°/-10°)

a successful trial is that the robot completes the maneuver without approaching the tipping point even if the LR wheel loses ground contact and the robot undergoes oscillatory motions. An unsuccessful trial occurs when the operator deems the motion becoming dangerously unstable (for example, leading to imminent tipping), or if the robot in fact has tipped over and is prevented from rolling over by the supporting gantry.

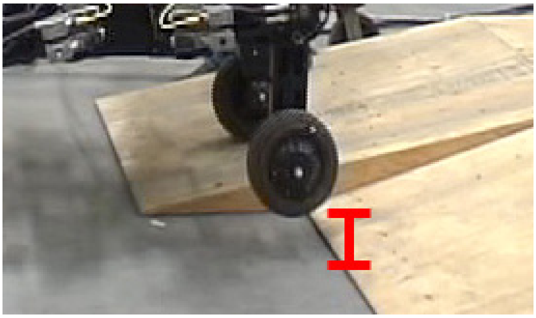


Fig. 13 LR wheel losing ground contact while descending the ramp

Reliable ground contact detection and instantaneous compensation control would likely prevent this problem. Ground contact detection can be achieved by using pressure sensors, or machine learning algorithms to detect loss of contact by relating current wheel speed and the free spinning wheel speed at the current voltage input. Furthermore, the employed posture control uses a simple proportional control law, which was tuned using the simulation model and modified slightly for the real robot. Introduction of integral and derivative terms into the control law may help prevent such overshoot and oscillatory motion of the chassis resulting from loss of contact with the ground.

6 Conclusions

The hybrid wheel-legged quadruped Micro-Hydraulic Toolkit platform was introduced as a service UGV that fills the gap in military robots of that size and actuation type. Several analyses were performed to determine MHT's feasible workspace. Finally, test maneuvers were implemented on the actual robot to gauge the performance of the inverse kinematics controller previously developed in simulation for posture reconfiguration and navigation maneuvers. Performance was found to be very good for stationary posture reconfiguration maneuvers, but inadequate in uneven terrain negotiation scenario. The latter was attributed to lack of ground contact detection, thus pointing to the need for a higher-level event driven (state machine) controller. While leader-follower behaviors are currently being implemented, future work also includes implementing navigation systems to test the trajectory tracking performance of the controller. Furthermore, the MHT's performance can be tested in outdoor environments, or using different controlled terrain profiles.

Acknowledgements. This work was supported by Natural Sciences and Engineering Research Council of Canada, Fonds québécois de la recherche sur la nature et les technologies, and Defense Research and Development Canada (DRDC) Suffield Research Centre. Many thanks for the support received from DRDC scientists as well as Amtech Aeronautical Ltd.

References

1. Beckman, B.: Predicted dynamic stability of the varying center of gravity Micro-Hydraulic Toolkit. University of Calgary (2008)
2. Beckman, B., Pieper, J., Mackay, D., Trentini, M., Erickson, D.: Two dimensional dynamic stability for reconfigurable robots designed to traverse rough terrain. In: Proc. Intelligent Robots and Systems, pp. 2447–2452. IEEE (2008)
3. Ben Amar, F., Grand, C., Besseron, G., Plumet, F.: Performance evaluation of locomotion modes of an hybrid wheel-legged robot for self-adaptation to ground conditions. In: Proc. ESA Workshop on Advanced Space Technologies for Robotics and Automation. Citeseer (2004)
4. Byers, D.B.: Multifunctional Utility/Logistics and Equipment (MULE) Vehicle Will Improve Soldier Mobility, Survivability and Lethality. Defense Technical Information Center (2008)
5. Casper, J., Murphy, R.R.: Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center. Trans. Systems, Man, and Cybernetics, 367–385 (2003)
6. Endo, G., Hirose, S.: Study on Roller-Walker (multi-mode steering control and self-contained locomotion). In: Proc. Robotics and Automation, pp. 2808–2814. IEEE (2000)
7. Forlizzi, J., DiSalvo, C.: Service robots in the domestic environment: a study of the roomba vacuum in the home. In: Proc. ACM SIGCHI/SIGART Conference on Human-Robot Interaction, pp. 258–265. ACM (2006)
8. Garcia, E., Estremera, J., de Santos, P.G.: A comparative study of stability margins for walking machines. Robotica, 595–606 (2002)
9. Grand, C., Ben Amar, F., Plumet, F., Bidaud, P.: Decoupled control of posture and trajectory of the hybrid wheel-legged robot Hylos. In: Proc. Robotics and Automation, pp. 5111–5116. IEEE (2004)
10. Hodoshima, R., Doi, T., Fukuda, Y., Hirose, S., Okamoto, T., Mori, J.: Development of TITAN XI: a quadruped walking robot to work on slopes. In: Proc. Intelligent Robots and Systems, pp. 792–797. IEEE (2004)
11. iRobot: 110 FirstLook,
`http://www.irobot.com/us/learn/defense/firstlook.aspx`
(cited September 10, 2013)
12. Lauria, M., Piguet, Y., Siegwart, R.: Octopus-an autonomous wheeled climbing robot. In: Proc. Climbing and Walking Robots. Citeseer (2002)
13. Michaud, S., Schneider, A., Bertrand, R., Lamon, P., Siegwart, R., Van Winnendael, M., Schiele, A.: SOLERO: Solar powered exploration rover. In: Proceedings of the 7th ESA Workshop on Advanced Space Technologies for Robotics and Automation (2002)
14. Playter, R., Buehler, M., Raibert, M.: BigDog. In: Proc. SPIE (2006)

15. Siegwart, R., Lamon, P., Estier, T., Lauria, M., Piguet, R.: Innovative design for wheeled locomotion in rough terrain. Robotics and Autonomous Systems 40, 151–162 (2002)
16. Smith, J.A.: Galloping, Bounding and Wheeled-Leg Modes of Locomotion on Underactuated Quadrupedal Robots. McGill University (2006)
17. Takahashi, M., Yoneda, K., Hirose, S.: Rough terrain locomotion of a leg-wheel hybrid quadruped robot. In: Proc. Robotics and Automation, pp. 1090–1095. IEEE (2006)
18. Thomson, T.: Kinematic control and posture optimization of a redundantly actuated quadruped robot. McGill University (2013)
19. Thomson, T., Sharf, I., Beckman, B.: Kinematic control and posture optimization of a redundantly actuated quadruped robot. In: Proc. Robotics and Automation, pp. 1168–1174. IEEE (2012)
20. Townsend, J., Biesiadecki, J.: Sliding GAIT Algorithm for the All-Terrain Hex-Limbed Extra-Terrestrial Explorer (ATHLETE). In: Dynamic Systems and Control Conference: 11th Motion & Vibration Conference, ASME (2012)
21. Turker, K., Sharf, I., Trentini, M.: Step negotiation with wheel traction: a strategy for a wheel-legged robot. In: Proc. Robotics and Automation, pp. 1168–1174. IEEE (2012)
22. Wilcox, B.H.: ATHLETE: A cargo and habitat transporter for the moon. In: Aerospace Conference, pp. 1–7. IEEE (2009)
23. Wong, C.Y.: Posture reconfiguration and step climbing maneuvers for a wheel-legged robot. McGill University (2013)

Roll Control of an Autonomous Underwater Vehicle Using an Internal Rolling Mass

Eng You Hong and Mandar Chitre

Abstract. A stable autonomous underwater vehicle (AUV) is essential for underwater survey activities. Previous studies have associated poor results in bathymetry survey and side-scan imaging with the vehicle's unwanted roll motion. The problem is becoming more prominent as AUVs are smaller nowadays. This causes reduction in the metacentric height of the AUVs which affects the inherent self-stabilization in the roll-axis. In this paper, we demonstrate the use of an internal rolling mass (IRM) mechanism to actively stabilize the roll motion of an AUV. We rotate the whole electronics tray, which has an off-centric center of gravity, to produce the required torque to stabilize the roll motion. The mechanical design of such mechanism and its dynamics modeling are discussed in detail. A Proportional-Integral (PI) controller is synthesized using the identified linear model. Results from tank tests and open-field tests demonstrate the effectiveness of the mechanism in regulating the roll motion of the AUV.

1 Introduction

A stable autonomous underwater vehicle (AUV) is essential for underwater surveys such as seafloor imaging using side-scan sonar, bathymetric mapping using multibeam sonar, and photo mosaicking using underwater camera. As compared with yaw and pitch, the roll of a torpedo-shape AUV has a smaller moment of inertia and drag. So, the roll dynamics is oscillatory when the AUV is subjected to induced propeller torque, unknown disturbances and banking motion during turns. Without roll stabilization, the unwanted roll motion of an AUV can be problematic [7].

Eng You Hong
ARL, Tropical Marine Science Institute, National University of Singapore, Singapore
e-mail: tmseyh@nus.edu.sg

Mandar Chitre
Department of Electrical & Computer Engineering and ARL,
Tropical Marine Science Institute, National University of Singapore, Singapore
e-mail: mandar@nus.edu.sg

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_16, © Springer International Publishing Switzerland 2015

Singh, *et al.* [9], stated in their bathymetry paper, the roll bias is the most dominant error source as it directly affects the slope of the area being surveyed. Kirkwood, *et al.* [3] stated that the roll stability is critical to multibeam mapping and it is of high priority. For a side-scan sonar application, the AUV roll motion may cause layover to occur [10]; the affected samples are hard to interpret and need to be discarded. The unwanted roll motion can also affect both diving and steering performance of the AUV. This is due to the fact that most feedback controllers are designed on the assumption that yaw and pitch motion are decoupled. When the roll of the AUV is non-zero, the assumption is violated and thus the performance of a decoupled controller may be affected [6].

The problem is becoming more prominent as AUVs are smaller nowadays. Smaller AUVs are built in order to reduce manufacturing costs and ease deployment by one or two operators. Smaller AUVs pose constraints in placement of internal components and cause reduction in the metacentric height of the AUVs. This affects the inherent self-stabilization in the roll-axis. As a result, smaller AUVs are vulnerable to oscillatory roll motion.

In this study, we investigate the use of an internal rolling mass mechanism to actively stabilize the roll motion of an AUV. Internal actuators have few appealing features. Firstly, they can be used at low speeds where fins lose their usefulness. Secondly, they can be housed completely inside the vehicle and therefore are less prone to damage due to impact or corrosion [11]. Thirdly, they do not create external drag.

The use of an internal moving mass is not new in underwater vehicle applications. It has been used in underwater gliders such as SLOCUM, the Spray glider and the Seaglider [5]. The use of internal moving mass is also found in some AUVs. One example is the hybrid AUV – eFolaga [1] where the battery is moved along the longitudinal axis to provide pitch control. However, the use of an internal mass for roll control is rare because of the limited lateral space available for any significant linear motion. Furthermore, the use of linear motion requires a runway for the moving mass which is practically infeasible as the internal space already crowded with the essential components. We got around this limitation by designing a rolling mass mechanism that made use of the whole electronics tray (including batteries) as a moving mass.

The moving mass is capable of rotating with respect to the longitudinal axis of the AUV – hence we call it as an internal rolling mass (IRM) mechanism. The center of gravity (CG) of the IRM is off-centric. By rotating the IRM, we effectively change the CG of the AUV. By using the gravity force that acting through the CG, we can therefore generate the required torque to stabilize the roll dynamics.

In this paper, we tackle the unwanted roll motion through active roll stabilization by using the IRM mechanism. To the best of our knowledge, we are the first to report on the use of internal moving mass to stabilize the roll of an underactuated, tail-thrusted and fins-controlled AUV. We illustrate the design of the IRM mechanism by implementation on the STARFISH AUV [4]. The STARFISH AUV is a streamlined

AUV with a single thruster and four control surfaces at the tail. This is the most widely used configuration for AUVs in both commercial and research communities. Hence our discussion of the IRM mechanism is relevant and widely applicable to many AUVs in-use today.

This paper is organized as follows: In section 2, we illustrate the mechanical design of the IRM mechanism. This is followed by the modeling of the roll dynamics of the AUV which the IRM acts as an actuator in section 3. We present the results of system identification where parameters of the model were identified in section 4. In section 5, we show how controller was designed to regulate the roll motion. Experimental results are presented in section 6. Lastly, in section 7 we present the conclusions.

2 Mechanical Design

2.1 STARFISH AUV

The STARFISH AUVs are torpedo-shaped with 200 mm diameter. They are designed to be modular. The baseline STARFISH AUV consists of 3 basic sections - a nose section, a command & control section and a tail section. The total length is about 1.6 m and it weighs about 45 kg. Additional payload sections can be added to the baseline STARFISH AUV depending on the application. We currently have a Doppler Velocity Log (DVL) section, a side-scan sonar section and a chemical sensor section in our collection of payloads. The interested reader may refer to [4] for detailed discussions on the mechanical, electrical and software interfaces between the sections of the STARFISH AUV.

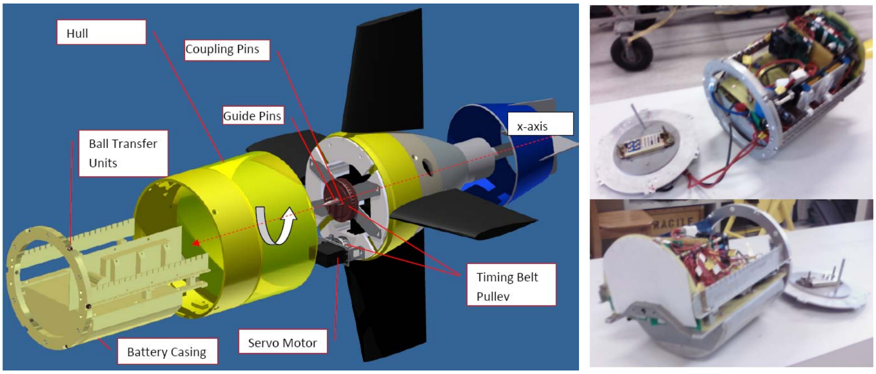


Fig. 1 Mechanical design of the Internal Rolling Mass (IRM) mechanism. Pictures on the right show the tail electronic tray which has a battery tray that attached to its bottom half.

2.2 Design Requirements

We need a mechanism that able to shift the CG of the AUV in the sway axis such that the roll equilibrium of the AUV can be changed by $\pm 5^{\circ}$. In order to shift the CG, we need some form of moving mass. So, it can be either a linear moving mass or a rotating mass. Our implementation using a rotating mass is illustrated in Fig. 1.

The actuation is provided by a servomotor mounted at the bottom end of the tail section through a bracket. It has a maximum torque of 1.92 Nm and maximum speed of 6.16 rad/s. Two timing belt pulleys are used for power transmission from the servomotor to the central axis. The drive pulley ratio is 1:2, thus increasing the output torque by a factor of two. Guide pins are used to guide the assembly of the whole tail tray (nickel bright in the figure) into the hull. Two coupling pins are used to transmit the torque from the central pulley to the tail tray. As the main mass of the tail tray is contributed by the battery placed in the bottom half, we effectively change the CG as the tail tray rolls inside the hull.

This design fulfills the following requirements :

2.2.1 Space Constraint

Constrained by the AUV diameter of 200 mm, there is no sufficient runway for a linear moving mass to have an effective change in CG. In addition, the existing components, such as electronics and battery, have already taken up most of the space in the tail section. So, without affecting components in other AUV's sections, we consolidate all the existing components in the tail section onto a tail tray, and make the tail tray as our moving mass. We were able to find space for a servomotor, two pulleys and a timing belt without making changes to the existing tail section hull (such as elongate it).

2.2.2 Energy Consumption

By having the mechanism at the tail section, we make use of the existing micro-controller to control the servomotor. The same micro-controller is used for thruster and fins control. Six ball transfer units are located on the outer ring of the tail tray. This effectively uses the ring as a bearing and allows low friction rotational motion. In order to provide the required torque and accuracy, we used a Futaba digital servomotor which consumes maximum 12 W. We use a timing belt drive system which has a low power transmission loss.

2.2.3 Ease of Assembly

Ease of assembly is one of the important design criteria. We occasionally need to disassemble the vehicle for routine maintenance and repairs. With the design, the assembly and disassembly work can all be performed by a single engineer in our laboratory within half an hour.

2.2.4 Effective change of CG

The servomotor has a usable range of 80°. After the pulley ratio, the range reduces to 40°. By placing the IRM at the center, we are able to roll the IRM to ±20°; this translates to an effective change of CG to give a roll of ±5°[1] at equilibrium.

3 Modeling

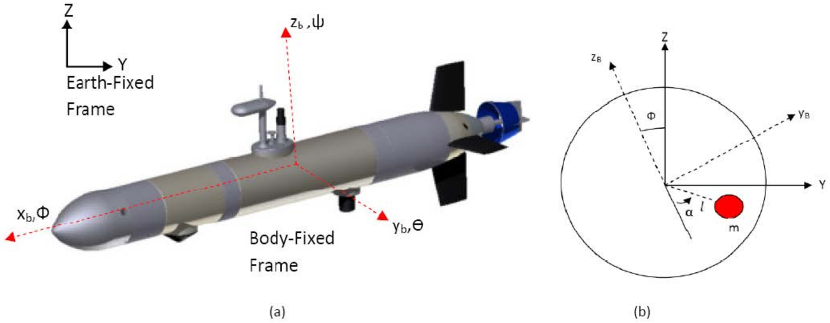


Fig. 2 (a) Coordinate Reference Frame (b) Free Body Diagram

In this section, we derive the dynamics model for the AUV's roll under consideration of CG shift due to the IRM. A six degree-of-freedom (DOF) dynamics model of an AUV is commonly described by a set of nonlinear equations with respect to two coordinate frames as indicated in Fig. 2(a). Detailed discussion on the modeling can be found in [2, 8]. However, for the purpose of this paper, we restrict our analysis only on rolling motion and treat coupling torque induced by others DOFs to be disturbances.

In Fig. 2(a), we have the body-fixed frame at the center of buoyancy (CB) of the AUV. So the CB is located at $z_b = 0$ and $y_b = 0$ with respect to body-fixed frame. The CG is located below the CB in order to provide righting moment. So the CG location (y_g, z_g) has negative z_g with respect to body-fixed frame.

From Newton's Second Law of Motion (rotation), we can write the net total torque as the product of the moment of inertia I_{xx} and roll angular acceleration $\ddot{\phi}$.

$$\sum \tau = I_{xx}\ddot{\phi}. \tag{1}$$

The sum of the external torque consists of following components:

[1] Depending on the vehicle payload configuration, this range might change.

3.1 Hydrostatic Righting Moment

The hydrostatic righting moment is the combined effect of the vehicle's weight W and buoyancy B. The STARFISH AUV is slightly positively buoyant but as we put the body-fixed frame at the CB, buoyancy does not play a role in the equation. The roll torque due to hydrostatic righting moment is

$$\tau_{Hydro} = -y_g W \cos\phi + z_g W \sin\phi. \tag{2}$$

The IRM is treated as a point mass with effective length l from the center. The effective length l is the distance from the CB to the CG of the tail tray. Let α denote the angular position of the point mass as illustrated in Fig. 2(b). When the point mass is rolling in the AUV, it is effectively changing the CG of the AUV. The new CG position (y'_g, z'_g) is described in following two equations:

$$y'_g = y_g + \frac{m}{M} l \sin\alpha \tag{3}$$

$$z'_g = z_g - \frac{m}{M} l \cos\alpha \tag{4}$$

where m is the mass of tail tray and M is the mass of the whole AUV.

By substituting (3) and (4) into (2), the hydrostatic righting moment becomes

$$\tau_{Hydro} = -(y_g + \frac{m}{M} l \sin\alpha) W \cos\phi + (z_g - \frac{m}{M} l \cos\alpha) W \sin\phi. \tag{5}$$

It is useful to note that the hydrostatic moment stabilizes the roll motion as the moment always acts against any deflection in roll. So the roll dynamics are self-stabilized in this sense.

3.2 Rolling Drag

As a streamlined AUV, the main rolling drag of the STARFISH AUV comes from the four fins that protrude out from the center axis. We model the drag as a quadratic drag:

$$\tau_{Drag} = K_{pp} p |p| \tag{6}$$

where K_{pp} is the rolling quadratic drag coefficient and p is the angular velocity of the roll. Since we restrict our discussion in roll axis only, we have $p = \dot{\phi}$.

3.3 Rolling Added Mass

Added mass is a measure of the mass of the moving water when the vehicle accelerates. For a streamlined AUV, rolling added mass due to the AUV hull is small. So the main rolling added mass is again due to the fins. We model the moment due to the added mass as follows:

$$\tau_{AM} = K_{\dot{p}} \dot{p} \tag{7}$$

where $K_{\dot{p}}$ is the rolling added mass coefficient and $\dot{p}$ is the angular acceleration of roll. Similarly, we have $\dot{p} = \ddot{\phi}$.

3.4 Propeller Induced Torque

When the propeller rotates clockwise to provide the forward thrust, it also creates an anti-clockwise torque acting on the AUV. This is commonly known as the *torque effect*. The magnitude of the torque depends on the power output of the thruster, P and propeller revolution, ω in the following equation:

$$\tau_{prop} = \frac{P}{\omega}. \tag{8}$$

Power produced by the thruster is the product of thrust F, and speed of the AUV V. However during steady state (constant velocity) AUV motion, thrust is equal to the drag force, F_{drag}, and therefore

$$P = FV = F_{drag}V \tag{9}$$

$$F_{drag} = \frac{1}{2}\rho AC_d V^2 \tag{10}$$

where ρ is the sea water density; A is the frontal area; C_d is the drag coefficient. So, by running different constant thrusts experiments, we plot the induced torque against the propeller revolution in Fig. 3. The data best fit a quadratic equation showing $\tau_{prop} \propto \omega^2$.

In our subsequent analysis, we omit the induced torque and treat it as a disturbance to the system. However, we pre-roll the AUV to +5° during weight trimming to compensate for the thruster torque at nominal speed. When the AUV moves at its nominal speed of 1.4 m/s with 1400 rpm, the induced torque will roll back the AUV to zero roll position and thus leave sufficient room for IRM to compensate for the rest of the variations.

By substituting (5), (6), (7) and (8) into (1) and rearranging the terms, we have

$$\begin{aligned}(I_{xx} - K_{\dot{p}})\ddot{\phi} = &-(y_g + \frac{m}{M} l \sin\alpha)W\cos\phi \\ &+(z_g - \frac{m}{M} l \cos\alpha)W\sin\phi \\ &+K_{pp}p|p| \\ &+\tau_{prop}.\end{aligned} \tag{11}$$

We obtain the transfer function of roll ϕ, as a function of α in (12) by first linearizing (11) at the operating point $\phi = 0$. At this point $\cos\phi \simeq 1$ and $\sin\phi \simeq \phi$. α can be assumed to be small. Therefore $\cos\alpha \simeq 1$ and $\sin\alpha \simeq \alpha$. Next, we approximate the quadratic drag $K_{pp}p|p|$ as linear drag $K_p p$. By trimming condition, y_g is close to zero and thus ignored. Lastly, τ_{prop} is treated as disturbance and is not included in the equation.

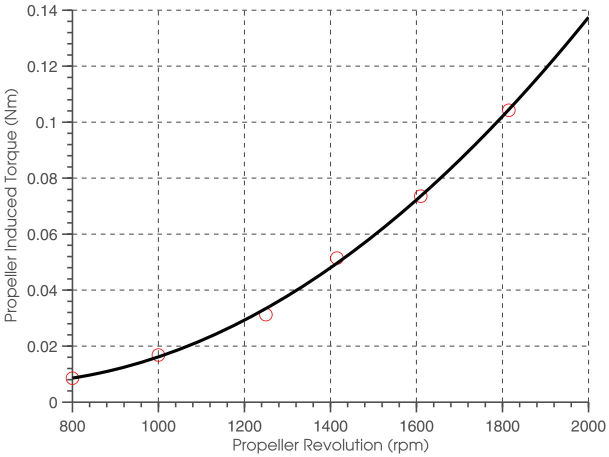


Fig. 3 Propeller induced torque versus propeller revolution

$$\frac{\phi}{\alpha} = \frac{-\left[\frac{(\frac{m}{M})lW}{I_{xx}-K_{\dot{p}}}\right]}{s^2 - \left[\frac{K_p}{I_{xx}-K_{\dot{p}}}\right]s - \left[\frac{(z_g-(\frac{m}{M})l)W}{l_{xx}-K_{\dot{p}}}\right]}. \tag{12}$$

By assigning the constant parameters k, a, and b to its corresponding coefficient respectively, (12) becomes:

$$\frac{\phi}{\alpha} = \frac{k}{s^2+as+b}. \tag{13}$$

4 System Identification

In this section, we estimate the three unknown parameters a, b and k of the linear second-order roll-axis model presented in (13). We also identify $K_{\dot{p}}$, K_{pp}, and l for nonlinear equation (11). Others parameter such as I_{xx}, y_g, z_g, m, M, W can either be measured directly or calculated through computer-aided design (CAD) softwares. Numerical values for those parameters are tabulated in Table 1.

Generally, we need to perform open-loop testing by changing α using a step function between $\pm 20^\circ$ and then record the roll response. Ideally, the test should be carried out while the AUV is maintaining constant thrust, depth and heading. This will minimize the coupling torque generated by those degrees of freedom. However, the open loop tests might pose danger to the operation of the AUV as we are testing some unknown behavior of the roll dynamics. A more natural choice would be to carry out the open-loop test while AUV is at rest in a water tank. This turns out to be sufficient to obtain a nominal model for the roll dynamics for the following reasons. First, in our model, we treat the propeller induced torque as a disturbance. So, whether the thruster is running or not, it is not included in the model.

Table 1 Model parameters

Calculated Parameters	Values	Identified Parameters	Values
I_{xx}	0.474 kg m^2	a	0.24
y_g	0 mm	b	5.21
z_g	-3.4 mm	k	-0.61
m	2.00 kg	$K_{\dot{p}}$	-0.08 kg m^2
M	61.41 kg	K_{pp}	-1.21 Nms^2
W	602.5 N	l	43.36 mm

Second, the roll dynamics model is derived under a decoupled assumption and therefore it is free from excitation from other axis. Third, the tank test underestimates the drag coefficient as the conning tower and the top fin are not fully submerged in the water. However it is better to underestimate the drag in our case, as higher drag will make the roll dynamics more stable. It will also ensure that the designed controller will also work properly when the vehicle is on the surface before it starts diving.

While the AUV is static in the tank, we command three step inputs of α (-20°, 0° and 20°) and observe how the roll responds to the step change of α. Sufficient time was allowed for the roll response to decay before another step change. The results are shown in Fig. 4. The simulated roll response is overlaid together with the experimental measured roll response. The result shows a good match between the two. The simulated roll is generated from the nonlinear model after the unknowns are identified. The 3 unknown parameters were identified by numerically minimizing the root mean square error ϕ_{rms} defined as:

$$\Phi_{rms} = \sqrt{\frac{\sum_{i=1}^{n}(\phi_i - \hat{\phi}_i)^2}{n}}. \tag{14}$$

where $\hat{\phi}$ is the simulated roll response and n is the number of samples. The Nelder-Mead simplex method was used to search for the optimal parameters set in least square sense.

It is important to note that the α is the command given to the servomotor. There is no instrument to measure the position of the rolling mass. So, some latency is expected between the commanded α and the actual α. We model the latency by a first order system with a time constant τ_{delay}. In order to identify the time constant, we perform a dynamic test by commanding α randomly between ±20° to obtain the response shown in Fig. 5. Similarly, the time constant is identified by minimizing ϕ_{rms}. The resultant transfer function in (13) becomes:

$$\frac{\phi}{\alpha} = \left(\frac{1}{\tau_{delay}s+1}\right)\left(\frac{k}{s^2+as+b}\right) \tag{15}$$

with $\tau_{delay} = 0.5\ s$.

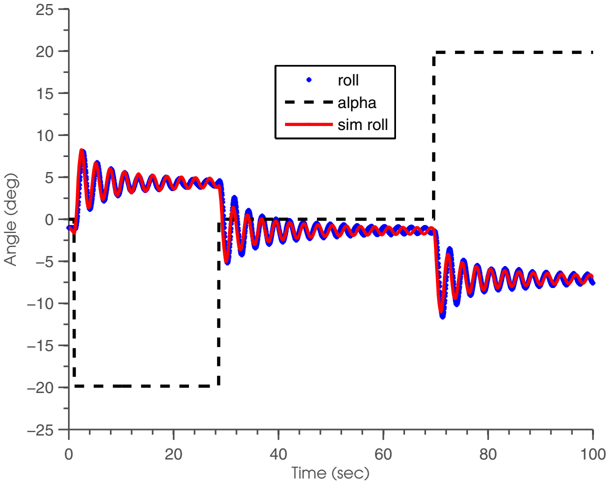


Fig. 4 Simulated and measured roll response under step input. The simulated roll response matched closely with the measured roll response despite small differences in amplitude and phase.

5 Controller Design

In this section, we design a Proportional-plus-Integral (PI) controller that stabilizes the AUV's roll motion. The PI controller is used to reduce the roll oscillation by increasing the damping of the system and at the same time maintain zero steady state error. The controller was synthesized base on root locus design (Fig. 6).

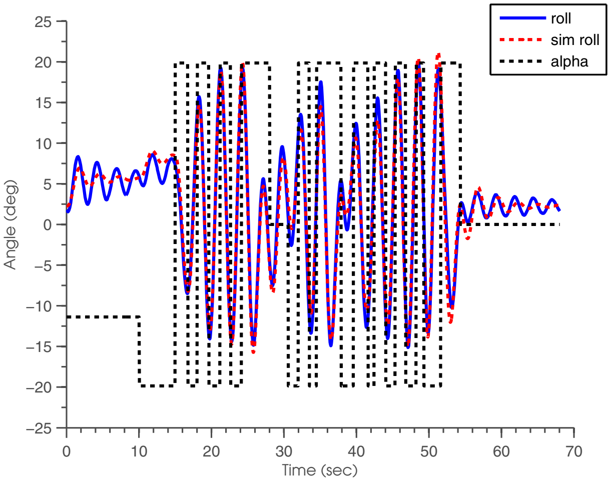


Fig. 5 Simulated and measured roll response under random input

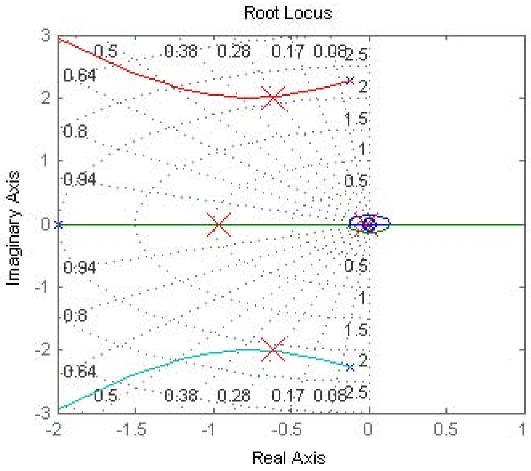


Fig. 6 Root locus plot for compensated system

The open loop transfer function has a pair of complex-conjugate *poles* close to the imaginary axis in the s-plane. This indicates the system is lightly damped with a damping ratio of 0.07. Fig. 6 also shows that the system is only stable for a small region of the root locus; it is stable for closed-loop gain range between $(0 < Kp < 8.50)$. The portion that is stable appears to be lightly damped as well. By increasing the gain, we bring the pair of complex-conjugate *poles* to a region of higher damping. However, the third *pole* moves closer to the right-half plane as the gain increases. As the *poles* are close to each other, we cannot analyze the system purely based on a second-order approximation. Instead, we simulate the nonlinear model and fine tune the controller gain using the simulation results. An ideal integral was added with a *zero* at 0.01. The fourth closed-loop *pole* is found at -0.0144,

Table 2 Open and Closed Loop Plants

	Open loop	Closed loop
Plant	$\frac{K}{(s+2)(s^2+0.24s+5.21)}$	$\frac{K(s-0.01)}{s(s+2)(s^2+0.24s+5.21)}$
K	-1.22	-6.10
Kp	-	5
Poles	$-0.12+2.2794i$	$-0.617+2.0077i$
	$-0.12+2.2794i$	$-0.617+2.0077i$
	-2	-0.9609
		-0.0144
Zeros	-	0.01
System Type	0	1

close enough to the *zero* to cause *pole-zero* cancellation. All *poles* and *zero* of the open and closed-loop plant are tabulated in Table 2. Integrator windup is avoided by preventing the integral term from accumulating above or below 20°.

6 Result and Discussion

The performance of the internal rolling mass in controlling the roll was first studied in a tank test and later at an open-field trial. For the tank test, we gave an impulse

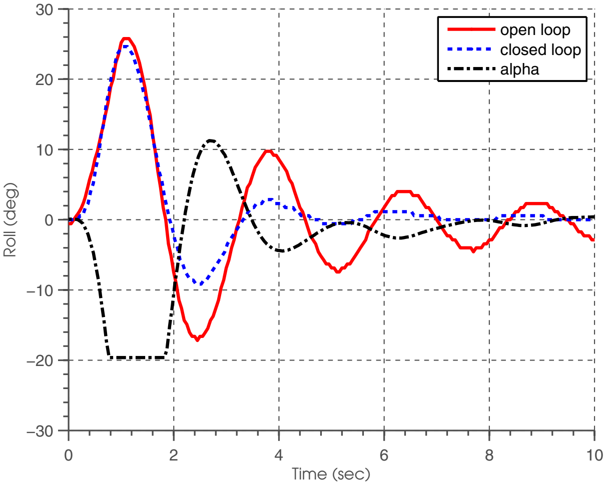


Fig. 7 Tank Test Result. The result shows that despite actuator saturation, the IRM mechanism manages to damp down the oscillation faster.

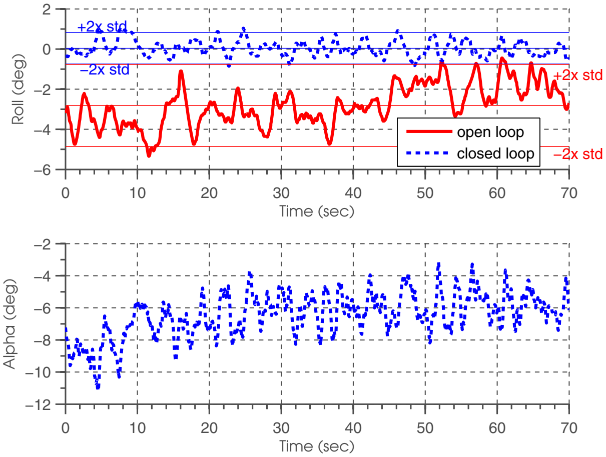


Fig. 8 Open Field Test Result

to the AUV by pushing AUV to roll to 25° and observed how the roll decays for open-loop and closed-loop control respectively. The results are shown in Fig. 7. The closed-loop response settle down within 4 seconds whereas the open-loop system takes more than 10 seconds to settle down. Fig. 7 also shows how the α changes with time in order to damp down the roll. For the open-loop test, α was kept at a constant 0°.

Fig. 8 shows the AUV's roll response during a constant 2 m diving mission at the speed of 1.4 m/s when traveling on a straight path. When the IRM mechanism was turned off (open loop), the AUV's roll response was oscillatory with standard deviation of 1.02°. On the contrary, when the IRM mechanism was turned on (closed loop), the oscillatory roll motion was damped. The moving mass rolls to negative alpha region to neutralize the induced propeller torque. The standard deviation of roll reduced to 0.393°. Table 3 summarizes the test results into two statistics: mean and standard deviation. Looking at the mean value, the IRM mechanism also made oscillation centered at zero angle. In short, the result shows that the IRM mechanism suppressed the unwanted roll oscillation to a smaller amplitude with center around zero.

Table 3 Open and Closed Loop Performance

	Open loop	Closed loop
Mean	-2.808 °	0.039 °
Standard Deviation	1.023 °	0.393 °

7 Conclusions

We demonstrated the use of an internal rolling mass (IRM) mechanism for active roll stabilization in a tail-thrusted and fin-controlled AUV. The mechanism was designed and implemented in the STARFISH AUV. A nonlinear model was first developed to describe the dynamics of the AUV's roll. The model was later linearized to obtain a transfer function for controller synthesis. The model's parameters were identified through open-loop testing in the water tank. A PI controller was then designed to increase the overall system damping and remove the steady state error. The capability of IRM to stabilize the roll motion was demonstrated in a tank test as well as through open-field tests.

References

[1] Alvarez, A., Caffaz, A., Caiti, A., Casalino, G., Gualdesi, L., Turetta, A., Viviani, R.: Folaga: a low-cost autonomous underwater vehicle combining glider and auv capabilities. Ocean Engineering 36(1), 24–38 (2009)

[2] Fossen, T.: Guidance and control of ocean vehicles, New York (1994)

[3] Kirkwood, W., Anderson, W.R., Kitts, C.: Fault tolerant actuation for dorado class, auvs. Instrumentation Viewpoint (8), 40–41 (2009)

[4] Koay, T., Tan, Y., Eng, Y., Gao, R., Chitre, M., Chew, J., Chandhavarkar, N., Khan, R., Taher, T., Koh, J.: Starfish–a small team of autonomous robotic fish. IJMS 40, 157–167 (2011)

[5] Leonard, N., Graver, J.: Model-based feedback control of autonomous underwater gliders. IEEE Journal of Oceanic Engineering 26(4), 633–645 (2001)

[6] McEwen, R., Streitlien, K.: Modeling and control of a variable-length auv. In: Proc. 12th UUST (2001)

[7] Petrich, J., Stilwell, D.: Robust control for an autonomous underwater vehicle that suppresses pitch and yaw coupling. Ocean Engineering (2010)

[8] Prestero, T.: Verification of a six-degree of freedom simulation model for the remus autonomous underwater vehicle. Master's thesis, Massachusetts Institute of Technology and Woods Hole Oceanographic Institution (2001)

[9] Singh, H., Whitcomb, L., Yoerger, D., Pizarro, O.: Microbathymetric mapping from underwater vehicles in the deep ocean. Computer Vision and Image Understanding 79(1), 143–161 (2000)

[10] Woock, P.: Deep-sea seafloor shape reconstruction from side-scan sonar data for auv navigation. In: 2011 IEEE OCEANS, pp. 1–7. IEEE, Spain (2011)

[11] Woolsey, C., Leonard, N.: Moving mass control for underwater vehicles. In: Proceedings of the 2002 American Control Conference, vol. 4, pp. 2824–2829. IEEE (2002)

Part V
Humanoid and Space

Human Biomechanical Model Based Optimal Design of Assistive Shoulder Exoskeleton

Marc G. Carmichael and Dikai K. Liu

Abstract. Robotic exoskeletons are being developed to assist humans in tasks such as robotic rehabilitation, assistive living, industrial and other service applications. Exoskeletons for the upper limb are required to encompass the shoulder whilst achieving a range of motion so as to not impede the wearer, avoid collisions with the wearer, and avoid kinematic singularities during operation. However this is particularly challenging due to the large range of motion of the human shoulder. In this paper a biomechanical model based optimisation is applied to the design of a shoulder exoskeleton with the objective of maximising shoulder range of motion. A biomechanical model defines the healthy range of motion of the human shoulder. A genetic algorithm maximises the range of motion of the exoskeleton towards that of the human, whilst taking into account collisions and kinematic singularities. It is shown how the optimisation can increase the exoskeleton range of motion towards that equivalent of the human, or towards a subset of human range of motion relevant to specific applications.

1 Introduction

Exoskeletons are a type of robot worn by the operator to provide physical assistance. These systems have the potential to significantly benefit numerous industrial and service applications such as nursing [19], agriculture [18], rehabilitation [13], and reduce fatigue or injury [6] in tasks like materials handling as depicted in Figure 1. Exoskeletons are commonly categorised into systems that assist either the upper or the lower limbs. Although applications exist which would benefit from both types, most examples that have reached commercialisation are for the lower limb. It is speculated that a factor contributing to this may be the additional challenges associated with designing an exoskeleton for the human shoulder, which has a large

Marc G. Carmichael · Dikai K. Liu
University of Technology, Sydney, Centre for Autonomous Systems
e-mail: {marc.carmichael,dikai.liu}@uts.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_17, © Springer International Publishing Switzerland 2015

range of motion and is described as the most complicated of the major articulations in the human body [7]. Activities utilising the upper limb require a large amount of dexterity, hence for an exoskeleton to be beneficial a design that does not impede the wearer's natural range of motion (ROM) is required. However designing an exoskeleton that achieves a human equivalent ROM is challenging as it is required to do so whilst encompassing the wearer, maintaining no collision with the wearer, and be satisfactorily far from kinematic singularity throughout its operation.

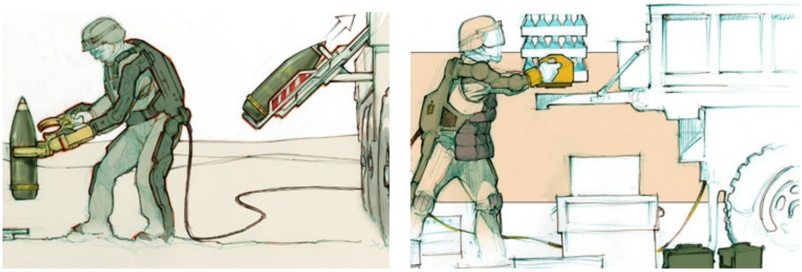


Fig. 1 Depiction of robotic exoskeletons assisting the upper limb in materials handling applications [8]. Copyright 2010 Raytheon Sarcos [17].

Robotic exoskeletons that encompass the shoulder exist in a number of kinematic variations in the literature with ranging levels of complexity and sophistication [9, 16, 1, 14, 3, 10]. When designing the shoulder mechanism a common approach is to focus on the location of the singularity and manually position it outside or at the edge of the desired workspace to maximise the usable range of motion [9, 16, 1, 2]. In this work we design a shoulder exoskeleton using an optimisation process that incorporates a biomechanical model of the human arm [12, 11]. With the human shoulder ROM defined by the biomechanical model, the design parameters of an exoskeleton are optimised using a genetic algorithm to maximise its ROM towards that of the human, whilst accounting for singularity and collisions. Additionally the optimisation is utilised with a subset of human ROM corresponding to workspace regions relevant to different tasks. This demonstrates how the presented optimisation is a useful tool for adapting new or existing exoskeleton designs to specific industrial or service applications.

2 Human Shoulder Range of Motion (ROM)

Maximising exoskeleton ROM towards that equivalent of the human shoulder requires that the ROM of the human shoulder first be defined. The human shoulder ROM is defined as the region of 3D space the humerus can be located. With translation of the shoulder not considered, the ROM is represented simply by all the orientations the humerus can reach. Many sources in the literature describe shoulder ROM, however these are often isolated to individual planar articulations. A general model of shoulder ROM representing all reachable orientations not limited to distinct articulations is required.

The biomechanical model developed by [12] represents the upper limb, including the shoulder complex, as a serial chain of rigid links connected by ideal joints. This model was developed to determine the reachable workspace of the human wrist, and includes equations describing the limits of each of the joints in the model. This allows the orientation of the humerus to be tested whether or not it is located within these limits, and hence is in a biomechanically reachable orientation. Although this shoulder model consists of five joints representing clavicle, scapula and humerus movement, only three coordinates are used to describe the orientation of the humerus relative to the torso. These are ϕ_A for abduction/adduction, ϕ_F for flexion/extension, and ϕ_R for medial/lateral rotation. Limits for these joints are shown in (1) taken from [12], where ϕ_{Am}, ϕ_{AM}, ϕ_{Fm}, ϕ_{FM}, ϕ_{Rm} and ϕ_{RM} are constants based on clinical measurements of the subject. These are intended to allow the reachable workspace of patients with upper limb impairments to be calculated. In this work we consider the person wearing the exoskeleton as not having an impairment and hence use the healthy values shown in Table 1 taken from [11] and adapted for the model [12].

$$\begin{aligned} \phi_A &= \left(\phi_{Am} \ , \ \phi_{AM}\right) \\ \phi_F &= (\phi_{Fm} + \frac{1}{3}\phi_A \ , \ \phi_{FM} - \frac{1}{6}\phi_A) \\ \phi_R &= (\phi_{Rm} + \frac{7}{9}\phi_A - \frac{1}{9}\phi_F + \frac{4}{9\pi}\phi_A\phi_F \ , \ \phi_{RM} + \frac{4}{9}\phi_A - \frac{5}{9}\phi_F + \frac{10}{9\pi}\phi_A\phi_F) \end{aligned} \tag{1}$$

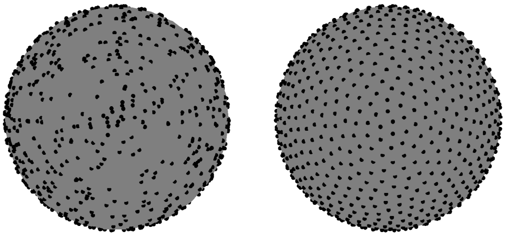
Table 1 Parameters used in the biomechanical model based on healthy human shoulder [11]

Abduction/Adduction		Flexion/Extension		Internal/External rotation	
ϕ_{Am}	ϕ_{AM}	ϕ_{Fm}	ϕ_{FM}	ϕ_{Rm}	ϕ_{RM}
−10°	170°	−60°	170°	−90°	60°

Representation of the humerus ROM is made by generating a large set of orientations the humerus can reach in the workspace. Later this set is used to evaluate the ROM of the exoskeleton based on the number of orientations the exoskeleton can also satisfactorily reach. One approach to generate this set is to recursively step through the range of each joint in the biomechanical model to produce a set of feasible humerus rotations. A problem with this approach is the resulting set of rotations are not evenly distributed which causes problems when quantifying the ROM that the exoskeleton can reach. Consider the case where the exoskeleton is unable to reach a region in the human's ROM. If this region so happens to contain a dense distribution of rotations in the set, then this would skew the calculated exoskeleton ROM. Hence it is essential that an even distribution of rotations be used when quantifying the ROM the exoskeleton can achieve.

To mitigate skewing of the ROM calculation a method that creates a more even distribution is implemented. A set of randomly distributed 3D rotations is created represented by unit quaternions. Since each quaternion contains four parameters with unit length, the set can be treated as points lying on the surface of a 4-dimensional unit sphere. Points on the sphere were constrained to only one hemisphere since points that are diametrically opposite from each other represent identical rotations. Once the random set of quaternions is created, a routine is implemented where the points on the sphere are repelled from each other. This is performed recursively whilst constraining the points to the surface of the sphere until a distribution considered as even is achieved. Figure 2 shows this process before and after using a 3-dimensional sphere equivalent for visualisation purposes.

Fig. 2 Randomly generated points on unit sphere. (a) before the even distribution procedure. (b) after the even distribution procedure. The same procedure is used on a 4D sphere to evenly distribute a set of quaternion rotations.

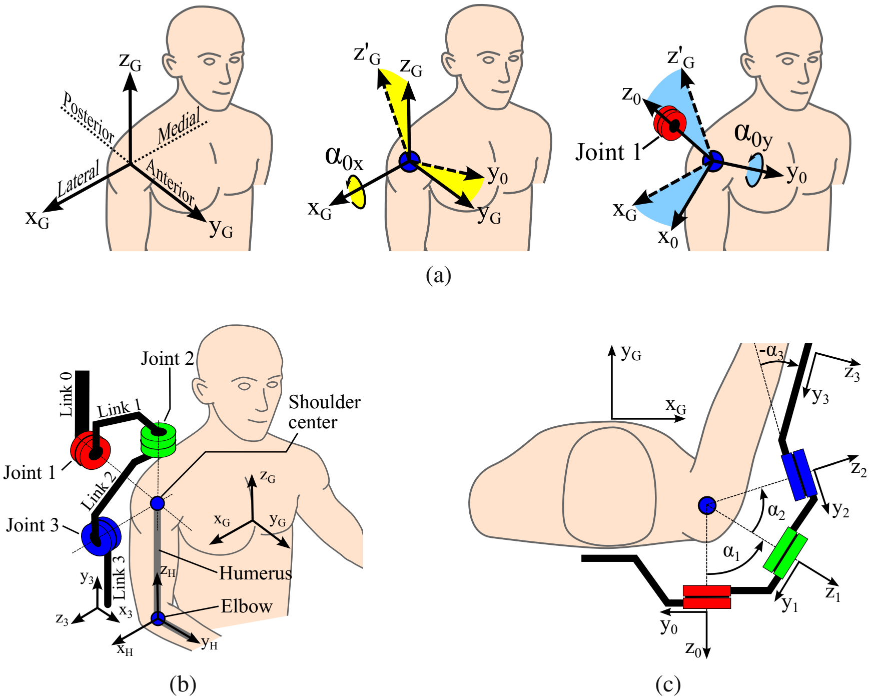


(a) Before routine (b) After routine

A set $\psi = \{R_{H1}, R_{H2}, \cdots, R_{Hn}\}$ containing 30,000 evenly distributed rotations is made. Each element in this set represents the orientation of the human humerus relative to the torso. A subset of ψ is created by utilising the biomechanical shoulder model to determine if each individual rotation is reachable by the human humerus. Inverse kinematics is performed to solve for the humerus abduction, flexion and rotation angles of the shoulder model, as detailed in the Appendix. If these angles are found to be within range according to (1) then we consider the rotation to be biomechanically feasible. Collision between the humerus and torso is also checked as this is not accounted for in the model [11]. The resulting subset ψ_H contains 5,227 humerus rotations which are determined as biomechanically feasible, and have on average a rotation of 7.6° between neighboring rotations in the set.

3 Exoskeleton Design

The exoskeleton design optimised in this work has a configuration similar to some described in the literature [9, 3]. It consists of three revolute joints interconnected by angled links such that their axes of rotation intersect to produce a 3 degree of freedom (DOF) spherical joint. The centre of rotation is coincident with the wearer's glenohumeral joint as shown in Figure 3. When optimising this design it is assumed the exoskeleton is worn by the user, hence link 0 is in a fixed location relative to the torso. Likewise the end effector is attached to the upper arm, hence link 3 is in a fixed location relative to the humerus. The human shoulder complex is not a pure spherical joint as it translates as well as rotates, however the translation is ignored as it is assumed differences in exoskeleton and human arm kinematics can be accommodated by passive compliance between the robot and the wearer [9]. Lastly we assume the radii of the joints and links to the shoulder's centre can be designed after optimisation in such a way that exoskeleton self collision can be avoided, therefore robot collision with itself is not considered during optimisation.



(a)

(b)

(c)

Fig. 3 (a) Location of exoskeleton joint 1 with respect to the human torso frame R_G, determined by design parameters α_{0x} and α_{0y}. (b) Shoulder exoskeleton with links and joints labelled. Coordinate frames for the human torso (R_G), humerus (${}^GR_H$), and link 3 (${}^GR_3$) are shown. (c) Relationship between the global torso frame R_G and exoskeleton link frames ${}^GR_0$, ${}^GR_1$, ${}^GR_2$, and ${}^GR_3$. The design parameters and joint angles shown in this subfigure are; $\alpha_{0x} = 90°$, $\alpha_{0y} = 0°$, $\theta_1 = 90°$, $\theta_2 = 0°$, $\theta_3 = 0°$

Since the mechanism forms a spherical joint the kinematics of the exoskeleton can be expressed solely by rotations. Rotation matrix R_G represents the global coordinate frame and is assigned to the human torso with x-axis directed laterally, y-axis anteriorly, and z-axis directed superiorly. Orientation of the human humerus is defined by rotation matrix ${}^{G}R_{H}$, with the superscript denoting that it is represented with respect to the global frame. The humerus frame has z-axis pointing from the elbow to the shoulder, y-axis pointing from the elbow to the wrist (if the elbow is bent at 90° right angle), and x-axis being their cross product to form a right-handed coordinate system. The relative rotation between R_H and the coordinate frame of link 3 remains fixed as shown in Figure 3b. Orientation of link 0 is represented by rotation matrix ${}^{G}R_0$ which is parameterised using two successive rotations, α_{0x} about the x-axis followed by α_{0y} about the y-axis (2). The convention of defining each joint axis as coinciding with the z-axis of the preceding link's frame is used, hence ${}^{G}R_0$ determines the location of joint 1.

$${}^{G}R_0 = \begin{bmatrix} \cos\alpha_{0y} & 0 & \sin\alpha_{0y} \\ \sin\alpha_{0x}\sin\alpha_{0y} & \cos\alpha_{0x} & -\sin\alpha_{0x}\cos\alpha_{0y} \\ -\cos\alpha_{0x}\sin\alpha_{0y} & \sin\alpha_{0x} & \cos\alpha_{0x}\cos\alpha_{0y} \end{bmatrix} \tag{2}$$

The coordinate frame for link 1 is located at its distal end, and hence relative to link 0 it is a rotation about its joint axis followed by another rotation due to the curvature of the link. With the joint rotating θ_1 degrees about the z-axis, followed by the link bending α_1 degrees about the x-axis, the frame of link 1 relative to link 0 is represented by equation (3) where $i = 1$. Rotations for links 2 and 3 are represented likewise with $i = 2$ and $i = 3$, respectively.

$${}^{i-1}R_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix} \tag{3}$$

3.1 Design Parameters

As the human maneuvers their upper limb the exoskeleton is required to position link 3 in the appropriate location relative to the humerus. This is achieved during operation by controlling the joint angles θ_1, θ_2 and θ_3. However from the previous equations it is obvious that parameters α_{0x}, α_{0y}, α_1, α_2 and α_3 also play an important role. These are the design parameters which will be optimised to maximise the exoskeleton ROM. Upper and lower bounds for these parameters can be set using appropriate rationale based on the exoskeleton's design. Parameters α_{0x} and α_{0y} determine the location of joint 1 relative to the torso. We limit α_{0x} to between $-45°$ and 90° as outside these bounds joint 1 is likely to interfere with the torso or the arm during anterior and posterior reaching motions, as shown in Figure 4a. Similarly we limit α_{0y} to between 0° and 45°, as outside these bounds joint 1 is likely to interfere with the neck or the arm during lateral reaching motions as shown in Figure 4b.

Parameters α_1 and α_2 define the bend arc angle in links 1 and 2. As these angles approach 0° or 180° the joint axes become aligned and a kinematic singularity occurs, as shown in Figure 4c. We limit these values to be between 20° and 160°.

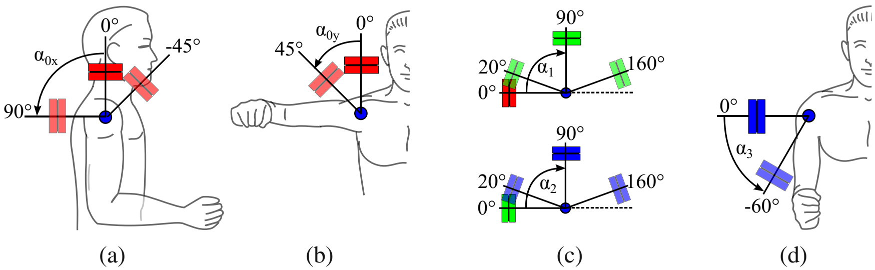


Fig. 4 Exoskeleton design parameters. (a) α_{0x} defines the anterior/posterior location of joint 1. (b) α_{0y} defines the lateral/medial location of joint 1. (c) α_1 and α_2 define the arc angle of links 1 and 2. (d) α_3 defines the angle between joint 3 and the humerus.

We constrain joint 3 to be located medially to the humerus because if positioned anteriorly it will collide with the torso during medial shoulder rotation. If located posteriorly it will collide during lateral rotation, hence locating it laterally to the humerus is the most practical. With joint 3 confined to this plane it is possible to define an angle between it and the humerus, as is done by examples in the literature [9, 3]. Parameter α_3 defines the angle that joint 3 makes with the humerus which we allow to be in the range $-60°$ to $0°$ as shown in Figure 4d. All of the design parameters are arranged into set $DP = \{\alpha_{0x}, \alpha_{0y}, \alpha_1, \alpha_2, \alpha_3\}$ which are summarised in Table 2 along with their upper and lower bounds.

Table 2 Summary of exoskeleton design parameters and their bounds used for optimisation

Symbol	Description	Lower bound	Upper bound
α_{0x}	Link 0 x-axis rotation	$-45°$	90°
α_{0y}	Link 0 y-axis rotation	0°	45°
α_1	Link 1 bend arc angle	20°	160°
α_2	Link 2 bend arc angle	20°	160°
α_3	Link 3 bend arc angle	$-60°$	0°

3.2 Exoskeleton Range of Motion (ROM)

The human humerus cannot achieve every possible orientation in 3D space as it is subject to its own physiological constraints, hence the exoskeleton is only required to reach the orientations within the ROM of the human. The ROM of the exoskeleton is quantified by the percentage of the human ROM (i.e. ψ_H) which it can successfully reach. Three criteria determine whether or not the exoskeleton can reach a humerus orientation for a given set of design parameters. Firstly we check if an inverse kinematic solution exists. If a solution does exist, we check if this solution causes any collision between the exoskeleton and the wearer. Lastly we also evaluate if the robot is considered too close to the singularity condition.

3.2.1 Inverse Kinematic Solution

For each humerus orientation in set ψ_H we solve for the exoskeleton joint angles θ_1, θ_2, θ_3. Because the position of joints 1 and 3 are defined by the exoskeleton design parameters and the humerus orientation, we can check if an inverse kinematic solution exists by whether of not it is able to reach the angular distance required between these two joints. The z-axis of ${}^{G}R_0$ is the axis of joint 1 which we define as $\mathbf{z}_0$. Similarly the z-axis of ${}^{G}R_2$ is the axis of joint 3 which we define as $\mathbf{z}_2$. The maximum angle the exoskeleton can produce between its first and third joint axes is $\alpha_1 + \alpha_2$ when $\theta_2 = 0$. The minimum angle is $|\alpha_1 - \alpha_2|$ when $\theta_2 = \pi$. From the angle required (i.e. $\cos^{-1}(\mathbf{z}_0 \cdot \mathbf{z}_2)$) we use (4) and (5) to determine if the exoskeleton can reach a specific humerus orientation for a given set of exoskeleton design parameters.

$$\mathbf{z}_0 = \begin{bmatrix} \sin\alpha_{0y} \\ -\sin\alpha_{0x}\cos\alpha_{0y} \\ \cos\alpha_{0x}\cos\alpha_{0y} \end{bmatrix} \qquad \mathbf{z}_2 = R_H \begin{bmatrix} \cos\alpha_3 \\ 0 \\ \sin\alpha_3 \end{bmatrix} \tag{4}$$

$$c_1(R_H, DP) = \begin{cases} 0, & \text{if } |\alpha_1 - \alpha_2| < \cos^{-1}(\mathbf{z}_0 \cdot \mathbf{z}_2) < \alpha_1 + \alpha_2 \\ 1, & \text{otherwise} \end{cases} \tag{5}$$

If a solution does exist, it can be solved using inverse kinematics methods. However we need to consider that when a solution does exist there are typically two solutions, i.e $\pm\theta_2$. To handle this we define the exoskeleton as operating in either one of two modes. Mode 1 has θ_2 in the range $0 < \theta_2 < \pi$, and mode 2 the range $-\pi < \theta_2 < 0$. As the robot transitions between these modes ($\theta_2 = 0$ or $\theta_2 = \pi$) the exoskeleton becomes singular. During operation the robot should operate solely in one mode. By specifying which mode the exoskeleton operates in allows kinematic solutions that are consistent during operation to be analysed during optimisation.

3.2.2 Collision

Every inverse kinematic solution is checked for collisions between the exoskeleton and the human. If a collision is present, then we identify that the exoskeleton is unable to reach the corresponding humerus orientation R_H for the given design parameters DP. This is formalised by equation (6).

$$c_2(R_H, DP) = \begin{cases} 0, & \text{if no collision present} \\ 1, & \text{if collision present} \end{cases} \tag{6}$$

To implement this requires a fast and efficient method for performing collision checking between the exoskeleton and the wearer during the optimisation. Methods that represent both the exoskeleton and human as solid meshes and then perform mesh-mesh collision checking were found to be too computationally expensive. Instead a simpler method was used based on a sphere with its centre at the shoulder, as seen in Figure 5. On this sphere two regions were defined where if the exoskeleton was located it is likely to be colliding with the human. The first region represented collision between the exoskeleton and the upper arm. A point was created where the line connecting the shoulder and elbow intersect the sphere, hence this point varied for different humerus orientations. Any points on the sphere what are within a defined distance to this point are said to be within the region and hence colliding with the humerus. The second region represented collision between the exoskeleton and the torso. This region was defined using two points on the sphere which remained static regardless of humerus orientation. Any point on the sphere within a defined distance to either of these two points, or to the shortest arc joining them, were said to be within this region and hence colliding with the torso. The joint axes of the exoskeleton are projected onto the sphere surface, and if they are located within either of these regions then the exoskeleton is deemed to be colliding. To account for collisions of the links, mid points between each joint axis were also checked. This method was efficient to calculate and hence feasible for use in the optimisation.

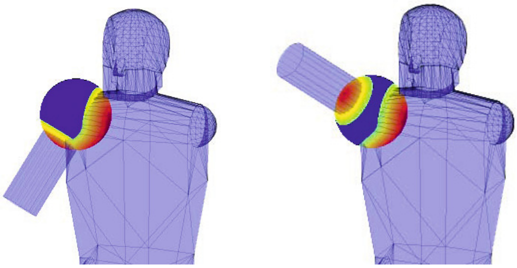


Fig. 5 Robot-human collision model defines regions on a sphere about the shoulder in which if the exoskeleton joints or links were located, they are likely to collide with the torso or upper arm

3.2.3 Singularity

The exoskeleton enters a kinematic singularity when all the joint axes lay in a plane, reducing the workspace degrees of freedom from three to two. For each inverse kinematic solution the singularity condition is checked. Several singularity measures based on the manipulability of the robot have been developed, many related to the Jacobian [15]. The 3×3 Jacobian matrix **J** relating joint velocity to angular workspace velocity is calculated from the axis of rotation of each joint using (7).

$$\mathbf{J} = \left[\begin{array}{c|c|c} {}^{G}R_0 \begin{bmatrix}0\\0\\1\end{bmatrix} & {}^{G}R_0\,{}^{0}R_1 \begin{bmatrix}0\\0\\1\end{bmatrix} & {}^{G}R_0\,{}^{0}R_1\,{}^{1}R_2 \begin{bmatrix}0\\0\\1\end{bmatrix} \end{array} \right] \tag{7}$$

We define the eigenvalues of $\mathbf{J}$ as λ_1, λ_2 and λ_3. The square root of the minimum eigenvalue corresponds to the minimum workspace velocity that can be achieved by a unit joint velocity vector, which at singular configurations reduces to zero [15]. The singularity of each inverse kinematic solution is analysed by comparing the minimum eigenvalue square root with a threshold value ε. If it is less than this threshold then it is considered as being too close to singularity and hence R_H cannot be reached for the given set of design parameters DP, formalised by equation (8).

$$c_3(R_H, DP) = \begin{cases} 0, & \text{if } \min\left(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \sqrt{\lambda_3}\right) > \varepsilon \\ 1, & \text{if } \min\left(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \sqrt{\lambda_3}\right) \leq \varepsilon \end{cases} \tag{8}$$

4 Optimisation Model and Results

To find the optimal design we create a fitness function to calculate how much of the human ROM the exoskeleton design can reach. Each element in the humerus orientation set ψ_H is tested for validity based on inverse kinematic solution (5), collision (6), and singularity (8). Using (9) a subset ψ_E is created containing the valid orientations the exoskeleton can reach with all three conditions satisfied. The ROM of the exoskeleton with respect to the humerus is calculated by the number of elements in ψ_E relative to ψ_H (10). A fitness function (11) is defined which approaches zero as the exoskeleton ROM approaches 100%. Optimisation of the fitness function was implemented in MATLAB using a genetic algorithm (GA) in the Global Optimization Toolbox. A population size of 500 was used with the inputs being the bounded design parameters detailed in Section 3.1.

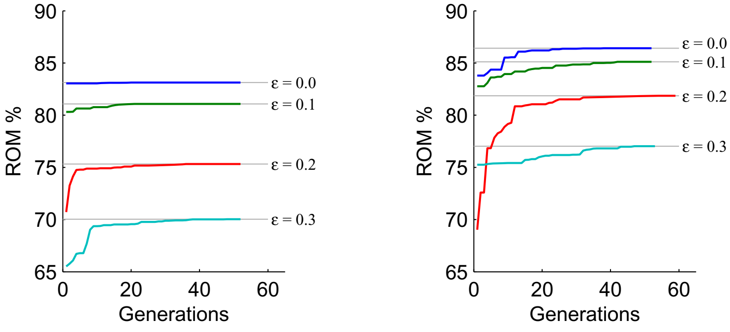
$$\psi_E = \{\psi_H \mid c_1(\psi_H, DP) = 0,\ c_2(\psi_H, DP) = 0,\ c_3(\psi_H, DP) = 0\} \tag{9}$$

$$\text{ROM} = 100\% \times \frac{|\psi_E|}{|\psi_H|} \tag{10}$$

$$\text{Minimise } f(DP), \text{ where } f(DP) = |\psi_H| - |\psi_E| \tag{11}$$

4.1 ROM Optimisation Results

Optimisation was performed with the singularity threshold ε set at 0, 0.1, 0.2 and 0.3. This was repeated with the inverse kinematic solution performed in both mode 1 and mode 2, as described in Section 3.2.1. Figure 6 shows the best ROM result calculated in the population versus generation during optimisation. The corresponding optimal design parameters are shown in Table 3. It is seen that the optimised ROM result decreases as ε is increased, which is expected since a larger ε puts more stringent constraints on the singularity condition. Comparison of the results indicate that mode 2 is the preferred kinematic solution, obtaining a greater ROM compared to mode 1 across all ε values tested.



(a) Mode 1

(b) Mode 2

Fig. 6 Optimisation results of the best ROM calculated by the GA versus generation. (a) Inverse kinematic solution mode 1. (b) Inverse kinematic solution mode 2.

Table 3 Optimised design parameters and corresponding calculated ROM for both inverse kinematic solution modes and the four singularity threshold (ε) values analysed

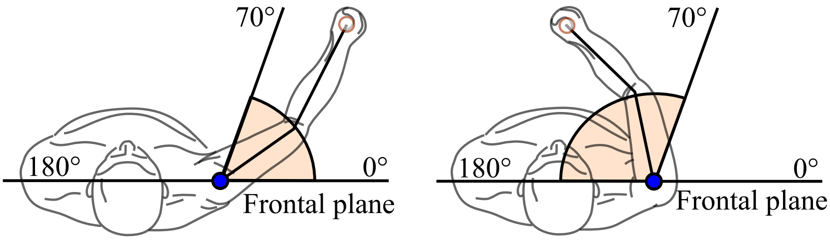
Kinematic solution	ε	α_{0x}	α_{0y}	α_1	α_2	α_3	ROM
Mode 1	0.0	32.6°	26.9°	45.8°	39.8°	0.0°	83.13%
Mode 1	0.1	27.2°	21.8°	47.3°	39.8°	0.0°	81.08%
Mode 1	0.2	10.2°	13.2°	48.8°	49.2°	0.0°	75.32%
Mode 1	0.3	8.8°	19.4°	55.0°	69.3°	−31.1°	70.04%
Mode 2	0.0	71.5°	32.3°	63.8°	71.1°	−25.7°	86.42%
Mode 2	0.1	68.8°	32.2°	64.1°	72.0°	−23.3°	85.12%
Mode 2	0.2	70.5°	33.6°	65.2°	75.1°	−28.3°	81.86%
Mode 2	0.3	67.7°	35.7°	67.5°	80.0°	−38.7°	77.02%

As a comparison an exhaustive method was implemented using a 10° step size to search through design parameter combinations. This was done for kinematic mode 2 with $\varepsilon = 0$. This approach took around 15.5 hours to complete using an 8 core high performance computer, much longer than the GA which took approximately 175 minutes using the same computer setup. The best result using the exhaustive search was a ROM of 83.7%, less than the 86.42% result using the GA. This increases our confidence that the GA found a solution close to the global optimum.

4.2 Task-Specific Optimisation

In specific applications having an exoskeleton reach the entire human ROM may not be necessary. For tasks that only require a small part of this workspace, it could be beneficial to maximise the exoskeleton's ROM in this smaller subspace relevant to the task. This may allow a large percentage of the ROM subset to be reached, while allowing the optimisation to achieve a design with better secondary characteristics, for example more stringent singularity constraints. We demonstrate this by dividing the frontal workspace into two areas as shown in Figure 7. We can imagine that these two areas corresponds to two different tasks. Task A has the humerus between 0° and 70° in the anterior-lateral workspace as shown in Figure 7a. Task B has the humerus between 70° and 180° in the anterior-medial workspace as shown in Figure 7b. Subsets of the human ROM are then created by extracting humerus orientations that lay within these two areas. We apply the optimisation to maximise the exoskeleton ROM towards these two humerus ROM subsets.

Results for the task-specific optimisation, calculated with $\varepsilon = 0.3$ and kinematic mode 2, are shown in Table 4. The ROM achieved was 93.57% and 99.67% for the tasks A and B respectively. Even with a stringent singularity threshold of $\varepsilon = 0.3$ the ROM obtained for these two specific tasks is much larger than the ROM obtained when attempting to achieve the equivalent to a complete human ROM, which was 86.42% with the most relaxed singularity constraint, i.e. $\varepsilon = 0$.



(a) Task A (b) Task B

Fig. 7 Two subsets of the humerus ROM are created by dividing the frontal workspace into regions corresponding to different conceptual tasks

Table 4 Optimised design parameters and corresponding calculated ROM when optimising the exoskeleton design for a subset of human ROM corresponding to two different tasks. Results were calculated for inverse kinematic solution mode 2 and singularity threshold $\varepsilon = 0.3$.

Task	α_{0x}	α_{0y}	α_1	α_2	α_3	ROM
A	90.0°	31.2°	59.6°	63.4°	−39.8°	93.57%
B	75.1°	45.0°	67.3°	82.4°	−27.7°	99.67%

5 Conclusion

The optimisation of the exoskeleton design was shown to maximise its ROM towards that achievable by the natural human humerus. The method used allowed factors such as collisions between exoskeleton and human, as well as an adjustable threshold on kinematic singularity to be considered during the optimisation. Unlike the commonly used approach of manually optimising a shoulder exoskeleton design by positioning the singularity, the optimisation method automatically determined the optimal singularity location, as well as additional design parameters such as the arc angle of the links. It was also demonstrated how the exoskeleton design can be optimised considering a subset of human ROM relevant to specific tasks. This highlights how the presented optimisation method can be a useful tool for the creation or modification of new and existing exoskeleton designs for use in specialised industrial or service applications.

For the optimisation to be made feasible several simplifications were made, for example the assumption of the shoulder behaving as a pure spherical joint. Using a similar optimisation framework as the one presented, more sophisticated methods can be incorporated. Future work will look at extending the optimisation using more advanced methods such as mesh-mesh approaches for realistic collision checking, as well as applying the optimisation on alternative exoskeleton designs. Our goal is to develop an upper limb exoskeleton platform with a large ROM for researching new assistive paradigms [5, 4].

Appendix – Biomechanical Model Inverse Kinematics

For a given humerus orientation defined by rotation matrix R_H we solve for the corresponding joint angles ϕ_A, ϕ_F and ϕ_R for the biomechanical shoulder model [12]. There actually exists two solutions for these angles that provide the specified humerus orientation.

Solution 1:	*Solution 2:*
$\phi_A = \text{atan2}\left(-R_{H13}, R_{H33}\right)$	$\phi_A = \text{atan2}\left(R_{H13}, -R_{H33}\right)$
$\phi_F = \text{asin}\left(-R_{H23}\right)$	$\phi_F = \pi - \text{asin}\left(-R_{H23}\right)$
$\phi_R = \text{atan2}\left(-R_{H21}, R_{H22}\right)$	$\phi_R = \text{atan2}\left(R_{H21}, -R_{H22}\right)$

To determine if R_H is a feasible orientation, we solve for both solutions. Then, if either solution is found to satisfy equation (1) then we consider R_H as a feasible humerus orientation. R_{Hij} is the element in row i and column j.

References

1. Ball, S., Brown, I., Scott, S.: Medarm: a rehabilitation robot with 5dof at the shoulder complex. In: 2007 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pp. 1–6 (2007), doi:10.1109/AIM.2007.4412446

2. Carignan, C., Liszka, M., Roderick, S.: Design of an arm exoskeleton with scapula motion for shoulder rehabilitation. In: Proceedings. 12th International Conference on Advanced Robotics, ICAR 2005, pp. 524–531 (2005)
3. Carignan, C., Tang, J., Roderick, S.: Development of an exoskeleton haptic interface for virtual task training. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009, pp. 3697–3702 (2009)
4. Carmichael, M., Liu, D.: Estimating physical assistance need using a musculoskeletal model. IEEE Transactions on Biomedical Engineering 60(7), 1912–1919 (2013)
5. Carmichael, M.G., Liu, D.: Towards using musculoskeletal models for intelligent control of physically assistive robots. In: 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society EMBC, pp. 8162–8165 (2011)
6. Carmichael, M.G., Liu, D., Waldron, K.J.: Investigation of reducing fatigue and musculoskeletal disorder with passive actuators. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2481–2486 (2010)
7. Engin, A.E.: On the biomechanics of the shoulder complex. Journal of Biomechanics 13(7), 575–590 (1980)
8. Hanlon, M.: Raytheon XOS 2: second generation exoskeleton (2010), http://www.gizmag.com/raytheon-significantly-progresses-exoskeleton-design/16479/ (viewed September 1, 2013)
9. Johnson, G.R., Carus, D.A., Parrini, G., Scattareggia Marchese, S., Valeggi, R.: The design of a five-degree-of-freedom powered orthosis for the upper limb. Proc. Inst. Mech. Eng. H 215(3), 275–284 (2001)
10. Klein, J., Spencer, S., Allington, J., Bobrow, J., Reinkensmeyer, D.: Optimization of a parallel shoulder mechanism to achieve a high-force, low-mass, robotic-arm exoskeleton. IEEE Transactions on Robotics 26(4), 710–715 (2010)
11. Klopčar, N., Tomšič, M., Lenarčič, J.: A kinematic model of the shoulder complex to evaluate the arm-reachable workspace. J. Biomech. 40(1), 86–91 (2007)
12. Lenarčič, J., Umek, A.: Simple model of human arm reachable workspace. IEEE Transactions on Systems, Man and Cybernetics 24(8), 1239–1246 (1994)
13. Lo, H.S., Xie, S.Q.: Exoskeleton robots for upper-limb rehabilitation: state of the art and future prospects. Med. Eng. Phys. 34(3), 261–268 (2012)
14. Mihelj, M., Nef, T., Riener, R.: Armin ii - 7 dof rehabilitation robot: mechanics and kinematics. In: 2007 IEEE International Conference on Robotics and Automation, pp. 4120–4125 (2007)
15. Murray, R.M., Li, Z., Sastry, S.S.: A Mathematical Introduction to Robotic Manipulation, 1st edn. CRC Press (1994)
16. Perry, J., Rosen, J., Burns, S.: Upper-limb powered exoskeleton design. IEEE/ASME Transactions on Mechatronics 12(4), 408–417 (2007)
17. Raytheon: Time magazine names the XOS 2 exoskeleton most awesomest invention of 2010 (2010), http://www.raytheon.com/newsroom/technology/rtn08_exoskeleton/
18. Toyama, S., Yamamoto, G.: Development of wearable-agri-robot - mechanism for agricultural work. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009, pp. 5801–5806 (2009)
19. Yamamoto, K., Ishii, M., Noborisaka, H., Hyodo, K.: Stand alone wearable power assisting suit - sensing and control systems. In: 13th IEEE International Workshop on Robot and Human Interactive Communication, ROMAN 2004, pp. 661–666 (2004)

Lunar Micro Rover Design for Exploration through Virtual Reality Tele-operation

Nathan Britton, Kazuya Yoshida, John Walker, Keiji Nagatani,
Graeme Taylor, and Loïc Dauphin

Abstract. A micro rover, code-named Moonraker, was developed to demonstrate the feasibility of 10kg-class lunar rover missions. Requirements were established based on the Google Lunar X-Prize mission guidelines in order to effectively evaluate the prototype. A 4-wheel skid steer configuration was determined to be effective to reduce mass, maximize regolith traversability, and fit within realistic restrictions on the rover's envelope by utilizing the top corners of the volume.

A static, hyperbolic mirror-based omnidirectional camera was selected in order to provide full 360° views around the rover, eliminating the need for a pan/tilt mechanism and motors. A front mounted, motorless MEMS laser scanner was selected for similar mass reduction qualities. A virtual reality interface is used to allow one operator to intuitively change focus between various narrow targets of interest within the wide set of fused data available from these sensors.

Lab tests were conducted on the mobility system, as well as field tests at three locations in Japan and Mauna Kea. Moonraker was successfully teleoperated to travel over 900m up and down a peak with slopes of up to 15°. These tests demonstrate the rover's capability to traverse across lunar regolith and gather sufficient data for effective situational awareness and near real-time tele-operation.

Nathan Britton · Kazuya Yoshida · John Walker · Keiji Nagatani
Tohoku University, Sendai, Miyagi, Japan
e-mail: {nathan,yoshida}@astro.mech.tohoku.ac.jp

Graeme Taylor
International Space University, Strasbourg, Alsace, France

Loïc Dauphin
ENSEIRB-MATMECA, Talence, France

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_18,

1 Introduction

A case study for a low cost lunar exploration mission is presented in this section to establish a general set of requirements for a lunar micro rover. The case study is based on the Google Lunar X-Prize (GLXP) and the goals of GLXP team Hakuto.

1.1 Google Lunar X-Prize

The GLXP is a privately funded competition to land a rover on the surface of the Moon, with prize money of $30 million USD available to privately funded teams who accomplish certain pre-determined goals, primarily to traverse 500m across the lunar surface and transmit HD video and photographs to earth. The GLXP requirements are considered essential minimum capabilities for the design of a micro-rover and for performance assessments.

Tohoku University's Space Robotics Lab (SRL) is in partnership with Hakuto, an official team contending for the GLXP. Hakuto has exploration goals in addition to the minimum GLXP requirements, detailed in Sec. 1.2, and provides the SRL with requirements based on the methods for launch, trans-lunar injection, and landing.

1.2 Landing Site and Exploration Target

In 2009, Haruyama et al. discovered evidence of collapsed lava tube skylights from JAXA's SELENE images of the surface of the moon[2]. Since then, many potential skylights have been discovered from various satellite imagery. The landing target is one of these potential skylights in the region known as Lacus Mortis, or "Lake of Death", south of Mare Frigoris in the northern, Earth-facing hemisphere of the Moon.

Team Hakuto plans to land within 500m of the skylight, which can be found at 44.95°N, 25.61°E, just south of a rille known as Rimae Bürg. Based on NASAs Lunar Reconnaissance Orbiter (LRO) images, the skylight is just under 400m in diameter, with the south-eastern edge of the pit collapsed, forming a natural ramp from the lunar surface to the cavern floor (Fig. 1).

Data from the Lunar Orbiter Laser Altimeter (LOLA) instrument of LRO also exist for this region[3]. The data is too sparse to develop a full digital elevation map, but does suggest a minimum average slope angle of 13°(Fig. 1 and Table 1). Tests with lunar regolith simulants have found that the angle of repose (maximum stable angle of a granular material) of lunar mare regolith can be up to 45-50° in Earth gravity and as high as 58° in lunar gravity[1].

The slope of the skylight ramp is likely varied, but *cannot be estimated more precisely than between 13° and 58°*; from safe to highly hazardous. There is insufficient data to determine a distribution within this range, therefore the feasibility of an unassisted descent must be decided on-site. In order to maximize the probability for and reduce the risks of a potential extended mission into the skylight, designing for stability during steep-slope traversal is a high priority.

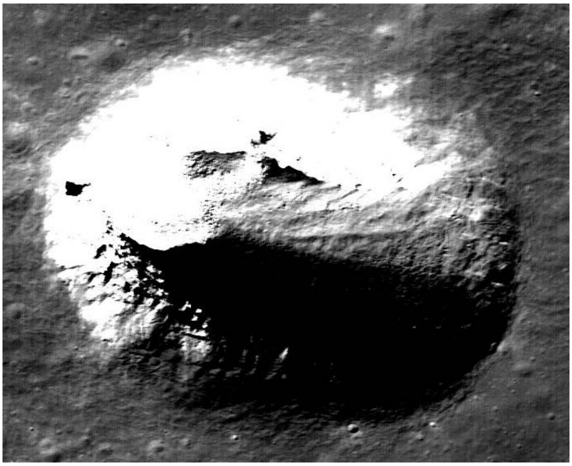


Fig. 1 The target skylight near Rimae Bürg, with the collapsed eastern wall clearly visible [4]. LOLA data (Table 1) indicate that the altitude at the eastern edge is -2042m, and a point in the center of the skylight, 150m from the edge, is -2077m altitude. 35m of depth across 150m indicates that the minimum average angle of the slope can be estimated at 13.13°.

Table 1 Lunar Orbiter Laser Altimeter (LOLA) measurements in the vicinity of the lava tube entrance[3]. One reading from the center of the pit suggests a minimum depth of 35m.

#	**Longitude**	**Latitude**	**Altitude**	**Estimated Location**
1	25.6222222	44.9595561	-2043.09	Vicinity East of the skylight
2	25.6222346	44.957638	-2042.38	Vicinity East of the skylight
3	25.6226786	44.9585525	-2042.14	Vicinity East of the skylight
4	25.616147	44.9562664	-2040.64	Southern edge of the skylight
5	25.6161021	44.9601059	*-2077.34*	*Center of the skylight*, 150m from #2
6	25.6165264	44.962933	-2055.96	Northern edge of the skylight

1.3 Communication Architecture

Hakuto plans to use the lander hardware to support a fixed communication relay station to the Earth. By eliminating the need for the micro-rover to establish a direct connection to the Earth, a greater portion of the mass budget can be alloted to mobility and imaging subsystems and other payloads. This architecture has benefits which apply to potential missions beyond the scope of the GLXP. In any situation where a permanent base station is nearby the micro-rover's tasked operation zone, this architecture is effective for reducing mass.

For the prototype and field tests discussed in this paper, a laptop was used to emulate the function of a stationary lander, and 802.11 wireless networks used for communication. A lower frequency radio communication system with lower bandwidth is planned for the flight model, but not included in this paper. For this particular case study, should the traversal into the skylight be successful, fully autonomous exploration of the skylight is necessary due to the communication blackout; however full automation experiments are also not considered in this report.

1.4 Requirements

- Mass limit: **10kg**
- Stowed volume envelope: **50cm(length) x 40cm(width) x 30cm (height)**
- Use of a nearby base station as relay to Earth
- Travel at least *500m* across the surface
- Climb as steep an incline of regolith as possible (*minimum 13*°)
- Ability to take *self portraits* and display relevant logos
- Full 360 degree HD panoramic stills: *0.3mrad/pixel*, 8bit color

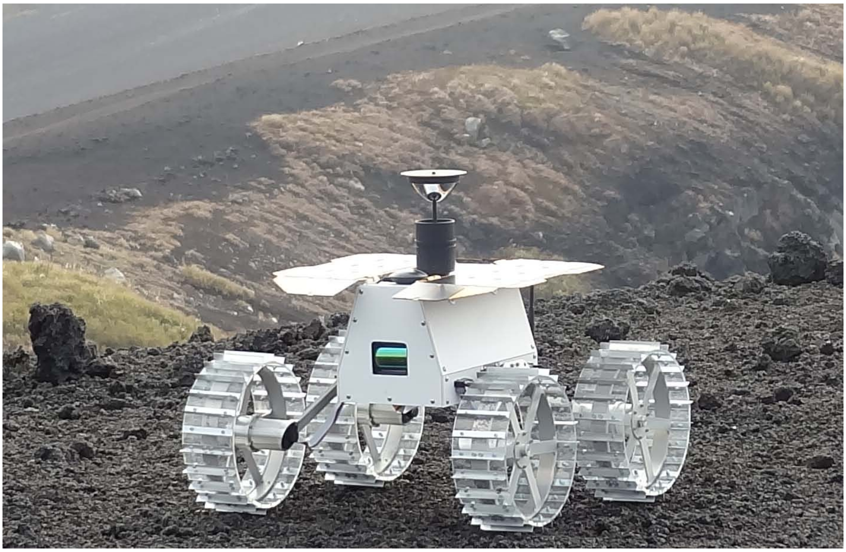


Fig. 2 Moonraker traversing over a mountain at Izu Oshima (Sec. 3)

2 Testbed Moonraker

Based on the case study mission and requirements, a micro rover prototype codenamed Moonraker was constructed for Earth-based testing and validation. In order to meet the strict size constraints, redundancy is reduced, with greater focus on reliability to mitigate risk. Non-essential actuation points were removed where possible to minimize mass, power consumption, and the total number of failure modes. The final mass and power budgets can be seen in Table 2. The resulting system is unique among similar micro-rover missions, such as the 1997 Mars Pathfinder rover component Sojourner, particularly due to the large wheel-size to system-size ratio.

Moonraker uses an omnicamera, giving full 360° by 80° views of the surrounding environment. A MEMS-based laser range finder (LRF) is equipped on the front, to measure accurate 3D positions of any imminent obstacles. These data, along with an IMU, are fused together in a 3D environment for viewing with a virtual reality headset. The mobility system consists of four 20cm wheels, for a large wheel-size to body-size ratio. Each wheel is independently controlled by 12W motors; maneuvers are conducted through skid steering. The wheels are stowed above the rover body during transit to maximize grouser length allowance.

As indicated in Table 2, the thermal subsystem and solar power generation designs were unimplemented in the terrestrial prototype discussed in this report. Aluminum solar panel mockups, identical in surface area to the flight system design, are used to properly obstruct the environment around the wheels as in a real lunar mission (Fig.2).

Table 2 Moonraker mass and power budgets. The mass of Moonraker during tests was 8107g. *indicates estimations based on unimplemented designs.

Subsystem	Target Mass	Actual Mass		Nominal Power
Mobility	25%	24.6%	2462g	30W
Power	20%	18.9%*	1890g*	
Structure+Thermal	20%	20.3%*	2027g*	
Sensors	10%	9.7%	965g	5W
Avionics+Comms	5%	3.6%	363g	8W
Payload for Sale	5%	8.9%	893g*	
Margin/Cables	15%	14.9%	1493g	
Total	100%	(81%)	10000g*(8107g)	43W

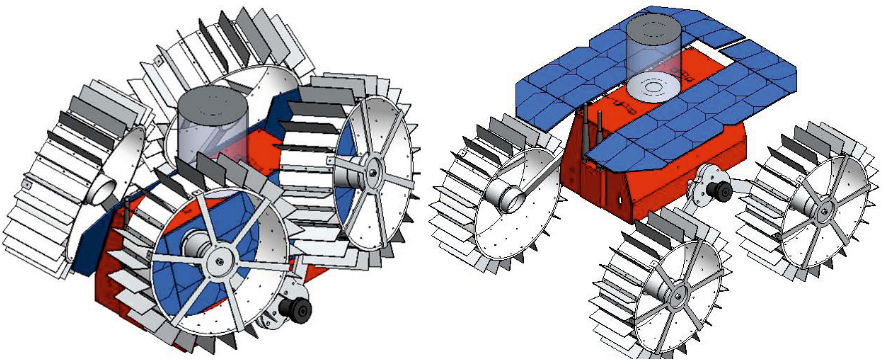


Fig. 3 Moonraker, shown with 25mm grousers, stowed (left) vs deployed (right)

2.1 Sensors

2.1.1 Mirror Omni-Camera

In order to meet the requirements of the case study mission, Moonraker must be equipped with the sensors necessary to construct a full 360° panoramic image, and view itself. A standard camera equipped on a pan/tilt mechanism would be capable of meeting these requirements, but the additional mass necessary for such a mechanism is a challenge to accommodate within a 1kg total sensor mass budget (Table 2). The potential for motor failure also introduces risk that redundancy may be necessary to mitigate. In order to circumvent these challenges, wide FOV camera options were evaluated and a hyperbolic mirror omnicamera was selected as the primary sensor. Using a static camera pointed vertically towards a hyperbolic mirror, the omnicamera system is able to construct 360°panoramic images in single frames.

Using a calibration model developed by Scaramuzza[6], each pixel on the image plane (bottom of Fig.4) can be projected onto a sphere to represent a ray in 3D space through the focus of the hyperbola, as shown in Fig.6. This model is useful for displaying and evaluating the characteristics of the environment, as the spatial relationship between pixels is known. This allows for undistorted views as seen in Fig.5. With a 70mm diameter mirror and 10 megapixel camera, the average visual acuity is estimated to approach 0.3 mrad per pixel.

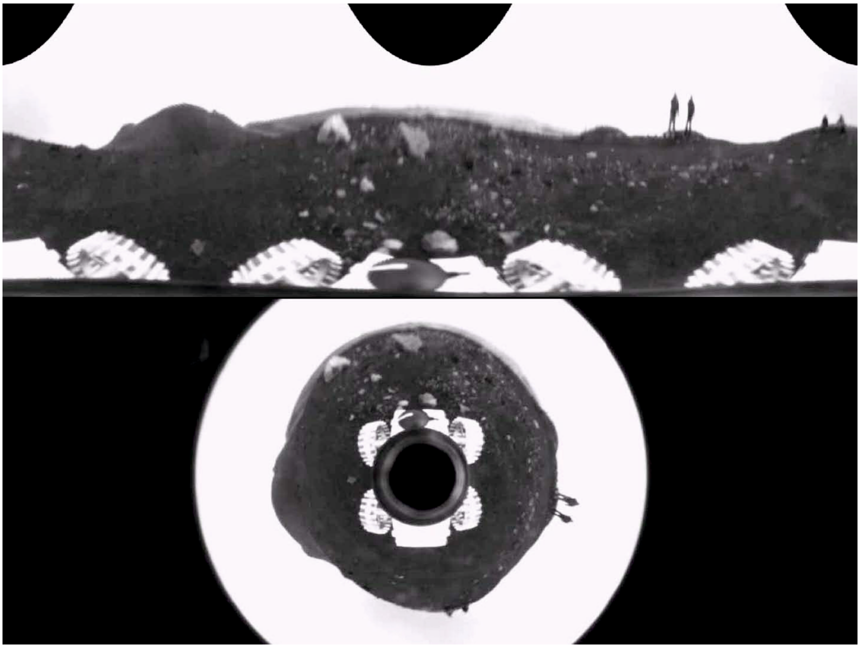
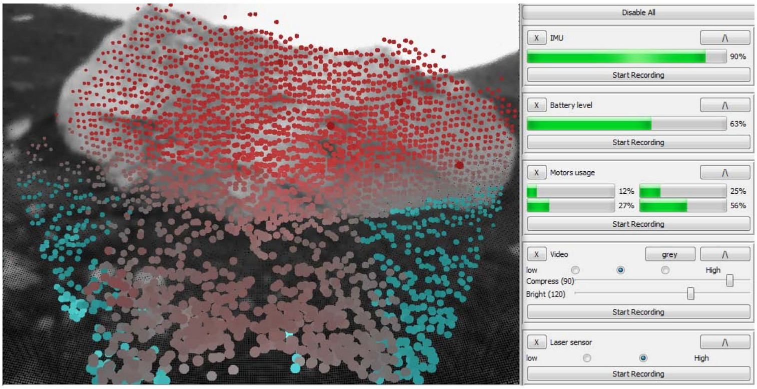


Fig. 4 Omnicamera raw image (bottom) and corresponding panoramic projection (top)

2.1.2 MEMS Laser Range Finder

Although the omnicamera system enables collection of data on the surrounding environment over a wide area, it must be placed on top of the rover to achieve this. This limits visibility in the immediate vicinity in front of the rover, especially where the rover body obstructs the view of the ground. An additional sensor for detecting imminent obstacles and determining their exact position relative to the wheels is therefore highly desirable; a LRF is capable of satisfying these needs.

A MEMS-based LRF is able to achieve a 3D scan over a more narrow field of view than traditional LRF units, but is able to do so without using motors or large moving parts. A small MEMS mirror is precisely controlled to scan an area in front of the sensor, using less power and keeping all moving parts in a small, isolated chamber inside the sensor. The sensor selected for Moonraker (FX8, from Nippon Signal) is able to scan 6000 points over a 60° by 50° region four times per second. The range is specified as 5m, but effective range during field tests rarely exceeded 3m. Although limited for applications requiring full 3D data on the entire surrounding environment, this is sufficient for the requirements of Moonraker.



IMU 90%

Battery level 63%

Motors usage 12% 25% 27% 56%

Video grey

Compress (90)

Bright (120)

Laser sensor

Fig. 5 Technical specialist's graphical interface, with a snapshot of the mission specialist's view of fused sensor data (left) and telemetry with controls for various subsystems(right)

2.2 Teleoperation Interface

Two-person teams under a shared-roles model, with a task breakdown between one mission/payload specialist and one technical specialist, have been found to be highly effective for field robotics teleoperation[5]. Moonraker's teleoperation software interface is designed to reduce potential bottlenecks to understanding and acting on available information quickly.

The mission specialist software implements a virtual reality (VR) interface to Moonraker's sensor data. This allows the operator to easily focus on and intuitively change focus between various narrow targets of interest within a set of wide and otherwise cumbersome data. Without this interface, distortion prevents an accurate understanding of the spatial relationships between obstacles, and navigating the large panoramic image or raw circular data becomes the primary operational challenge.

The technical specialist uses a more traditional mouse-based graphical interface in parallel to view telemetry and control subsystem settings. This support interface is rendered in a window on the same field-laptop (Figure 5), and displays the status of each subsystem. The status of the IMU, remaining battery power, torques and power used by the motors, and the signal strength of the connection with the rover, can be seen. The resolution of both the camera and the LRF, as well as the camera color settings, brightness settings, and compression rate can be controlled. A snapshot of what the mission specialist is doing is available to facilitate effective communication between the operators.

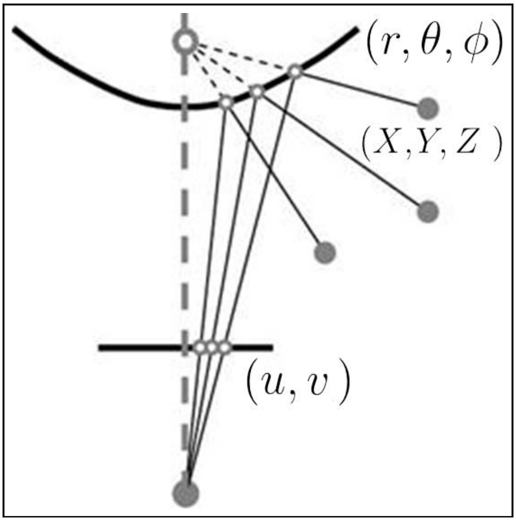


Fig. 6 The pixels on an image plane (u, v) are projected, through the hyperbolic mirror above it, into rays through 3D space (r, θ, ϕ). These rays can be used as unit vectors (X, Y, Z) to compose a virtual semi-sphere for displaying undistorted images as viewed from the focus of the hyperbola, at the center of the sphere. (image source: Scaramuzza[6])

2.2.1 Virtual Reality (Sensor Fusion)

The data from all three sensors (camera, laser, IMU) are fused to provide the VR interface display. The camera system streams compressed video of the panoramic images to the operator computer. These panoramas are undistorted into a spherical projection (Fig.6), and adjusted to match the orientation of the rover from IMU data. Using an open source 3D engine (Ogre3D), the projection is rendered around the position of two cameras.

Frames of pointcloud data from the LRF are synchronized with the camera frames and orientation and are displayed in alignment to give precise distance information about potential obstacles in front of the rover. The HMD used (Oculus Rift) implements stereoscopy, requiring one rendering per eye, which gives the VR operator a natural sense of depth when viewing the pointcloud. The operator can also see any obstacle's position relative to the rover, its wheels and any other feature of interest in view of the camera.

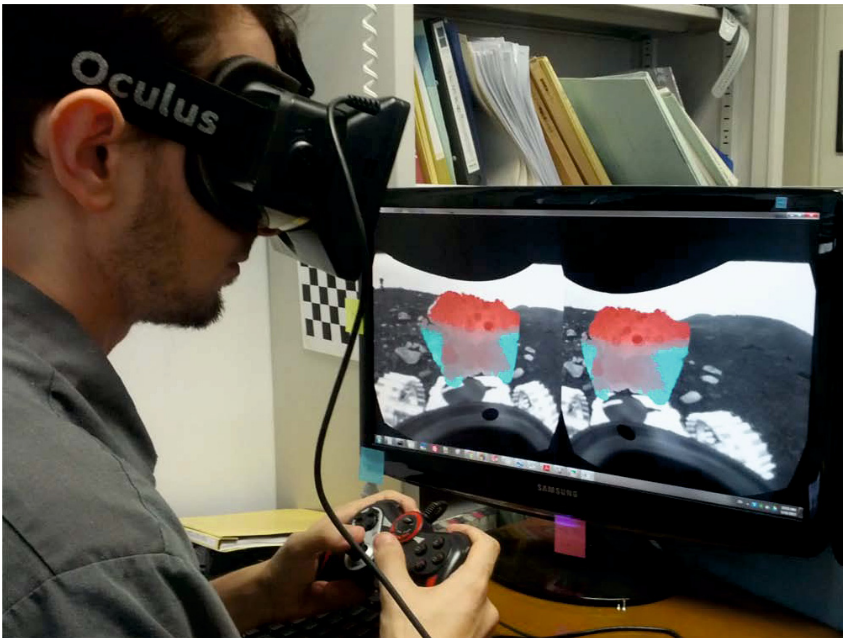


Fig. 7 Mission specialist's graphical interface, with a rendered view of fused camera and LRF data for each eye. A joystick is used to control the mobility system.

2.3 Mobility

2.3.1 Four Wheels

Tracks are known to have advantages over wheels for clearing rough terrain and maximizing contact area with soil, but are also heavier than wheels[9]. Tracks are also less reliable than wheels, due to a susceptibility to failure from jamming, which is difficult or impossible to repair in an exploration mission[7]. Wheels were therefore selected as the mobility system for Moonraker.

More wheels are generally considered to result in higher performance, due to the additional contact points with the surface[10]. A primary advantage of the popular 6-wheel configuration is from rocker-bogie suspension, which facilitates agile traversal over rocky and uneven terrain. This advantage, however, does not apply to the primary mobility challenge of slippage on steep soft regolith slopes of the target lunar mare environment. The two primary wheel parameters that effect slippage on loose soil are *wheel diameter and grouser length*.

The envelope length of 50cm (Sec. 1.4) is the primary constraint to wheel size. With a 10cm gap between wheels, in a 6-wheel configuration the space available for wheel and grousers together is 10cm diameter; in a 4-wheel configuration, the diameter doubles to 20cm. Despite its popularity, for mass/volume-limited missions in loose soil environments, *the 6-wheel rocker-bogie design is a waste of valuable wheel diameter potential*, and results in higher risk for slippage in loose soil.

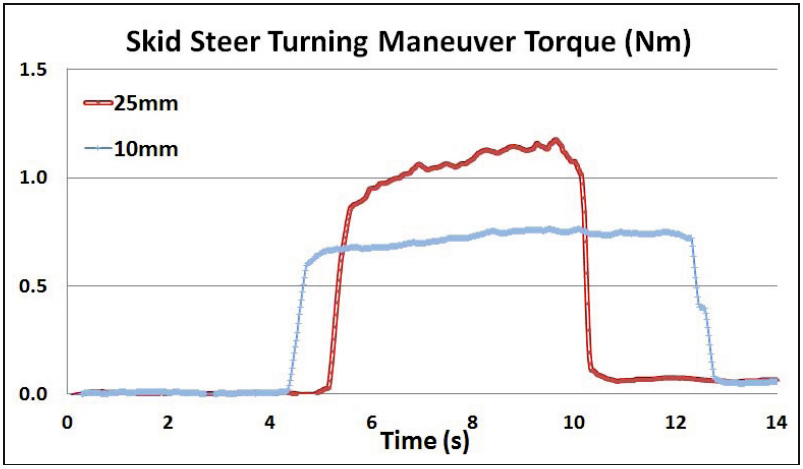


Fig. 8 The average torque per wheel required to perform a 90° spot-turning maneuver. The area under the 10mm curve is 8% larger than the taller 25mm curve.

2.3.2 Grouser Length, Steering and Laboratory Experiment

Increasing the diameter of a wheel and the length of its grousers will improve linear traveling performance over loose soil by decreasing slip. Grouser length has a greater impact on slip than diameter, to the extent that lengthening grousers at the expense of diameter still improves performance[8]. This improved performance directly impacts traversability up steep slopes - a primary requirement for Moonraker. The maximum possible grouser length is therefore desirable for linear travel, however the impact of grouser length on steering is not as well known.

Steering can either be performed precisely by spinning the wheels about the axis perpendicular to the ground with additional motors, or by skid steer, where the difference between the rotational speed of the left and right motors causes rotational slip. Although less precise, skid steering is simpler and more mass efficient. However, the reliance of skid steering on slip and the additional power required to overcome resistance against the soil was expected to reduce traveling efficiency. Experiments were conducted in an indoor laboratory environment to determine the impact of grouser length on skid steering spot-turn maneuvers (turning in place).

Moonraker was equipped with 20cm wheels and comparisons were made between the performance of 10mm grousers and 25mm grousers on operations conducted in a sandbox of Toyura sand, measured with an Osprey motion capture system. The linear slip ratio was calculated to quantify the benefit of the longer grousers for linear performance. The torques of the wheels were also measured during 90° spot-turning maneuvers at a constant speed of 2.9rpms to quantify the difference in skid steering torques. Longer grousers were expected to require more torque, but the extent was not known.

With 10mm grousers, slip began at 8° and by 16° became severe (Fig.9). With 25mm grousers, no measurable slip occurred even at 16°. This dramatic difference in the slip ratio confirms previous research with 2-wheel rovers and, in isolation from steering concerns, result in a preference for 25mm grousers on Moonraker.

Based on the linear performance results, higher requisite torques and expended energy for turning operations was anticipated. As shown in Fig.8, although the torques required for a spot-turning operation with 25mm grousers reach up to 60% higher than with 10mm grousers, the turning operations were in fact completed in almost half the time. *This increases, rather than decreases, the efficiency of spot-turning operations by 8%.*

Our proposed explanation for the unexpected result is that the sand-flow against the wheel during a spot-turn rotational slip approaches a direction parallel to the grousers. The grousers' push against the sand moves the rover, causing a yaw rotation. Longer grousers create greater resistance against the sand in the direction of wheel rotation, but if the rover is not moving forward during the turn, the grousers do not create greater resistance in the direction of the rover's yaw. This allows longer grousers to maneuver the rover more quickly while sand flows between grousers with little resistance. Further testing is required to prove this hypothesis.

2.3.3 Suspension and Wheel Deployment

Using a rocker suspension system ensures that each of the wheels remain in contact with an uneven surface, and when traversing over small rocks. With a vertical envelope restriction of 30cm, the clearance between the rover body and the ground is restricted, which limits traversability over obstacles. In the initial iteration of the Moonraker design, almost 50% of the volume in the 50x40x30cm envelope was empty. It was found that stowing wheels above the rover body, for deployment post-landing, dramatically increased the utilization of the limited envelope.

Placing the stowed wheels diagonally to fill the corners of the volume gives the largest improvement in space available to the wheels. This allowed for an increase in wheel-grouser diameter from 20cm to 25cm, the clearance over the ground from 9cm to 15cm, the wheel width from 4cm to 8cm, and the lateral distance between left and right-side wheels of the rover from 40cm to 50cm. These improvements, which increase stability and traveling performance on slopes, are otherwise not possible with the strict volume limitations.

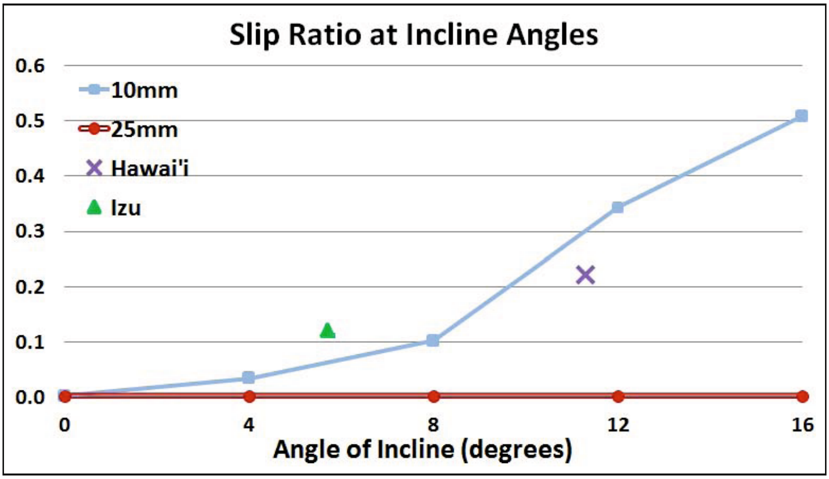


Fig. 9 Slip ratios from lab tests of 10mm and 25mm grousers, compared with field tests(Sec.3)

3 Field Tests

Field tests of Moonraker were conducted on Mt. Aso in Kyushu, Japan (2011), Mt. Mihara at Izu Oshima, Japan (2012), and Mauna Kea, Hawai'i (2013). Each of the test sites provided a range of environments from very soft, loose volcanic ash to rocky pumice, on and near slopes of mountains and volcanic craters. This type of environment is the closest available, on Earth, to what can be expected on the lunar surface. In each test Moonraker was equipped with 10mm grousers and successfully traveled over 500m towards points of interest. Issues in mobility and teleoperation efficiency were identified in each case, informing iterative improvements.

In Kyushu, scattered pumice over loose sand and ash was found at Sunasenri near the Mt. Aso crater. A 600m course was successfully run across the southern edge of the crater. With 6W motors, Moonraker was unable to traverse over small rock obstacles when the incline reached 10° and failed to reach the top of the crater. Motors were subsequently upgraded to 12W for a 5x increase in torque (up to 7Nm/motor).

Izu Oshima provides an environment covered in pumice from coarse sand to rocks. A 900m course up to the peak of Mt. Kushigata from the south and down the north was completed, with an average ascent incline of 5.7°, and a maximum of 15°. The average slip ratio over the course of the ascent was 0.12, as shown in Fig.9, which is slightly higher than the prediction of laboratory tests for soft sand.

In Hawai'i, a 600m course was successfully completed over a sandy area in Halewahine Valley on the southern slope of Mauna Kea. A 20° slope of soft dark volcanic ash, with similar mechanical properties to lunar regolith simulant, was found on the north side of the valley. Due a slip ratio approaching 1, direct traversal up the slope was not possible, but by progressing in alternating lateral trajectories, roughly 45° from direct ascent, it was possible to climb 30m (Fig.10). Over a 100m path, at an average incline of 11°, the slip ratio was 0.22 (Fig.9).

This field testing helped to develop the interface, which evolved from simple command line to video and gamepad control to the interface described in Sec. 2.2. Difficulty in control with communication delay have informed the next step in interface development. The next techniques to be implemented in field testing are mapping, semi-autonomy and global coordinate commands.

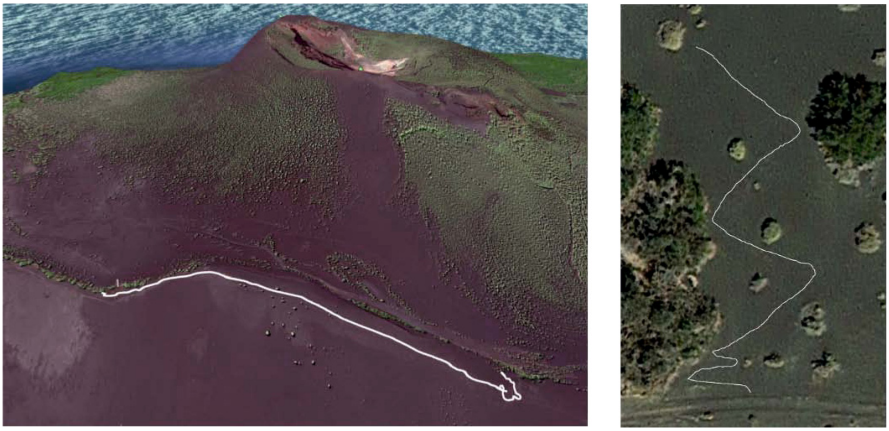


Fig. 10 900m path of a field test over Mt. Kushigata, near Mt. Mihara on Izu Oshima Island, Japan (left). 100m path up a 20°slope of soft volcanic ash performed on Mauna Kea, Hawai'i (right).

4 Conclusion

Micro rovers in the 10kg-class are an attractive option for performing an exploration mission at low cost. Although the reduction in size can limit capability, with careful design, a micro rover can still provide rich data on the explored environment while still being able to provide extensive mobility over across varied terrain.

A micro rover testbed, code-named Moonraker, was designed based on requirements for a specific lunar mission to investigate a lunar skylight. Static sensors which require no moving parts, such as an omnicamera and a MEMS laser range finder allow for a lightweight but effective solution to extensively image the environment and collect sufficient data for full situational awareness. The wide coverage area of the sensors make a virtual reality interface an effective part of a shared-roles teleoperation model, where one operator can intuitively change focus between various narrow targets of interest within a wide set of available data.

A 4-wheel skid steering mobility system demonstrates the potential for high traveling performance on loose soil for micro rovers, if wheel size is not sacrificed to keep the system small. In limited-volume conditions, wheel size can be maximized by utilizing the corner space of the envelope above the rover body. Laboratory experiments were conducted to validate the efficiency of the 4-wheel skid steering maneuvers and found that longer grousers make spot-turn maneuvers more efficient and dramatically improve traversability on inclines of loose soil.

Field tests conducted with Moonraker validate the design by demonstrating successful completion of the case study mission goals. Slip ratio results from the field were found to deviate only marginally from laboratory tests. These real world field tests helped to highlight shortcomings in mobility and software interface, leading to opportunities for effective iterative improvements.

References

1. Fritz, I.C.: Repose behavior of lunar simulants. NASA SEED Program Report, Carthage College, Department of Physics and Astronomy (2010)
2. Haruyama, J., Hioki, K., Shirao, M.: Possible lunar lava tube skylight observed by selene cameras. Geophysical Research Letters 36(21), 1–5 (2009)
3. Lunar Reconnaissance Orbiter Camera. LOLA RDR Data Products Between $44.957^{d}eg$ N, $25.616^{d}eg$ E and $44.963^{d}eg$ N, and $25.623^{d}eg$ E. Washington University, PDS Geosciences Node (2011)
4. Lunar Reconnaissance Orbiter Camera. LROC Observation M126759036L. Arizona State University (2010)
5. Murphy, R.R., Burke, J.L.: From remote tool to shared roles: Human-robot interaction in teleoperation for remote presence applications. IEEE Robotics & Automation Magazine 15(4), 39–49 (2008)
6. Scaramuzza, D. Omnidirectional Vision: From Calibration to Robot Motion Estimation. PhD thesis, ETH ZURICH (2008)
7. Seeni, A., Schäfer, B., Hirzinger, G.: Robot mobility systems for planetary surface exploration - state-of-the-art and future outlook: A literature survey. In: Arif, T.T. (ed.) Aerospace Technologies Advancements. ch. 10, pp. 189–209. InTech (2010) ISBN: 978-953-7619-96-1
8. Sutoh, M.: Traveling Performance Analysis of Lunar/Planetary Robots on Loose Soil. PhD thesis, Tohoku University (2013)
9. Wakabayashi, S., Sato, H., Nishida, S.: Design and mobility evaluation of tracked lunar vehicle. Journal of Terramechanics 46(3), 105–114 (2009)
10. Zhang, P., Deng, Z., Hu, M., Gao, H.: Mobility performance analysis of lunar rover based on terramechanics. In: IEEE/ASME International Conference on Advanced Intelligent Mechatronics. IEEE (2008); Available via IEEE Xplore

Part VI
Mapping and Recognition

Localization and Place Recognition Using an Ultra-Wide Band (UWB) Radar

Eijiro Takeuchi, Alberto Elfes, and Jonathan Roberts

Abstract. This paper presents an approach to mobile robot localization, place recognition and loop closure using a monostatic ultra-wide band (UWB) radar system. The UWB radar is a time-of-flight based range measurement sensor that transmits short pulses and receives reflected waves from objects in the environment. The main idea of the poposed localization method is to treat the received waveform as a signature of place. The resulting echo waveform is very complex and highly depends on the position of the sensor with respect to surrounding objects. On the other hand, the sensor receives similar waveforms from the same positions. Moreover, the directional characteristics of dipole antenna is almost omnidirectional. Therefore, we can localize the sensor position to find similar waveform from waveform database. This paper proposes a place recognition method based on waveform matching, presents a number of experiments that illustrate the high positon estimation accuracy of our UWB radar-based localization system, and shows the resulting loop detection performance in a typical indoor office environment and a forest.

1 Introduction

An ultra-wide band (UWB) radar is a time-of-flight based range measurement sensor that transmits short pulses and receives reflected waves from objects in the environment. The resulting echo waveform is the sum of the reflected waves, and the delay of each component reflected wave depends on the distance between the sensor and the object. Consequently, the received waveform is a superposition of individual

Eijiro Takeuchi
International Research Institute of Disaster Science, Tohoku University
e-mail: takeuchi@rm.is.tohoku.ac.jp

Alberto Elfes · Jonathan Roberts
Autonomous Systems Research Program, CSIRO Computational Informatics
e-mail: {Alberto.Elfes,Jonathan.Roberts}@csiro.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_19, © Springer International Publishing Switzerland 2015

echoes that depend on ranging distance, material, shape and radar reflectance of the obstacles in the environment.

UWB radar has a number of potential advantages over other sensors, particularly optical devices such as stereo systems and laser scanners, that are extensively used on mobile robots. The echo waveform measured by an UWB radar contains information than that of an optical sensor of comparable range, as it provides data from inside and beyond obstacles. UWB radar system is able to see through and beyond obstacles, walls and vegetation, and can detect objects through smoke and fog. Hence, they can be used for reliable localization under a broad range of conditions, which is especially promising for field, search and rescue, and service robotics.

Applications of UWB radars have included imaging the environment [1, 2], detecting and tracking people through walls [3], and localization. Several localization methods using UWB radar have been proposed [4, 5, 6, 7, 8]. However, almost all of these methods rely on augmenting the environment with artificial beacons.

This paper proposes a place recognition method using a UWB radar based on waveform matching. The full radar echo signal is used as a signature for a given location. These signatures are collected and stored in a database of places, and are subsequently used for place recognition and robot localization when the same location is revisited by the robot. The proposed method results in accurate estimates of location in real environments without the use of any artificial beacons. The paper also presents a number of experiments that illustrate the high position estimation accuracy obtained by our UWB radar-based localization system, and we show the resulting localization and loop detection performance in a typical indoor office environment and a forest.

1.1 Related Work

One of the major applications of UWB radars is penetration imaging of objects using synthetic aperture radar (SAR) techniques [1, 2]. These techniques are used for ground and underground imaging from satellites, subsurface imaging from airborne platforms, and imaging of buildings from ground-based systems. The environment model is estimated using reflected waves measured from different positions. Other applications include through-wall moving object detection, which can be used for tracking people [3]. Since the UWB radar receives the sum of reflected waves from various objects, moving objects can be detected from temporal changes in the detected waveform.

Segura *et al.* implemented a functional SLAM and localization system using UWB radar sensor [4]. The receiver on a mobile robot measured waves from multiple fixed transmitting antennas and estimated the transmit antenna positions and the robot position using an extended Kalman filter (EKF).

Blanco *et al.* proposed a range-only SLAM-based method using a Rao-Blackwellized particle filter and a sum of Gaussians approximation [5]. The paper reported SLAM experiments using an UWB transceiver and radio beacons.

Several other UWB radar-based localization methods have been proposed for mobile robots, including [6, 7] and [8], who demonstrated localization using active nodes or base stations.

1.2 Waveform Matching for Localization

In contrast to previous research, this paper proposes a reflected radar waveform matching method for localization and loop closing. The method estimates positions in real environments without the need of artificially introduced beacons, reflectors or any other special devices.

Data retrieval based place recognition methods have been actively studied in the computer vision community [9, 10]. These methods recognize a location by comparing it to similar images retrieved from a database. Similarly, we use the measured reflected waveform as a feature vector that encodes the signature of a place, and localize a mobile robot by comparing this signature to other signatures (measured waveforms) retrieved from a wave signature database.

The remainder of this paper discusses the waveform matching based place recognition method and shows that it leads to a very high precision, high recall rate, and accurate localization estimation. We also show how wave-based database building and localization are performed in indoor environments and a forest without beacons.

2 UWB Radar

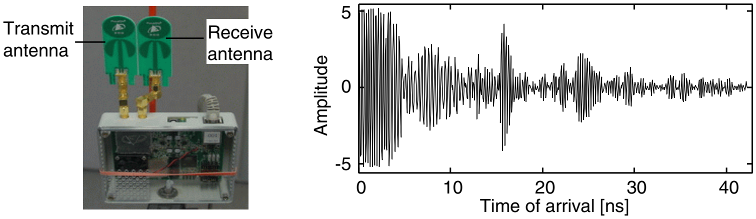
The UWB radar used in this work and shown in Fig. 1(a) is a Time Domain Inc. P400 system [11] with a range of approximately 30 m. The system is small, low power, and has no moving parts. The radar unit has two antennas, the transmit antenna and the receive antenna. The antennas are approximately omnidirectional in the azimuth direction [12], leading to a doughnut-shaped beam pattern.

The radar transmits a short pulse and receives reflected waves from various objects in the environment. The resulting echo waveform is the sum of the reflected waves, and the delay of each component reflected wave depends on the distance between the sensor and the object. The received waveform is a superposition of individual echoes that depend on ranging distance, material, shape and radar reflectance of the obstacles in the environment. Since the radio waves can penetrate objects, the received waveform will depend on a large 3D volume of the environment around the radar.

2.1 Reflected Waveform for the UWB Radar

Figure 1(b) shows a typical received echo for the TD P400 UWB radar unit. As can be seen, the return signal shows a very complex waveform.

The echo or received waveform is the sum of the waves reflected by various objects. This is expressed by the following simplified wave function [2]:



(a) UWB Radar P400

(b) Received wave

Fig. 1 The UWB radar used in this work and a measured reflected waveform example

$$f(t) = \sum_{i=0}^{N-1} (a_i \delta(t - \tau_i))$$

In this equation, $\delta(t)$ is the transmitted wave, a_i is the strength or amplitude of the reflection from a single object i, τ_i is the time of arrival, and N is the number of objects, which will depend on the distance between the radar and the objects in the environment, as well as the objects shape and reflectance characteristics.

3 Waveform Matching Method for UWB Radar

In this section, a correlation based matching and retrieval method for UWB radars is described. For a given measured echo, the proposed method searches for similar waves in a place signature (measured echo waveforms) database.

3.1 Wave Correlation

The echo signature database $\boldsymbol{M}$ contains the reflected waveform $\boldsymbol{f}_i$ measured at each position $\boldsymbol{x}_i$:

$$\boldsymbol{M} = (\boldsymbol{f}_i, \boldsymbol{x}_i) \quad (i = 0, ..., N-1)$$

where

$$\boldsymbol{f}_i = (f_{i0}, \cdots, f_{it}, \cdots, f_{iT-1}), \quad \boldsymbol{x}_i = (x_i, y_i)$$

In these equations, t is discretized time and N is the id of the signature waveform (and of the associated location) in the database.

In this paper, the similarity between waveforms i and j is defined as:

$$E_{ij} = \eta \sum_{t=0}^{T-1} (w_t f_{it} - w_t f_{jt})^2$$

where w_t is a weighing coefficient and η is a normalization factor defined as:

$$\eta = 1/\min(\sum_{t=0}^{T-1} (w_t f_{it})^2, \sum_{t=0}^{T-1} (w_t f_{jt})^2)$$

If the radar echoes are received at the same position, they will be similar and consequently E_{ij} will be low. Conversely, if E_{ij} is large, the signature echoes will likely have been measured at different locations. The normalization factor is used to keep the values of E_{ij} commensurate.

3.2 Finding the Peak of the Correlation

The proposed method finds corresponding locations using the similarity function as follows:

1. measure the new echo waveform $\boldsymbol{f}_j$
2. calculate similarities E_{ij} with all possible signatures i in the signature database $\boldsymbol{M}$
3. calculate minimum value E_{min} of the similarities E_{ij}
4. find additional signature candidates by selecting those k such that $|E_{kj} - E_{min}| \leq \varepsilon$, where ε is a predefined tolerance
5. check the position variance of candidates $Var(\boldsymbol{x}_k)$

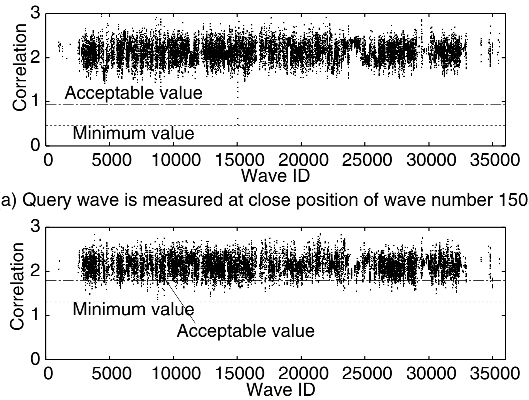
In the results shown later in the paper, the acceptable value of ε is 0.5 and the acceptable variance of position is 0.03 m.

Figure 2 shows search examples where the radar signature from a specific position is correlated against the signatures in the database. For this dataset, the database contains 36858 signatures. Figure 2(a) shows the correlations for an echo measured very close to position number 15054, which is the matched location. There were two additional candidates within the defined tolerance, which are locationally very close. Figure 2(b) shows the results for a location that had not been previously visited, where no acceptable match was found.

3.3 Sensitivity Weight

Due to power dispersion, echoes received from obstacles become weaker with increasing distance to the obstacle. However, the more distant components of place signatures change more slowly, and are therefore more robust to small changes in receiver position. To compensate the diminishing intensity of the echoes from more distant targets, we use the sensitivity weights w_t.

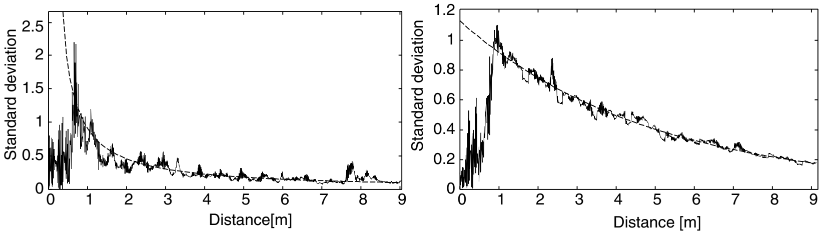
In this research, the sensitivity weight is designed to flatten the effect from unknown obstacle as a function of distance. To define the sensitivity weight, consider that we have a human moving within the range of the UWB radar. Figure 3(a) shows the standard deviation of the echoes for a moving person. In these experiments, the human moves in front of radar with constant velocity. The plot is the standard deviation of the echoes, while the dashed black line is a fit using an inversely proportional



(a) Query wave is measured at close position of wave number 15054

(b) Query wave is measured at outside of database positions

Fig. 2 Correlation of a signature against the wave signature database



(a) Standard deviation of received echo for a moving human (b) Standard deviation of echoes measured at various positions

Fig. 3 Standard deviation of received echoes for a moving human

function. In this figure, the standard deviation shows only effects of moving objects because reflected waves from static objects are almost constant value. The figure illustrates that the standard deviation decreases with distance and the relationship is almost inversely proportional. At the near distances, the standard deviation is decreased, and this was caused by saturation.

This is compensated using the following proportional function as the sensitivity weight:

$$w_t = kt$$

For the later results, the parameter k was set to 0.01.

3.4 Range and Sensitivity

To accurately find similar signatures in the database, it is necessary to use the portions of the signature that correspond to more distant objects as they provide more robust information. At the same time, the signature search and matching time depends on the effective range of the signature that was used. This section discusses this effective range.

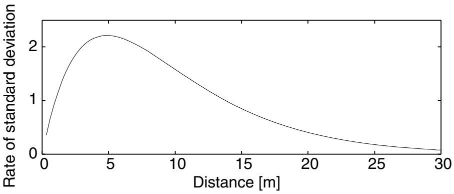


Fig. 4 Products of the sensitivity weight and information intensity of echoes

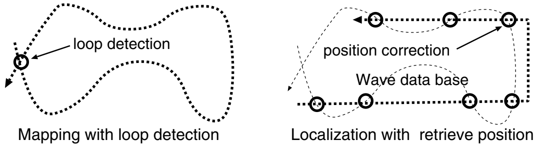

loop detection
Mapping with loop detection
position correction
Wave data base
Localization with retrieve position

Fig. 5 Loop detection and localization using radar signature matching

Similarly to what was shown in Fig. 3(b) shows the standard deviation of signatures measured at various position in a static indoor environment, while the dashed line is an exponential function fitted using a least square method. The standard deviation indicates breadth of the value change over the environments, and the high deviation ranges are more effective ranges to identify the waves. Again, this shows that the intensity of the information decreases exponentially with distance.

The sensitivity weight is increased propotionally with distance and the environment information decreases exponentially with distance. As a result, we can find the sensitive distance of the echoes by multiply these characteristics. Figure 4 shows the sensitive distance. As can be seems, the most useful information is in the middle range of the signatures. This research uses a range out to 9 m for signature matching.

4 Experiments

This section describes several experimental localization results using the proposed signature matching method. Figure 5 illustrates localisation and loop closure process. In the mapping phase, the robot moves around an environment using a meandering motion. This motion increases the number of potential crossing points and consequently of signature candidates for localization and loop closure. Once the robot finds a matching signature in the database, it closes the trajectory using a graph-based SLAM approach [13] [14][15] to keep consistency of the database. In the localization phase, if the robot finds several similar signatures in the database, it corrects the position using a position estimation method such as an EKF [14].

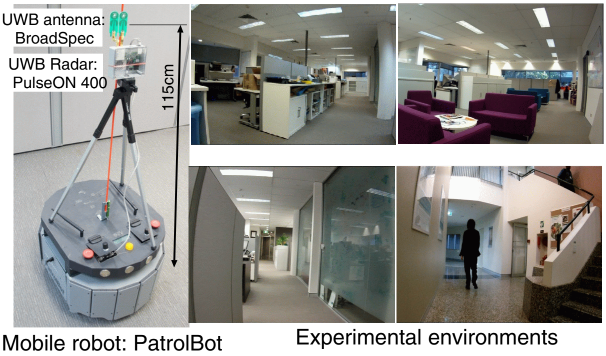


Fig. 6 Experimental vehicle and indoor environments used for testing at the Autonomous Systems Lab, Brisbane

4.1 Experimental Setup

Figure 6 shows the experimental setup and the experimental environment. The mobile robot was a PatrolBot developed by Mobile Robots Inc., and the UWB radar was a PulsOn 400 MRM developed by Time domain Inc. The radar was mounted on the robot at a height of 1.15 m. The experimental environment was a typical indoor office area with meeting rooms, desks and people. The UWB radar has a range of up to 30 m, of which the first 9 m were used as discussed previously. The sampling rate at which the signatures were collected was 25 Hz. The mobile robot moved at speed under 10 cm/s, and consequently the inter-measurement distances were under 4 mm.

4.2 Waveform Retrieval with Unknown Objects and a Moving Target

The first experiment was a robustness test with unknown objects. The robot repeated a move and stop motion in the environment, and a human moved around the robot randomly when the robot stopped. The distance of the person from the robot was between 1 m and 10 m. Figure 7 shows the similarity and the obtained results. In this figure, the upper triangle shows the similarity values, while the bottom triangle is the matched results. The white points on the upper triangle side indicate high correlation. On the lower triangle the gray points show true positive matches, the white points are false negatives, and the black points are true negatives.The false positives are almost nothing. The true positives mean that the distance between the retrieved signature wave position and the query wave position was under 1 cm. The gray squares show places where the robot stopped for varying amounts of time.

In this experiment, the precision of retrieval was 0.999 and the recall rate was 0.955, showing that the method was robust to the disturbance of having a person moving around the vehicle in an indoor environment.

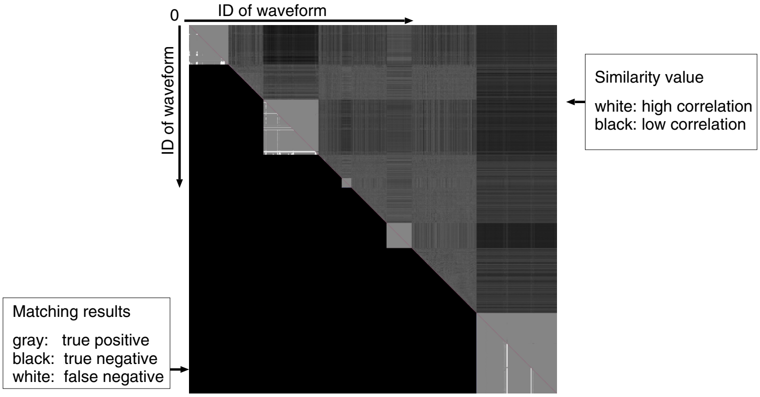


Fig. 7 Waveform retrieval results using radar signature matching

4.3 Loop Detection and Closing

The following experiments demonstrate how the technique can be used for loop detection and closing to build maps. Here, the robot moved around the laboratory using a meandering route. Figure 8 shows the loop detection results using the proposed method. The solid line shows the route taken by the robot, and the dashed lines are corresponding points detected by the signature matching method. The route has 23 crossing points and the proposed method detects loops on 22 crossing points. The corresponding pairs are found irrespective of the direction that the vehicle is pointing at, as the radar antennas have omnidirectional sensitivity in the azimuth direction. For this experiment, these were no corresponding pairs with large distance errors. However, there were cases where multiple corresponding pairs were found clustered close to the actual position, leading to small errors in the estimated position. The total number of signatures in the database is 36858, and the mean matching/retrieval time is 25ms on an Intel core i7 2.8 GHz processor. The graph had 1263 nodes and 1460 constraints, and the graph closing time is approximately 0.4 s.

Figure 9 shows loop closing results using a graph-based SLAM method [15]. The solid line is the corrected graph, the black points are laser scanner mapping results using the corrected positions, circles are detected loops and the square is a non-detected loop. Note that the laser scanner was used only for reference, but not for loop detection and localization. The results show that the trajectory was corrected and the laser point clouds overlap well.

4.4 Localization Using the Radar Echo Signature Database

The next experiment was conducted to demonstrate the localization performance achieved using our radar echo signature matching and retrieval method. This experiment estimated the position of the robot using an Extended Kalman Filter [14]. The

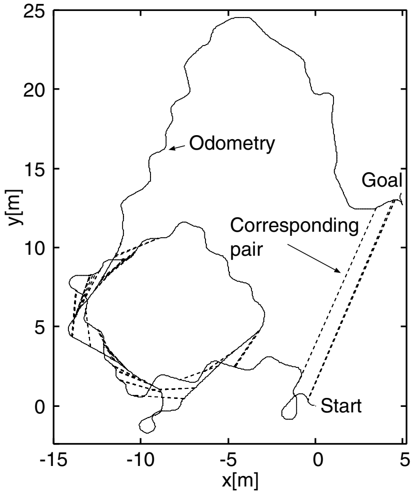


Fig. 8 Loop detection results

motion model used vehicle odometry and the measurement model is the retrieved position match using UWB radar signatures. The localization method uses as a map the radar signature database obtained from previous loop closing experiments, and finds the position using radar signature matching and retrieval.

Figure 10(a) shows the localization results obtained using signature matching. The experiment was conducted three days after the mapping experiment discussed earlier. The solid line is the estimated position using an EKF with UWB radar data, the gray line is the position associate with the signature stored in the database, the crosses are corrected points, and the dashed line is the reference position estimated using a scan matching method with laser scanner data [16]. The laser-based scan matching method estimates the vehicle position using the point clouds that were obtained in the previous experiment, and as a result the ”reference” position has its own, relatively small errors. This figure illustrates that in almost all cases the estimated positions using the UWB radar are close to the reference positions. For this experiment, the total number of signatures in the database is 36858, the search area is 2 m, and the mean matching and retrieval time is 9 ms.

Figure 10(b) shows the difference between the estimated position and the reference position. The black line shows the difference in the x-axis and the gray line the difference in the y-axis, indicating that almost all position differences were smaller than 0.2 m. The results show that the localization accuracy achieved using the UWB radar signature retrieval method is close to the laser scanner localization performance.

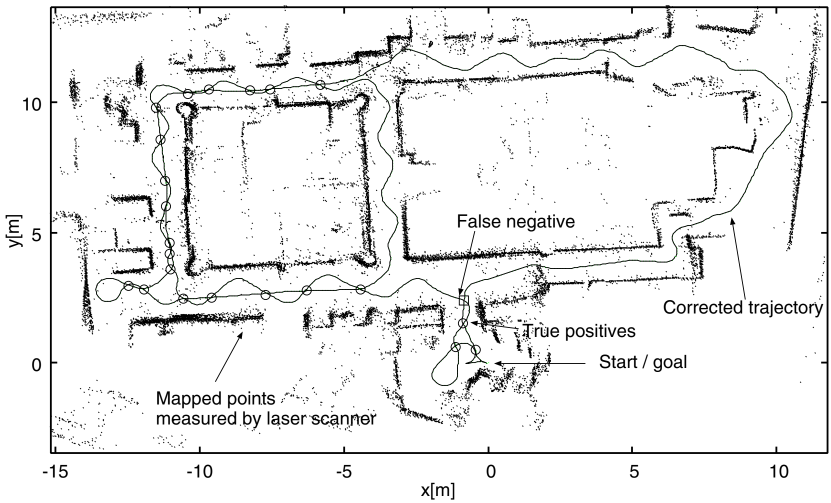
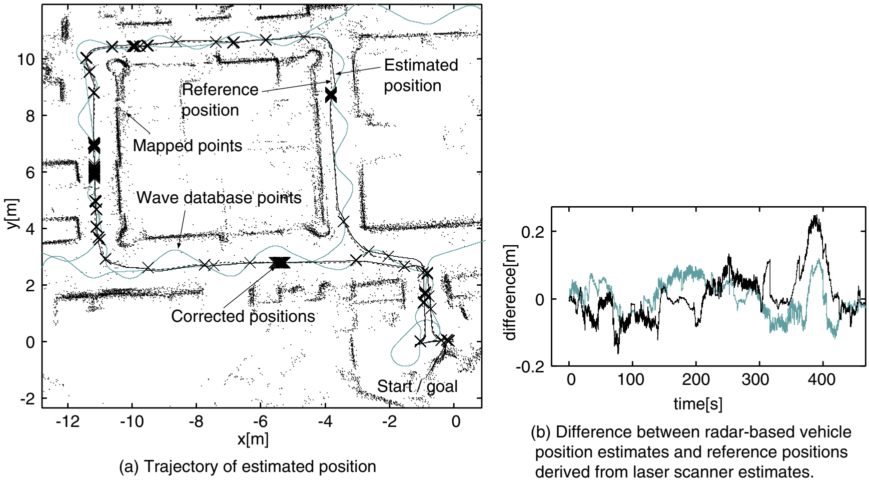


Fig. 9 Loop closing results. The laser scanner was used for mapping, but not for localization.



(a) Trajectory of estimated position

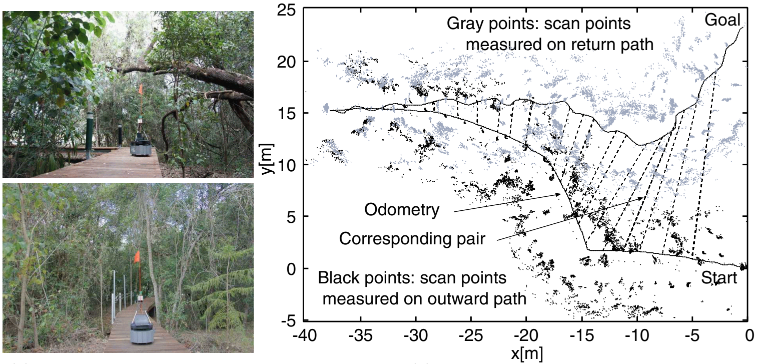
(b) Difference between radar-based vehicle position estimates and reference positions derived from laser scanner estimates.

Fig. 10 Localization results. The laser scanner was used for mapping, but not for localization.

4.5 Loop Detection and Closing in Forest

The final experiment was conducted to demonstrate that the proposed method works in heavy vegetation and forest environments.

Figure 11(a) shows the forest area used for the experiments, which is located within the area of the Queensland Centre for Advanced Technologies (QCAT). The PatrolBot used for the experiments is not an outdoor vehicle, but we took advantage



(a) Experimental environments (b) Loop detection results

Fig. 11 Forest area used for outdoor experiments and loop detection results

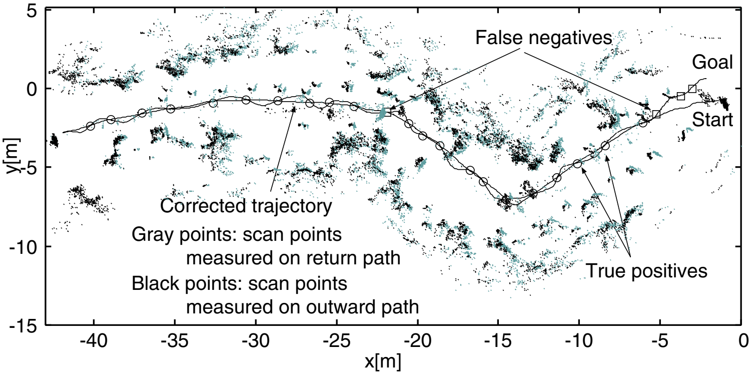


Fig. 12 Loop closing results in forest

of a boardwalk through the forest which provided a route that was traversable by the vehicle. The route is surrounded by trees, heavy vegetation, and some light posts.

In this experiment, a straight trajectory following the boardwalk was used on the outward path, while a meandering trajectory was used on the return path. The parameters used in our method were the same as in the indoor experiments.

Figure 11(b) shows the loop detection results, and Fig. 12 the loop closing results.

The solid line shows the route taken by the robot, the dashed lines are corresponding points detected by the signature matching method, the black points are laser scanner mapped results for reference using the estimated positions on the outward path, and the gray points are mapped points on the return path. In Fig. 12, the circles are detected loops and the squares are non-detected loops. The route has 25 crossing points and our method detects loops on 21 crossing points. The graph had 1090 nodes and 1235 constraints. After loop closing, Fig. 12 shows that the outward and inbound point clouds overlap well. The results show the proposed method can detect crossing points in natural environments.

5 Conclusions

This paper presents an approach to robot loop detection and localization using an UWB radar and an echo waveform signature matching and retrieval method. We have shown that the radar signature waveform is highly correlated with the position of the sensor, that the correlation of waves have strong minima, and that the global minimum can be found using a computationally efficient method.

The experiments illustrate the robustness of the proposed method in the presence of a moving human, and show loop detection and closing, as well as localization results. The robustness experiments show that the method retrieves the correct signature with a high level of precision and recall rate, even with a moving target in the scene. The loop detection results show that the method detects almost all crossing points on a route, and that an accurate map is obtained from the loop closing result using graph-based SLAM. Finally, the localization results using radar are compared with laser scan-matching localization, showing that the differences between UWB radar-based localization and laser scanner-based localization are typically below 0.2 m. Accurate results were obtained for both indoor and outdoor environments.

In summary, the experimental results confirm the usefulness of an UWB radar for place recognition.

Future work will include a more accurate evaluation of the proposed method, the extension of the technique to more complex indoor and outdoor environments, and the development of higher position localization accuracy using predictive modelling of the radar echo signatures. The proposed localization method may have the ability to localize within a millimeter-order accuracy, but the experiments outlined in the paper are not precise enough to assess this potential accuracy.

The UWB radar can detect objects through walls, smoke, fog, and vegetation. Consequently, the sensor and the methods discussed are applicable to domains such as forest traversal or search and rescue in smoky environments. However, the proposed method has only been used to detect crossing points along a 2D trajectory, and while it is also applicable to 2 1/2 D environments, it will need further research to be generalized to 3D trajectories.

Acknowledgements. The authors acknowledge the support of Dr. Pavan Sikka and other members of the Autonomous Systems Research Program at CSIRO Computational Informatics.

References

1. Aftanas, M.: Through-Wall Imaging with UWB Radar System. Lambert Academic Publishing (2010)
2. Judson Braga, A., Gentile, C.: An Ultra-Wideband Radar System for Through-the-Wall Imaging Using a Mobile Robot. In: Proceedings of the 2009 IEEE International Conference on Communications, pp. 4081–4086 (2009)
3. Chang, S., Wolf, M., Burdick, J.W.: Human Detection and Tracking via Ultra-Wideband (UWB) Radar. In: Proc. of ICRA 2010, pp. 452–457 (2010)

4. Segura, M.J., Cheein, F.A.A., Toibero, J.M., Mut, V., Carelli, R.: UltraWide-Band Localization and SLAM: A Comparative Study for Mobile Robot Navigation. In: Sensors 2011, pp. 2035–2055 (2011)
5. Blanco, J.-L., Fernández-Madrigal, J.-A., González, J.: Efficient Probabilistic Range-Only SLAM. In: Proc. of IROS 2008, pp. 1017–1022 (2008)
6. Shen, G., Zetik, R., Yan, H., Jovanoska, S., Malz, E., Thomä, R.S.: UWB Localization of Active Nodes in Realistic Indoor Environments. In: 2011 Loughborough Antennas & Propagation Conference (2011)
7. Mut, V.A., Mut, V.A., Patiño, H.D.: Mobile Robot Self-Localization System using IRUWB Sensor in Indoor Environments. In: Proceedings of IEEE International Workshop on Robotic and Sensors Environments, pp. 29–34 (2009)
8. González, J., Blanco, J.L., Galindo, C., Ortiz-de-Galisteo, A., Fernández-Madrigal, J.A., Moreno, F.A., Martínez, J.L.: Mobile Robot Localization based on Ultra-Wide-Band Ranging: A Particle Filter Approach. Robotics and Autonomous Systems 57(5), 496–507 (2009)
9. Sivic, J., Zisserman, A.: Video Google: A Text Retrieval Approach to Object Matching in Videos. In: Proc. of ICCV 2003, pp. 1470–1477 (2003)
10. Cummins, M., Newman, P.: Accelerated Appearance-Only SLAM. In: Proc. of ICRA 2008, pp. 1828–1833 (2008)
11. Time Domain, "PulsOn 400", `http://www.timedomain.com/p400-mrm.php`
12. Time Domain, "BroadSpec UWB Antenna", `http://www.timedomain.com/datasheets/320-0385A%20Broadspec%20Antenna.pdf`
13. Lu, F., Milios, E.: Globally Consistent Range Scan Alignment for Environment Mapping. Autonomous Robots 4, 333–349 (1997)
14. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press (2005)
15. Takeuchi, E., Tsubouchi, T.: Multi Sensor Map Building based on Sparse Linear Equations Solver. In: Proc. of IROS 2008, pp. 2511–2518 (2008)
16. Biber, P., Straßer, W.: The Normal Distributions Transform: A New Approach to Laser Scan Matching. In: Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2743–2748 (2003)

Laser-Radar Data Fusion with Gaussian Process Implicit Surfaces

Marcos P. Gerardo-Castro, Thierry Peynot, and Fabio Ramos

Abstract. This work considers the problem of building high-fidelity 3D representations of the environment from sensor data acquired by mobile robots. Multi-sensor data fusion allows for more complete and accurate representations, and for more reliable perception, especially when different sensing modalities are used. In this paper, we propose a thorough experimental analysis of the performance of 3D surface reconstruction from laser and mm-wave radar data using Gaussian Process Implicit Surfaces (GPIS), in a realistic field robotics scenario. We first analyse the performance of GPIS using raw laser data alone and raw radar data alone, respectively, with different choices of covariance matrices and different resolutions of the input data. We then evaluate and compare the performance of two different GPIS fusion approaches. The first, state-of-the-art approach directly fuses raw data from laser and radar. The alternative approach proposed in this paper first computes an initial estimate of the surface from each single source of data, and then fuses these two estimates. We show that this method outperforms the state of the art, especially in situations where the sensors react differently to the targets they perceive.

1 Introduction

The ability to build high-fidelity representations of the environment is critical for autonomous robots [14]. Range scanners such as laser range finders and radars are widely used in field robotics [10]. However, like any sensor, they suffer from limitations, e.g. in terms of field of view, resolution and noise [1]. Consequently, robots need techniques that can estimate accurate representations of the environment with incomplete data and uncertainty.

Gaussian Processes (GP) have become a popular technique in recent literature of robotic perception, due to their ability to learn spatial representations from noisy data in a non-parametric Bayesian fashion [12]. Gaussian Process Implicit Surfaces

Marcos P. Gerardo-Castro · Thierry Peynot
Australian Centre for Field Robotics (ACFR),
The University of Sydney, NSW 2006, Australia
e-mail: {m.castro,tpeynot}@acfr.usyd.edu.au

Fabio Ramos
ACFR, School of Information Technologies,
The University of Sydney, NSW 2006, Australia
e-mail: f.ramos@acfr.usyd.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_20, © Springer International Publishing Switzerland 2015

(GPIS) [19] is a mechanism to estimate the surface of an object with uncertainty, within a GP framework, by representing the geometry of the object as an Implicit Surface [15]. GPIS applied on range data offers a number of benefits to overcome the aforementioned problems. Firstly, the generated model is fully predictive as it is able to predict a surface at arbitrary regions of an object that were not entirely observed by the range sensor [5]. Secondly, the model also yields the uncertainty of the estimates, at any point of the surface. By capturing the correlations between points using parametrized covariance functions, only a limited number of points are required to learn an accurate model. In addition, the GP can automatically handle the model selection (parameter estimation) efficiently.

The use of data from multiple sensing modalities can help to obtain a more complete and accurate representation of an object. As these modalities sense the environment using different physical processes, they also respond differently to some materials, textures, or environmental conditions [1, 4].Consider the example of a car with windows perceived by a laser and a radar. Lasers operate at near-infrared frequencies of the electromagnetic spectrum, which are close to visible wavelengths. Therefore, a window appears mostly transparent to their sensing. In contrast, a mm-wave radar operates at lower frequencies (e.g. around $94GHz$), and is getting more returns back from the surface of the windows. On the other hand, on the rest of the car, the laser sensing is more accurate [1]. Therefore, a more complete and more accurate representation of the car can be obtained by fusing the data from the two sensor modalities.

In this paper we perform the fusion of data from two distinct sensing modalities (a laser and a radar) within a GPIS framework, in the context of field robotics. We propose a thorough experimental analysis of the performance of 3D surface reconstruction from laser and radar data in a realistic field robotics scenario: an unmanned ground vehicle (UGV) scanning an outdoor environment. We first analyse the performance of the GPIS approach using raw laser data alone, and using raw radar data alone, with 4 different choices of covariance matrices and different resolutions of the input data, on a total of 8 objects with different geometries. We then evaluate and compare the performance of two different fusion approaches within the GPIS framework. The first approach directly fuses raw data from the two different sensor modalities in the GPIS, as in [2]. In the second approach, we fuse points extracted from two initial estimates of the object surface that were built using raw laser data only and raw radar data only, respectively. We show that this novel approach outperforms the first state-of-the-art approach, especially in cases where the sensing modalities react differently, perceiving different targets. Although we put special emphasis on the implementation of data fusion for laser and radar data in this paper, the frameworks for GPIS fusion may be used with different range sensors.

The paper is organised as follows. Sec. 2 discusses related work on surface reconstruction with uncertainty representations, GPIS, and multi-sensor data fusion. Sec. 3 develops the GPIS framework used in this work and the fusion methods considered. Sec. 4 describes the experimental setup and Sec. 5 presents the experimental evaluation and analysis. Finally, Sec. 6 proposes conclusions and elements of future work.

2 Related Work

The problem of estimating continuous representations of data from range sensors has been extensively studied in the recent literature [13, 8]. To build continuous representations from range sensor data while accounting for uncertainties, different variations of Gaussian Processes have been implemented in the robotics community. Gaussian Beam Processes [11] give an independent treatment of the noise using heteroscedasticity on the beams, however, this approach is limited to 2D scenarios. Other applications using GP in 3D scenarios are related to mapping and terrain modelling [17, 7]. These approaches adopt the same parametrisation problem by associating a single elevation value z with any given position (x, y) in 2D Euclidean space. While this way of mapping is effective for applications such as terrain modelling, it is not suitable for applications that consider the full 3D mapping case, e.g. full 3D modelling of an object, where there can be multiple elevation values for a given (x, y). Implicit Surfaces (IS) is a representation that is appropriate for this case [15]. It takes advantage of the geometry and topology of the objects. GPIS is a framework to estimate IS surfaces with uncertainties using Gaussian Process [19]. This approach has been applied to range sensor data for robotics applications in different contexts, such as change detection [18], active learning [5], and grasping [2].

Gaussian Process has been shown to be a powerful tool for multi-sensor data fusion when considering noisy input data [7, 3]. This concept has been adopted in [6], which proposes a sensor fusion framework based on a mixture of GPs. The application focussed on affect recognition, rather than object representations. GP data fusion was also explored in [17], where the author fuses raw data from laser scanner and Global Positioning System, with distinct noise models for each data source. However, this approach uses a representation that only allows for a single elevation value at a given (x, y) location, which is not appropriate for representing objects, as mentioned above. A similar fusion approach was used in [2] within a GPIS framework. Raw data from lasers and tactile sensors are directly fused in a GPIS with multi-variance noise in the input dimensions. However, the paper shows limited experimental results and no error analysis is provided. In this paper we propose an extensive experimental analysis of the performance of sensor data fusion within a GPIS framework, in the context of field robotics.

3 Gaussian Process Implicit Surfaces

3.1 Implicit Surfaces

Consider a set of points $X = [x_1, x_2, ...x_n]$ in Euclidean space corresponding to observations of the object. In order to model an object represented by X, an Implicit Surface is defined as the 0-level set of real-valued function $f : \mathbb{R}^3 \to \mathbb{R}$, where the function f specifies whether a point x is inside the surface ($f(x) > 0$), outside the surface ($f(x) < 0$), or on the surface ($f(x) = 0$). Such constraints values are assigned to the variable Y, so that $Y = [f(x_1), f(x_2)...f(x_n)]$. Direct observations made by range scanners are usually assumed to be on the surface of the object, therefore, *zero-value constraints* ($f(x) = 0$) are assigned to the sensor measurements. Additional observations of points known to be inside or outside the object may be added to help the estimation process.

3.2 Gaussian Process Implicit Surfaces

Gaussian process regression can be used to provide the surface estimate $f_*(x_*)$, with variance $\mathbb{V}(f_*(x_*))$ based on observation data from X and constraints Y (i.e. targets for the GP), defined as training data under the GPIS framework. This can be formulated as:

$$\mathbb{P}(f_*(x_*) \,|\, X, Y, \theta, x_*) = \mathscr{N}(\bar{f}_*, \mathbb{V}[f_*]), \tag{1}$$

where θ is a set of hyper-parameters. The mean $\bar{f}_*$ and variance $\mathbb{V}[f_*]$ at a selected point x_* given the measured data X are:

$$\bar{f}_* = \qquad k(x_*, X)^T (K + \sigma_n{}^2 I)^{-1} Y \tag{2}$$

$$\mathbb{V}[f_*] = k(x_*, x_*) - k(x_*, X)^T (K + \sigma_n^2 I)^{-1} k(x_*, X), \tag{3}$$

where K is a covariance matrix. The noise variance of the observed data is represented by $\sigma_n^2 I$. Note that σ_n^2 can be learnt along with the other GP hyper-parameters. In this paper, we implement different stationary kernels: the exponential covariance function,

$$k = k(x_i, x_j) = \sigma_f^2 \sum_{k=1}^{d} \exp(-(\frac{\Delta_k}{\ell})^\gamma), \tag{4}$$

and two variants of the Matérn type [12],

$$k_{M_{3/2}} = \sigma_f^2 \sum_{k=1}^{d} (1 + \frac{\sqrt{3}\Delta_k}{\ell}) \exp(\frac{-\sqrt{3}\Delta_k}{\ell}),$$

$$k_{M_{5/2}} = \sigma_f^2 \sum_{k=1}^{d} (1 + \frac{\sqrt{(}5\Delta_k)}{\ell} + \frac{5\Delta_k{}^2}{3\ell^2}) \exp(\frac{-\sqrt{5}\Delta_k}{\ell}).$$

The hyper-parameter σ_f^2 represents the signal variance and the length-scale is represented by ℓ, $\Delta_k = |x_i - x_j|$. $k_{M_{3/2}}$ and $k_{M_{5/2}}$ are stationary covariance functions used to amplify the sensitivity between the correlations of the points compared to the widely used square exponential (Eq. (4) with $\gamma = 2$), which produces a smooth kernel that drops off with distance. We extend our analysis to the exponential covariance function ($\gamma = 1$), which is even more sensitive to changes.

An important aspect of the GP is the optimisation of the hyper-parameters $\theta = (\sigma_f, \ell, \sigma_n)$. In this paper this was done by maximising the log-marginal likelihood. The Cholesky decomposition was used to obtain the predictors ($\bar{f}_*$ and $\mathbb{V}[f_*]$) and the log-marginal likelihood [12]. Once an estimate f_* of the surface has been obtained, 3D surface points and corresponding variances are then computed for values of $f_* = 0$ in Eq. (2) and Eq. (3) by querying in a region pre-defined by x_*.

3.3 Single-Sensing-Modality GPIS

We define a *single-sensing-modality GPIS* as a GPIS whose input is a set of data provided by a single sensor type (in this paper, laser or radar), as illustrated in Fig. 1. We name the process $GPIS_i$, The input data X_i and Y_i, the estimated surface $\bar{f}_{i_*}$, and the variance $\mathbb{V}_{i_*}$, where the index i specifies the nature of the input data: $i = L$ if the input data is provided by a laser, and $i = R$ if the data is from a radar. A global noise parameter ${\sigma_n}^2$ is used for all the input points in each $GPIS_i$ (see Eqs. (2) and (3)).

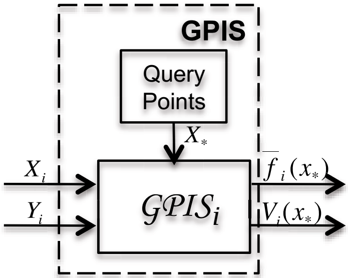


Fig. 1 $GPIS_i$ process, using laser ($i = L$) or radar ($i = R$) input data

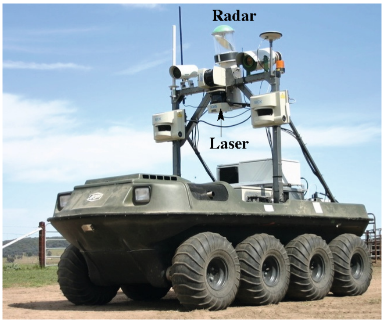
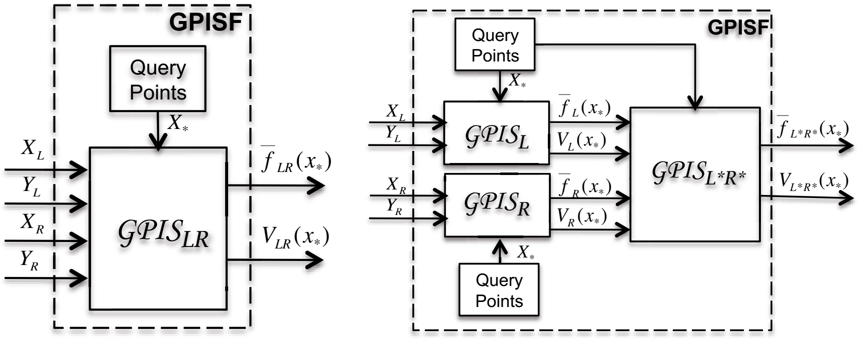


Fig. 2 Argo UGV equipped with the laser and radar sensors used in this study

3.4 Multi-sensor Data Fusion: $GPIS_{LR}$

The first fusion method, $GPIS_{LR}$, fuses two sets of raw data, X_L and X_R, acquired by laser and radar, respectively, in a single GPIS. The approach is similar to the one in [2]. The input data of $GPIS_{LR}$ is composed of all training points from each sensing modalities put together: $X = [X_L \ X_R]$ and $Y = [Y_L \ Y_R]$. A diagram of the process of $GPIS_{LR}$ is illustrated in Fig. 3(a). $GPIS_{LR}$ accounts for different noise parameters for each sensing modality by implementing an input-dependent noise process, i.e. heteroscedastic, similar to [11]. Let $\sigma^2 \in \mathbb{R}^n$ be the noise variances for n given sensing modalities. The predicted distributions become:



(a) $GPIS_{LR}$

(b) $GPIS_{L*R*}$

Fig. 3 The two alternative fusion processes: $GPIS_{LR}$ (a) and $GPIS_{L*R*}$ (b)

$$\bar{f}_* = k_*{}^T (K+H)^{-1} Y \tag{5}$$

$$\mathbb{V}[f_*] = k(x_*, x_*) - k_*{}^T (K+H)^{-1} k_* \tag{6}$$

where $H = diag(\sigma_1{}^2(X_1), \sigma_2{}^2(X_2) ... \sigma_n{}^2(X_n))$ is a non-fixed noise matrix.

3.5 Alternative Fusion Method: $GPIS_{L_* R_*}$

In the alternative fusion approach we propose, $GPIS_{L_* R_*}$ (see Fig. 3(b)), we first estimate the object surface from raw laser points and from raw radar points separately, using two independent GPIS (i.e. $GPIS_L$ and $GPIS_R$). We then query a set of points, X_{L_*} and X_{R_*}, which are randomly sampled from the points where $\bar{f}_{L_*} = 0$ and $\bar{f}_{R_*} = 0$ respectively, along with the corresponding variances, $\mathbb{V}_{L_*}$ and $\mathbb{V}_{R_*}$. The number of points in X_{L_*} and X_{R_*} is two times the original number of input points (i.e. in X_L and X_R). Associated constraints Y_{L_*} and Y_{R_*} are computed from the estimated surfaces $\bar{f}_{L_*}$ and $\bar{f}_{R_*}$. The next step is to compute the final estimate using $GPIS_{L_* R_*}$, with inputs: $X = [X_{L_*}\ X_{R_*}]$ and $Y = [Y_{L_*}\ Y_{R_*}]$. The predicted uncertainties of $\mathbb{V}_{L_*}$ and $\mathbb{V}_{R_*}$ are integrated in the final GPIS as fixed noise parameters. We substitute H in Eqs. (5) and (6), with $H = diag(H_{L_*}, H_{R_*})$, where H_{L_*} is a fixed noise matrix, defined by the variances $\mathbb{V}_{L_*}$ as $H_{L_*} = diag(\mathbb{V}_{L_{*1}}, \mathbb{V}_{L_{*2}} ... \mathbb{V}_{L_{*m}})$, and, similarly, $H_{R_*} = diag(\mathbb{V}_{R_{*1}}, \mathbb{V}_{R_{*2}} ... \mathbb{V}_{R_{*p}})$. m and p represent the number of points in X_{L_*} and X_{R_*}, respectively.

$GPIS_{L_* R_*}$ can be used to fuse two continuous surface estimates with uncertainties. Therefore, it may allow for a *consistency check* between the laser and radar perception prior to fusion. The data that passes this test should be fused to obtain a refined estimate, while the inconsistent data should not be fused. The implementation of this consistency check is left to future work.

4 Experimental Setup

4.1 Data Collection

Experiments were conducted with the Argo UGV (Fig. 2), used as a data collection platform. The Argo is equipped with a laser range finder, a mm-wave radar, and a *cm*-accuracy 6-DOF dGPS/INS localisation unit. The laser sensor is a Sick LMS-291 (range resolution: $0.01m$ and angular resolution: $0.25°$) and the radar is a 94 *GHz* Frequency Modulated Continuous Wave (FCMW) radar, custom-built at ACFR (range resolution: $0.2m$ and angular resolution: $2°$). The laser and radar were directed at the front of the vehicle with a constant nodding angle, so that the center of the beam intersected the ground at a look-ahead distance of approximately $11.4m$. The sensor data, along with the navigation data, were collected by the platform while it was moving around a rural environment. Consequently, the errors in the resulting 3D points were the result of the combination of 3 error sources: sensor noise, calibration and localisation. A detailed description of the platform, sensors and the datasets can be found in [9]. In particular, objects of different geometries (listed in Table 1) were partially scanned by the sensors on the UGV from distances varying from $2m$ up to $30m$ and used to evaluate the performance of the surface reconstruction techniques.

Table 1 List of objects, sizes and number of ground truth points

Object	Comp.	Car	Wall	Wall2	Trailer	Pole	Pole2	Fence
Dim. (m^3)	3.3x1.7x1.4	2.9x2.8x2.0	14x3.1x1.7	9.0x2.5x9.0	4.6x4.6x0.4	0.4x0.4x0.4	0.4x0.4x0.4	4.5x4.5x4.5
Nb. Pts.	7,958	22,736	16,738	15,584	18,784	2,578	1,032	6,470

4.2 Data Pre-processing

The data pre-processing follows the process described in Fig. 4. The data provided by the laser sensor consist of a single range value for each bearing angle in its scan, which are the result of the target extraction developed by the sensor manufacturer. The raw radar data consist of multiple range values, with corresponding intensities, for each bearing angle in its scan (note that the radar beam is much larger than the laser's). The extraction of the targets from the noise in the data is achieved using the

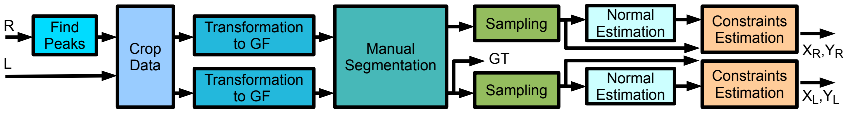


Fig. 4 Data Preparation Process

approach presented by the authors in [4]. We first extract peaks (i.e. local maxima) for each bearing angle. Most robotics applications only consider the highest peak (global maxima) as a target detected by the radar, thereby using the radar as a laser. However, this leads to the loss of useful information contained in the rest of the radar beam. Therefore, along with the highest peaks, we also extract local maxima that correspond to secondary targets by using an adaptive intensity threshold [4]. The result is a set of 3D points per scan, similar to a the data provided by a multi-echo laser sensor.

Laser and radar raw scans are then cropped to only keep data where the two sensors' FOVs overlap. Laser and radar points are then transformed into a common global navigation frame (GF). This transformation is obtained by combining the output of a prior extrinsic sensor calibration (using the technique in [16]) with the localisation of the Argo. The object of interest is then manually segmented from the full point cloud obtained with each sensing modality. A segmented object is a dense 3D point cloud, scanned from different perspectives. To evaluate the performance of the object reconstruction techniques, a small set of 5% of the data points is randomly sampled from the object. The rest of the *laser* data (i.e. 95% of the point cloud), constituting a dense point cloud, is used to build a ground truth (GT) (see Sec. 4.3). For each sampled observation provided by the range sensor, the normal to the surface of the object at that point, N_i, is approximated by the perpendicular to the segment between this point and the closest point in the same scan. Points inside and outside the surface are computed using the *normal value constraints* [15]. We place two points on N_i: one outside the object at a given distance $d = -0.5m$ from the surface, and one inside at a distance $d = 0.2m$ (see Fig. 5). In the GPIS, these points constitute positive ($f(x) = 1$) and negative ($f(x) = -1$) constraints, respectively, and populate X_L, Y_L and X_R, Y_R which are used as input data of the GPIS, for the laser and radar, respectively.

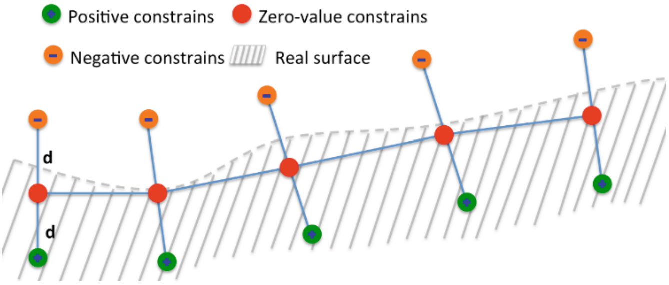


Fig. 5 The positive and negative constraints, given range sensor observations (red dots). The noise-free observations are assumed to be on the surface, represented by the dashed line.

4.3 Ground Truth (GT)

To quantify the errors made in the reconstructions, we used 95% of the full-resolution laser point cloud to obtain ground truth data, since the laser is the most accurate sensor available in our system. In this paper, the mean distance between any point in the full resolution point cloud and its closest neighbour was $3cm$. In some cases, due to the limitations of laser sensing, this dense point cloud of the object has to be corrected or completed to better reflect the actual surface of the object. For example, the side windows of the car shown in Fig. 8(a) were poorly represented (see Fig. 8(b) before correction), since the laser did not provide many returns from the windows surface. Therefore, points on the windows were manually added to complete the ground truth, using a resolution and noise level comparable to those of the original point cloud.

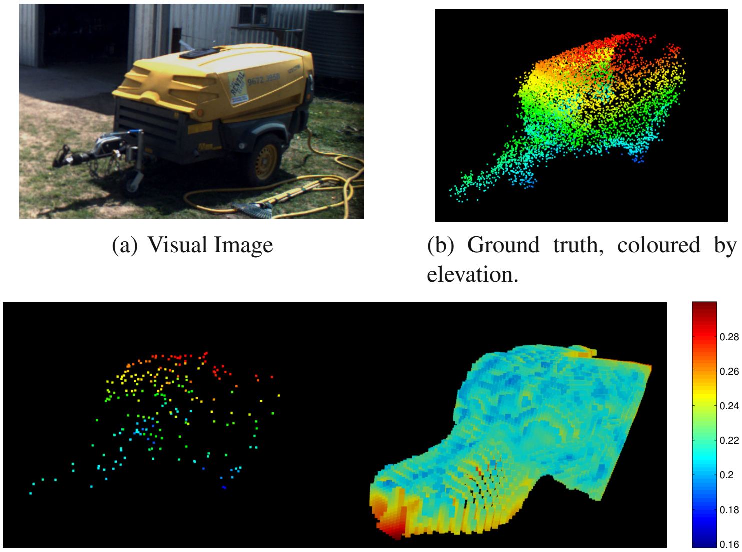
5 Experimental Results

5.1 Single-Sensing-Modality GPIS

Surfaces were estimated from laser (X_L) and radar (X_R) data separately, using the $GPIS_L$ and $GPIS_R$ processes described in Sec. 3.3. We evaluated the surfaces obtained with different resolutions of input data, and with the different covariance functions mentioned in Sec. 3.2. For all evaluations herein we queried the GPIS using a 3D grid of points (see x_* in Fig. 1) uniformly spaced at a resolution of $1cm$. In each of the 3D grid cells, values where $f(x_*) = 0$ were selected to represent the surface estimate. An example of a 3D surface reconstruction generated using GPIS with 376 input points is showed in Fig. 6. From a limited number of points, the general geometry of the object was recovered. A quantitative evaluation of the accuracy of surface estimates was performed by computing the 3D distances between points extracted from the estimated surfaces and our ground truth (the full resolution laser point cloud). Error statistics for each object were then obtained by calculating the root mean square (RMS) of these distances, providing an RMS error (RMSE) for each object. The analysis was performed for all 8 objects we considered, but for conciseness, we only show the RMSE for the different surface estimates of the compressor, in Fig. 7.

With an RMSE of $0.069m$ and $0.125m$ for laser and radar GPIS surfaces respectively, the exponential covariance function outperforms ($\gamma = 1$ in Eq. (4)) all the other kernels. Furthermore, the RMSE decreases as more sampled points are used, until the error does not change significantly.

Surface estimates obtained with $GPIS_L$ are more accurate than with $GPIS_R$. This was expected considering the higher accuracy and resolution of the laser sensor (see Sec. 4.1. However, in some cases radar surfaces showed an extended coverage of the object compared with laser surfaces. For example, Fig. 8(c) shows that most parts of the car are quite well modelled by $GPIS_L$, but the surfaces of the windows

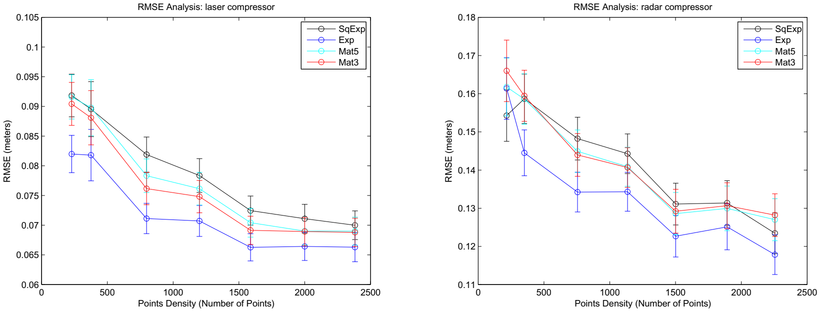


(a) Visual Image

(b) Ground truth, coloured by elevation.

(c) Left: Input laser points, coloured by elevation. Right: $GPIS_L$ surface. (d)

Fig. 6 Surface reconstruction of the compressor from sparse laser data. (a) Visual image of the compressor. (b) Full resolution 3D laser point cloud, used as ground truth only. (c) Surface estimated by $GPIS_L$ (right) using 376 input laser points (left). The surface is coloured by uncertainty (variance), using the colour scale shown in (d).

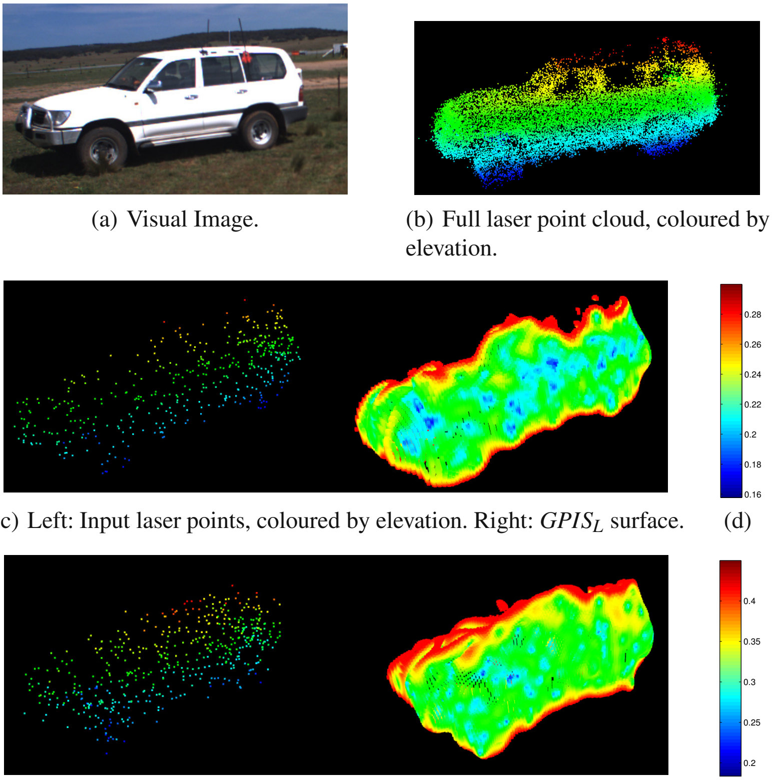


(a) RMSE with $GPIS_L$

(b) RMSE with $GPIS_R$

Fig. 7 RMSE of the compressor surface estimates obtained with $GPIS_L$ (a) and $GPIS_R$ (b) for different kernels and input point densities. Note the different scales of the errors obtained (larger errors with radar data).

are incomplete, since they are partially transparent to laser sensing. On the other hand, the radar perception of these windows is more complete, resulting in a more complete representation using ($GPIS_R$) (Fig. 8(e)).



(a) Visual Image.

(b) Full laser point cloud, coloured by elevation.

(c) Left: Input laser points, coloured by elevation. Right: *$GPIS_L$* surface.

(d)

(e) Left: Input radar points, coloured by elevation. Right: *$GPIS_R$* surface.

(f)

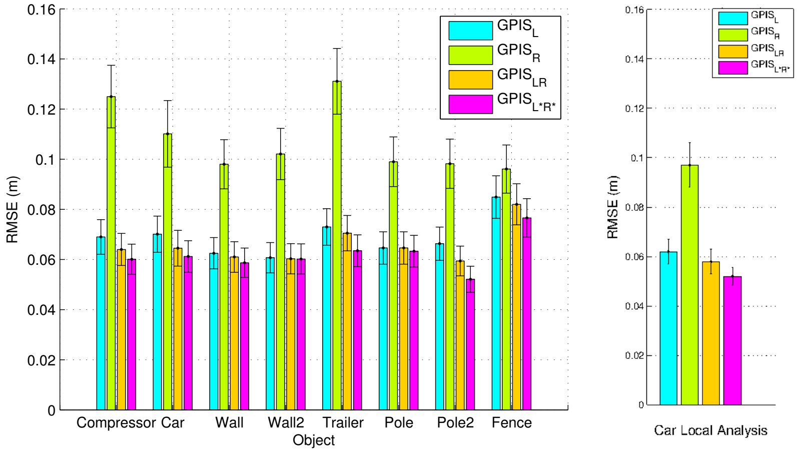
Fig. 8 Surface reconstruction of the car (a) using sparse laser or radar data. (b) Full resolution laser point cloud, used to build the ground truth data (shown *before* correction of the windows). (c) Surface estimated by $GPIS_L$ and (d) Surface estimated by $GPIS_R$. 1136 input points are used in each case. The surfaces are coloured by variance. Note the difference in scale between the variances for $GPIS_L$ and $GPIS_R$, the latter showing higher uncertainty overall. The edges of the car show high uncertainty since the car was only partially observed by both sensors.

5.2 Laser-Radar Data Fusion

Laser and radar data were fused using the $GPIS_{LR}$ and $GPIS_{L_*R_*}$ fusion methods described above. Table 2 shows the RMSE obtained using different covariance functions to estimate the surface of the compressor. The most accurate results seem to be obtained with the exponential kernel. However, the improvement is relatively insignificant, in particular for the $GPIS_{L_*R_*}$ method, whose accuracy is consistently the highest. We evaluated the performance of surface estimation on the 8 different objects listed in Table 1 using each GPIS method considered in this paper. Fig. 9(a) shows the RMSE for each surface estimate obtained using the exponential

Table 2 RMSE (in *m*) obtained for the compressor using different covariance functions

	$GPIS_L$	$GPIS_R$	$GPIS_{LR}$	$GPIS_{L_*R_*}$
SqExp	0.074	0.134	0.071	**0.063**
Exp	0.069	0.125	0.064	**0.060**
Mat3	0.071	0.132	0.069	**0.063**
Mat5	0.074	0.134	0.071	**0.063**

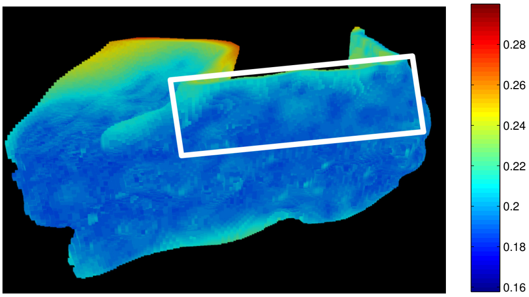


(a) Global errors for all objects

(b) Local errors (car windows).

Fig. 9 (a) RMSE (in *m*) obtained using the different *GPIS* methods to estimate the surfaces of all 8 objects. (b) Local analysis of the RMSE in the area of the car windows, shown in Fig. 10. The black error bars represent the standard deviation.

covariance function. As an example, Fig 10 shows the estimated surface of the car using $GPIS_{L*R_*}$. The estimation of the surfaces of the compressor, the car and the second pole (Pole2) were significantly improved by the fusion ($GPIS_{LR}$). In addition, further improvement was obtained using the alternative fusion method ($GPIS_{L_*R_*}$). The trailer was not significantly better represented by the state-of-the-art fusion method ($GPIS_{LR}$), however, the improvement is clearer when using ($GPIS_{L_*R_*}$). For the other objects, such as the walls, the improvements are less significant. To better illustrate some of the benefits of the laser-radar fusion, Fig.9(b) proposes a local analysis of the errors obtained with the car, focussing on the area of the windows (see white box in Fig. 10). These results show that due to the poor perception of the windows by the car, the accuracy of the surface estimation was particularly improved by the laser-radar fusion process.



(a) Surface estimated by $GPIS_{L*R_*}$.

(b)

Fig. 10 Surface of the car (seen in Fig. 8(a)), reconstructed using sparse laser and radar data (1136 points) fused with the $GPIS_{L*R_*}$ method. The surface is coloured by uncertainty, with low uncertainty shown in blue (see corresponding colour bar in (b)). Note that the windows of the car were correctly reconstructed, with low uncertainty and no holes. The white box delimitates the area used to perform the local analysis of errors in the area of the windows.

6 Conclusion

In this paper, we proposed an experimental analysis of the performance of continuous 3D surface reconstruction from laser and mm-wave radar data using Gaussian Process Implicit Surfaces, in a realistic field robotics scenario. We evaluated the performance of single-sensor approaches with different resolutions of input data and different kernels. We also compared the performance of these approaches with a state-of-the-art fusion approach and a new alternative method to multi-sensor data fusion. The GPIS fusion showed a significant improvement of the surface representations, especially when taking advantage of the complementarity of the two sensor modalities (e.g. in the case of the car windows, consistently detected by the radar but not by the laser). The proposed fusion process $GPIS_{L_*R_*}$ outperformed the state-of-the-art fusion method.

The next step of this work will be to implement a test within the $GPIS_{L_*R_*}$ framework to validate the consistency between the estimates obtained using laser and radar perception separately, prior to fusion. This will allow for higher resilience in challenging conditions, when laser and radar may not detect the same targets (e.g. in the presence of airborne dust, as in [4]). In this paper, the 3D data were manually segmented before applying the GPIS approaches. A direct extension will be the integration of automatic data segmentation in the same framework, or by analysing large scale environmental datasets rather than pre-segmented objects. In addition, the heterocedasticity treatment can be extended to sets of scans rather using a single noise parameter for all the points provided by each sensing modality.

Acknowledgements. The authors thank Simon O'Callaghan for his input. This work was supported in part by the ACFR and the NSW State Government. This material is based on research sponsored in part by the Air Force Research Laboratory, under agreement number FA2386-10-1-4153. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

1. Brooker, G.: Sensors for Ranging and Imaging. SciTech Publishing, Inc. (2009)
2. Dragiev, S., Toussaint, M., Gienger, M.: Gaussian process implicit surfaces for shape estimation and grasping. In: IEEE International Conference on Robotics and Automation (2011)
3. El-Beltagy, M.A., Wright, W.A.: Gaussian processes for model fusion. In: Dorffner, G., Bischof, H., Hornik, K. (eds.) ICANN 2001. LNCS, vol. 2130, pp. 376–383. Springer, Heidelberg (2001)
4. Gerardo-Castro, M.P., Peynot, T.: Laser-to-radar sensing redundancy for resilient perception in adverse environmental conditions. In: ARAA Australasian Conference on Robotics and Automation (2012)
5. Hollinger, G.A., Englot, B., Hover, F.S., Mitra, U., Sukhatme, G.S.: Active planning for underwater inspection and the benefit of adaptivity. International Journal of Robotics Research 32(1) (2013)
6. Kapoor, A., Ahn, H., Picard, R.: Mixture of gaussian processes for combining multiple modalities. In: International Workshop on Multiple Classifier Systems (2005)
7. O'Callaghan, S.T., Ramos, F.T., Durrant-Whyte, H.: Contextual occupancy maps incorporating sensor and location uncertainty. In: IEEE International Conference on Robotics and Automation (2010)
8. Ohtake, Y., Belyaev, A., Alexa, M., Turk, G., Seidel, H.: Multi-level partition of unity implicits. In: ACM SIGGRAPH Courses (2005)
9. Peynot, T., Scheding, S., Terho, S.: The Marulan Data Sets: Multi-Sensor Perception in Natural Environment with Challenging Conditions. International Journal of Robotics Research 29(13) (2010)
10. Peynot, T., Underwood, J., Scheding, S.: Towards reliable perception for unmanned ground vehicles in challenging conditions. In: IEEE/RSJ International Conference on Robotics and Intelligent Systems (2009)
11. Plagemann, C., Kersting, K., Pfaff, P., Burgard, W.: Gaussian beam processes: A nonparametric bayesian measurement model for range finders. In: Robotics: Science and Systems III (2007)
12. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. The MIT Press (2006)
13. Whelan, T., et al.: Robust tracking for real-time dense rgb-d mapping with kintinuous. Technical Report MIT-CSAIL-TR-2012-031, CSAIL, MIT (2012)
14. Thrun, S., Burgard, W., Fox, D.: A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. In: IEEE International Conference on Robotics and Automation (2000)
15. Turk, G., O'brien, J.F.: Variational implicit surfaces. Technical Report GIT-GVU-99-15, Georgia Institute of Technology (1999)
16. Underwood, J.P., Hill, A., Peynot, T., Scheding, S.J.: Error modeling and calibration of exteroceptive sensors for accurate mapping applications. Journal of Field Robotics 27(1) (2010)
17. Vasudevan, S.: Data fusion with gaussian processes. Robotics and Autonomous Systems 60(12) (2012)
18. Vieira, A.W., Drews, P.L.J., Campos, M.F.M.: Efficient change detection in 3d environment for autonomous surveillance robots based on implicit volume. In: IEEE International Conference on Robotics and Automation (2012)
19. Williams, O., Fitzgibbon, A.: Gaussian process implicit surfaces. In: Gaussian Processes in Practice Workshop (2007)

Cluster-Based SJPDAFs for Classification and Tracking of Multiple Moving Objects

Naotaka Hatao and Satoshi Kagami

Abstract. This paper describes a method for classifying and tracking multiple moving objects with a laser range finder (LRF). As moving objects are tracked in the framework of sample-based joint probabilistic data association filters (SJPDAFs), the proposed method is robust against occlusions or false segmentation of LRF scans. It divides tracking targets and corresponding LRF segments into clusters and able to classify each cluster as a car or a group of pedestrians. In addition, it can correct false segmentation of LRF scans. We implemented the proposed method and obtained experimental results demonstrating its effectiveness in outdoor environments and crowded indoor environments.

1 Introduction

This paper describes a method using a horizontal laser range finder (LRF) to track and classify multiple moving objects. Mobile robots need to detect and track moving objects, and moving object tracking can also be used in traffic stream measuring, security systems, and so on.

It is generally difficult to identify individual moving objects from horizontal LRF scans in crowded areas because the shapes of objects are not static, and some moving objects are occluded by other moving objects. Multiple hypothesis approaches are often efficient at coping with these uncertainties. Joint probabilistic data association filters (JPDAFs) [1] and multiple hypothesis tracking (MHT) are well-known tracking algorithms using multiple hypothesis methods [2], and several modifications of these algorithms have been proposed to improve tracking performance in cluttered environments [3][4][5]. These algorithms enumerate the correspondences between features extracted from sensor data and moving objects and build hypotheses using

Naotaka Hatao · Satoshi Kagami
National Inst. of AIST, 2-3-26 Aomi, Koto-ku, Tokyo, Japan
e-mail: n.hatao@aist.go.jp

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_21, © Springer International Publishing Switzerland 2015

sets of these correspondences. As these algorithms can handle false positives and negatives, they are able to track moving objects reliably even in noisy environments.

Particle filters are robust methods for tracking moving objects, but normal particle filters are unsuitable for tracking multiple targets because particles easily become gathered around a single target. Although several methods have been proposed to overcome this problem [6][7], they do not have a mechanism to treat false positives or negatives. Schulz et al. proposed the use of sample-based JPDAFs (SJPDAFs), which use particle filters instead of Kalman filters used in normal JPDAFs [8].

It is important to identify categories of moving objects. Because the shapes of LRF scans corresponding to moving objects are not stable, many researchers have used time-series LRF scans to classify such objects [9][10][11]. Their methods, however, assume that pedestrians in a group have a same velocity vector and do not explicitly treat the merging and separation of groups of moving objects.

Candidates of moving objects are extracted from LRF scans by dividing the scan points into segments, and segmenting LRF scans correctly is also a difficult problem. If the gap between adjacent LRF scan points exceeds a threshold distance, it becomes a boundary of segments.Segments, however, do not always correspond one-to-one with moving objects. For example, pedestrians often walk so close to each other that an LRF cannot measure the space between them. And LRF scans corresponding to a car are often divided into several parts.

In this paper, we propose an classification and tracking method based on SJPDAFs. SJPDAFs are robust against false positives and negatives, and they make it possible to flexibly design individual trackers using particle filters. The proposed method has two advantages to track moving objects effectively in the real world. The first is that it generates additional hypotheses to cope with the above-mentioned false segmentation. The second is that it divides tracking targets and corresponding LRF segments into clusters and classify each cluster as a car or a group of pedestrians. The numbers of pedestrians in clusters are also estimated, and each pedestrian is tracked individually. Individual tracking enables the merging or splitting of clusters.

This paper is organized as follows. An overview of the proposed method is presented in Section 2, and the methods for detecting moving object candidates are described in Section 3. The proposed method is described in greater detail in Section 4, experimental results are presented in Section 5, and a conclusion is given in Section 6.

2 Multiple Moving Object Tracking Using SJPDAFs

2.1 Overview of SJPDAFs

This section briefly introduces the concept of SJPDAFs. Suppose that n_k moving objects are being tracked and m_k features are measured at time k. For convenience we let i denote the ID number of a moving object and let j denote the ID number of a feature. $\boldsymbol{X}^k = \{\boldsymbol{x}_1^k, \cdots, \boldsymbol{x}_i^k, \cdots, \boldsymbol{x}_{n_k}^k\}$ and $\boldsymbol{Z}(k) = \{\boldsymbol{z}_1(k), \cdots, \boldsymbol{z}_j(k), \cdots, \boldsymbol{z}_{m_k}(k)\}$ denote

Table 1 Example hypotheses: the case of two moving objects and three features

	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	θ_{10}	θ_{11}	θ_{12}	θ_{13}
$\boldsymbol{x}_1^k$	$z_0(k)$	$z_0(k)$	$z_0(k)$	$z_0(k)$	$z_1(k)$	$z_1(k)$	$z_1(k)$	$z_2(k)$	$z_2(k)$	$z_2(k)$	$z_3(k)$	$z_3(k)$	$z_3(k)$
$\boldsymbol{x}_2^k$	$z_0(k)$	$z_1(k)$	$z_2(k)$	$z_3(k)$	$z_0(k)$	$z_2(k)$	$z_3(k)$	$z_0(k)$	$z_1(k)$	$z_3(k)$	$z_0(k)$	$z_1(k)$	$z_2(k)$
$m_k - \lvert\theta\rvert$	3	2	2	2	2	1	1	2	1	1	2	1	1

the state vectors of moving objects and the features, respectively. In addition, $\boldsymbol{Z}^k$ denotes the sequence of all features up to time k.

As mentioned above, SJPDAFs use particle filters to estimate the states of moving objects. SJPDAFs deploy N particles per moving object. The prediction and resampling steps of SJPDAFs are the same as those of normal particle filters, but the likelihoods of particles are calculated by the following expressions.

$$\omega_{i,n}^k = \alpha \sum_{j=0}^{m_k} \beta_{ji} p(\boldsymbol{z}_j(k)|\boldsymbol{x}_{i,n}^k) \tag{1}$$

$\boldsymbol{x}_{i,n}^k$ and $\omega_{i,n}^k$ denote the state vector and the likelihood of the nth particle in ith particle set at time k, respectively. β_{ji} is calculated as follows:

$$\beta_{ji} = \sum_{\theta \in \Theta_{ji}} [P(\theta|\boldsymbol{Z}^k)] \tag{2}$$

where

$$P(\theta|\boldsymbol{Z}^k) = \alpha\gamma^{(m_k-|\theta|)} \prod_{(j,i)\in\theta} \frac{1}{N} \sum_{n=1}^{N} p(\boldsymbol{z}_j(k)|\boldsymbol{x}_{i,n}^k). \tag{3}$$

θ and $P(\theta|\boldsymbol{Z}^k)$ respectively denote a hypothesis and its likelihood. γ denotes the probability that an observed feature is a false alarm (false positive), and $m_k - |\theta|$ denotes the number of false alarms in $\theta.\Theta_{ji}$ denotes the set of all hypotheses for which $\boldsymbol{z}_j$ corresponds to $\boldsymbol{x}_i$, and α represents a normalizer.

We assume that two objects are being tracked and three features are measured. All possible hypotheses are listed in Table 1, where $z_0(k)$ indicates a false negative. For example, in θ_5, $\boldsymbol{x}_1$ corresponds to $z_1(k)$ and no feature that corresponds to $\boldsymbol{x}_2$ is found. Also, in this example, Θ_{12} equals $\{\theta_2, \theta_9, \theta_{12}\}$.

The original implementation of SJPDAFs uses local minima of LRF scans as the features. It uses occupancy grid maps to calculate $p(\boldsymbol{z}_j(k)|\boldsymbol{x}_{i,n}^k)$, whereas the proposed method uses the shapes of contours of moving objects. See Sections 4.2 and 4.3 for details.

A major weakness of (S)JPDAFs is that they cannot estimate the number of moving objects, and the number of moving objects is estimated in a separate process. In the original implementation of SJPDAFs, it is assumed that the change in the number of moving objects follows a Poisson process, and the number of moving objects is calculated as follows:

$$
\begin{aligned}
P(N^k|\boldsymbol{M}^k) &= \alpha \cdot P(m_k|N^k,\boldsymbol{M}^k) \cdot P(N^k|\boldsymbol{M}^{k-1}) \\
&= \alpha \cdot P(m_k|N^k) \cdot \sum_n [P(N^k|N^{k-1}=n) \cdot P(N^{k-1}=n|\boldsymbol{M}^{k-1})] \qquad (4)
\end{aligned}
$$

N^k denotes the number of moving objects at time k, and $\boldsymbol{M}^k$ denotes the sequence of the number of features up to time k.

2.2 Cluster-Based SJPDAFs with Classification

This section describes the overview of the proposed method. The proposed method works both on LRFs fixed on the ground and on LRFs mounted on mobile robots.

In the original work on SJPDAFs, LRFs were mounted at a height of 40 cm and detected the legs of persons. In the proposed method, in contrast, LRFs are placed at about the height of a person's chest. This is because the detection of small objects such as legs at a large distance is unstable. In addition, if the LRFs on outdoor mobile robots are mounted low, the laser beams often hit the ground.

The original implementation of SJPDAFs estimates the number of people within the entire sensor area by assuming a Poisson process. In the real world, however, the areal density of moving objects varies considerably, thus a Poisson process is not suitable for representing the change in the number of moving objects. The proposed method therefore uses cluster-based SJPDAFs. A "cluster" in this paper means a set of SJPDAF particle filter components and corresponding LRF segments. "Segment" means a set of LRF scan points, and the boundaries of segments are placed where the gaps between adjacent scan points are large. Estimation of the number of pedestrians and classification are performed for each cluster.

The proposed tracking method is performed as follows:

1. Moving object candidates are extracted from the latest LRF scan. Each candidate is an LRF segment. If the "Extended Trajectories" method described in Section 3 is used, initial grouping of segments is performed.
2. Particles in existing clusters are updated according to $p(\boldsymbol{x}^k_{i,n}|\boldsymbol{x}^{k-1}_{i,n})$, and corresponding segments are enumerated.
3. Merging and splitting of clusters are performed. If two particle filters belonging to different clusters share a corresponding segment or grouped segment, those clusters are merged. In contrast, if a set of particle filters no longer shares corresponding segments with the remaining particle filters in the same cluster, the set of particle filters is split as a new cluster. (see Section 4.4 for details)
4. Each cluster is updated independently. (see Sections 4.2 and 4.3 for details)
5. Classification and estimation of the number of pedestrians are performed for each cluster. (see Section 4.1 for details)
6. New clusters are initialized for segments that are not associated with existing clusters.

3 Extraction of Candidate Moving Objects and Initial Velocity Estimation

Before moving objects can be tracked, candidate moving objects must be extracted from LRF scans. Three extraction methods are implemented in the proposed method.

1. **Occupancy grid map with a polar coordinate system** for fixed LRFs
2. **Occupancy grid map with a Cartesian coordinate system** for indoor robots
3. **Extended Trajectories** for outdoor robots

Extraction of moving object candidates using occupancy grid map is a common method. Grid maps are generated in advance before starting tracking. Each occupancy grid cell has one of three states: “free,” “occupied,” or “unknown.” “Occupied” means that there are static objects in the grid cell, whereas “free” means that there are no objects in the grid cell. If LRF scans appear in free grid cells, those scans might be associated with moving objects.

If the LRFs are fixed in the ground, grid maps with polar coordinate systems are selected. The origins of the polar coordinate systems are same as the measurement origins of the LRFs. This method has the lowest computational cost. If the LRFs are mounted on robots that move in indoor environments (i.e., they can estimate their coordinates precisely), grid maps with a Cartesian coordinate systems are selected.

If, on the other hand, the LRFs are mounted on robots that move outdoor environments, the “extended trajectories” method described below is used. This method has two advantages. The first is that it can estimate velocities. Since there are fast-moving objects in outdoor environments, tracking often fails if the initial velocity vectors follow a zero-centered Gaussian distribution. The second advantage is that the proposed method can perform initial grouping of LRF segments.

The “Extended Trajectories” extraction method also uses a grid map in which each grid cell has one of three states: “Unknown”, “Foreground”, “Background”. LRF scans are transformed to global coordinates using the odometry of the robot. Candidate moving objects at time k are extracted according to the following procedure.

1. Grids within L_1 mm of LRF scan points from time $k-t_1$ to time k are set as “Foreground.”
2. Grids within L_2 mm of LRF scan points from time $k-t_2$ to time $k-t_1$ are set as “Background.” Grids that are already set as Foreground are overwritten.
3. Foreground regions adjacent to Background regions are enumerated. Isolated foreground regions derive from false positives.
4. If an LRF scan segment at time k is on a Foreground region enumerated in step 3, it is extracted as a candidate moving object.

Fig. 1 shows the results of moving object initialization. This procedure builds “extended” trajectories of moving objects. That is, if there are moving objects, foreground regions growing on background regions appear.

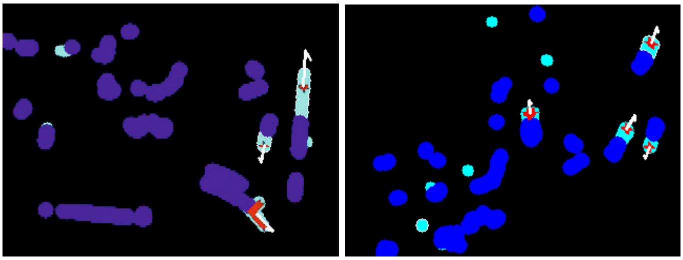


Fig. 1 Results of moving object initialization. Blue and aqua regions respectively indicate "Background" and "Foreground" regions. Red points and white arrows respectively indicate extracted candidate moving objects and their velocities.

The velocity vector of the *j*th candidate is estimated using the following expressions.

$$|\overline{v_j}| = \frac{\sqrt{(\overline{x_{o_j}} - \overline{x_{b_j}})^2 + (\overline{y_{o_j}} - \overline{y_{b_j}})^2} + L_2}{t_1} \tag{5}$$

$$\overline{\theta_j} = \arctan((\overline{y_{o_j}} - \overline{y_{b_j}}), (\overline{x_{o_j}} - \overline{x_{b_j}})) \tag{6}$$

$\overline{\boldsymbol{x}_{o_j}} = (\overline{x_{o_j}}, \overline{y_{o_j}})$ denotes the centroid of LRF scan points in the corresponding foreground region, and $\overline{\boldsymbol{x}_{b_j}} = (\overline{x_{b_j}}, \overline{y_{b_j}})$ denotes the centroid of border grids between the foreground and background regions. $|\overline{v_j}|$ and $\overline{\theta_j}$ respectively denote the estimated values of the velocity and angle of the *j*th candidate. In Eq.(5), the moving distances are approximated by sums of L_2 (the size of expansion) and the distances between centroids of border grids and the centroids of the current LRF scan points.

These estimated values are used for initialization of particles in SJPDAFs. If there are two background regions adjacent to the corresponding foreground region, two estimated values of velocity vectors are generated. In this case, particles are divided into two groups and each group is given a separate velocity vector.

Initial grouping of LRF segments is performed using extended trajectories. Segments in the same foreground region belong to the same group. In this process, separated LRF segments associated with a car belong to the same group.

4 Moving Object Classification and Tracking Classification Using Cluster-Based SJPDAFs

The proposed method divides LRF segments and particle filters into several clusters, and classification and tracking are performed independently for each cluster.

After classification of each cluster is performed, corresponding particle filters are applied. The proposed method assumes that LRF scans corresponding to pedestrians form cylindrical shapes and those corresponding to cars form rectangular shapes. It also uses different methods to build hypotheses for pedestrians and cars.

Table 2 Class definition of SVM

c_0	c_1	c_2	c_3	$\cdots$
false positive	one car	one pedestrian	two pedestrians	$\cdots$

4.1 Classification and Estimation of Number of Moving Objects

The proposed method performs classification and estimation of the number of pedestrian using a support vector machine (SVM). As the shapes of LRF scan segments are not stable, the method uses a time-series estimation.

The class definition of moving objects is shown in Table 2, where c_n indicates the label of each class.

We define the feature vector of LRF scans in a cluster as $\boldsymbol{z}_f(k)$ and define a set of feature vectors from time 0 to k as $Z_f^k = \{\boldsymbol{z}_f(0) \cdots \boldsymbol{z}_f(k)\}$. The value we want to estimate is $P(c_n|Z_f^k)$, and using the same assumption as in Eq.(4) we obtain

$$P(c_n(k)|Z_f^k) = \alpha \cdot P(z_f(k)|c_m(k)) \cdot P(c_m(k)|Z_f^{k-1}) \tag{7}$$

$$P(c_n(k)|Z_f^{k-1}) = \sum_m [P(c_n(k)|c_n(k-1) = m) \cdot P(c_n(k-1) = m|Z_f^{k-1})] \tag{8}$$

And from Bayes' theorem we obtain

$$P(\boldsymbol{z}_f(k)|c_n) = \alpha \frac{P(c_n|\boldsymbol{z}_f(k))}{P(c_n)} \tag{9}$$

$P(c_n|\boldsymbol{z}_f(k))$ can be estimated using an SVM, and $P(c_n)$ can be estimated using the SVM training sets. The features for the SVM are defined as followed:

z_{f0} : Number of LRF segments
z_{f1} : Sum of lengths of LRF segments
z_{f2} : Average speed
z_{f3} : Difference between angle of directed bounding box and angle of average velocity vector
z_{f4} : Length of long side of directed bounding box
z_{f5} : Length of short side of directed bounding box
z_{f6} : Residual error between directed bounding box and LRF scan points

Once a cluster is classified as a "False Positive," all corresponding particle filters are removed and tracking finishes. If the cluster class changes from "car" to "pedestrian(s)" or from "pedestrian(s)" to "car", new particle filters are deployed. In these cases, old particle filters are not removed immediately and continue to be tracked in preparation for false classifications. The average velocity vector of old particle filters is carried on as the initial velocity vector of new particle filters.

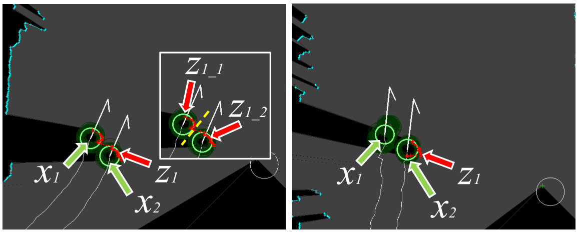


Fig. 2 Results of tracking a group of pedestrians. Dark green circles denote individual particles. Light green circles and white arrows denote results of particle filters (position, radius and velocity). White circles denote the robot, and black regions denote occluded areas.

Table 3 Hypothesis likelihood results for Fig. 2 left

	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9
$\boldsymbol{x}_1^k$	$z_0(k)$	$z_0(k)$	$z_0(k)$	$z_0(k)$	$z_1(k)$	$z_{1-1}(k)$	$z_{1-2}(k)$	$z_{1-1}(k)$	$z_{1-2}(k)$
$\boldsymbol{x}_2^k$	$z_0(k)$	$z_1(k)$	$z_{1-1}(k)$	$z_{1-2}(k)$	$z_0(k)$	$z_0(k)$	$z_0(k)$	$z_{1-2}(k)$	$z_{1-1}(k)$
$P(\theta\|Z^k)$	0.0054	8.7e-7	1.1e-9	0.027	3.2e-15	0.16	6.3e-21	0.80	1.2e-27

Table 4 Hypothesis likelihood results for Fig. 2 right

	θ_1	θ_2	θ_3
$\boldsymbol{x}_1^k$	$z_0(k)$	$z_0(k)$	$z_1(k)$
$\boldsymbol{x}_2^k$	$z_0(k)$	$z_1(k)$	$z_0(k)$
$P(\theta\|Z^k)$	2.8e-9	0.93	0.068

4.2 Tracking and Hypothesis Building for Pedestrians

The proposed method assumes that pedestrians have a cylindrical shape. The nth particle for the ith pedestrian at time k is defined as follows:

$$\boldsymbol{x}_{i,n}^{\boldsymbol{k}} = (x_{i,n}^k, y_{i,n}^k, v_{i,n}^k, \theta_{i,n}^k, r_{i,n}^k)^T \tag{10}$$

where $v_{i,n}^k$, $\theta_{i,n}^k$ and $r_{i,n}^k$ respectively denote the velocity, direction, and radius of the cylinder. The likelihood of a particle is calculated using sum of the distances from LRF scan points in a corresponding segment to the circumference of the particle.

$$p(z_j(k)|x_{i,n}^k) = \frac{1}{\sqrt{2\pi\sigma_p}} \exp\left(-\frac{\Sigma_{l=0}^{m_j^k}(d_{i,n,l}^k)^2}{2m_j^k\sigma_p^2}\right) \tag{11}$$

$$d_{i,n,l}^k = |\hat{x}_{i,n}^k - z_{j,l}(k)| - r_{i,n}^k \tag{12}$$

$z_{j,l}(k)$ denotes the lth LRF scan point in the jth LRF segment at time k, m_k denotes the number of LRF scans in the corresponding segment, and $\hat{x}_{i,n}^k = (x_{i,n}^k, y_{i,n}^k)$ denotes the position of the nth particle. σ_p is a constant.

When several pedestrians close together are moving, segmentation of a LRF scan often fails and large LRF segments are obtained. To cope with this problem, the method generates hypotheses in which large segments are divided into several small segments. How to divide the segments is decided on the basis of a single-step result of the SVM $P(\boldsymbol{z}_f(k)|c_n)$ described in Sections 4.1. For example, Hypothesis generation results corresponding to the left side of Fig. 2 are listed in Table 3. In this case, although only one segment is detected, the class that has the largest $P(\boldsymbol{z}_f(k)|c_n)$ is c_3 (two pedestrians). Thus, hypotheses in which the segment is divided into two segments are built. $\boldsymbol{z}_{1-1}(k)$ and $\boldsymbol{z}_{1-2}(k)$ denote divided segments. The most reliable hypothesis is θ_8, in which each tracker corresponds to divided segments.

Hypothesis generation results corresponding to the right side of Fig. 2 are listed in Table 4. The class that has the largest $P(\boldsymbol{z}_f(k)|c_n)$ is c_2 (one pedestrian) and dividing of segments does not occur. In this case the left pedestrian is hidden by the right pedestrian, and the most reliable hypothesisθ_2 supports this status.

4.3 Tracking and Hypothesis Building for Cars

There are two types of LRF scans corresponding to cars. One is associated with the car body, and the other is associated with the window frames. Both, however, are approximately rectangular.

Thus the *n*th particle for the *i*th car at time *k* is defined as follows:

$$\boldsymbol{x}_{i,n}^{\boldsymbol{k}} = (x_{i,n}^k, y_{i,n}^k, v_{i,n}^k, \theta_{i,n}^k, Ll_{i,n}^k, Lw_{i,n}^k)^T \tag{13}$$

where $Ll_{i,n}^k$ and $Lw_{i,n}^k$ respectively denote the lengths of the long and short sides of the rectangle.

$$p(z_j(k)|x_{i,n}^k) = \frac{1}{\sqrt{2\pi}\sigma_c}\exp(-\frac{\sum_{l=0}^{m_j^k}(d_{i,n,l}^k)^2}{2m_j^k\sigma_c^2}) \tag{14}$$

$$d_{i,n,l}^k = \min_{x_l \in L_{i,n}^k} |z_{j,l}(k) - x_l| \tag{15}$$

$L_{i,n}^k$ denotes the rectangle constructed using $x_{i,n}^k$. σ_c is a constant.

Unlike LRF scan segments corresponding to pedestrians, those corresponding to cars are often separated. The proposed method therefore generates hypotheses in which several segments correspond to one tracking target. Fig. 3 shows a result of car tracking. Although the shapes of individual LRF segments change dynamically, the overall shape remains rectangular. Hypothesis generation results corresponding to Fig. 3(1) and Fig. 3(2) are listed in Table 5 and Table 6. (where hypotheses that have small likelihoods are omitted). θ_{17} in Table 5, for instance, means that $\boldsymbol{z}_1$, $\boldsymbol{z}_2$, and $\boldsymbol{z}_4$ correspond to the car, and $\boldsymbol{z}_3$ and $\boldsymbol{z}_5$ are false positives. The most reliable hypothesis Table 5 is θ_{28}, in which $\boldsymbol{z}_3$(the scan segment corresponding to the driver

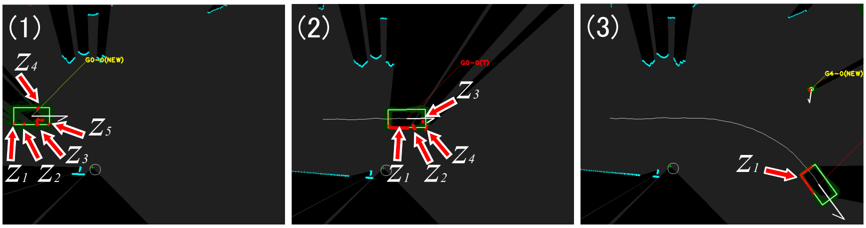


Fig. 3 Result of tracking a car. The car was climbing up a ramp, and LRF firstly detected the window frames, and then detected the body. Dark green squares denote individual particles. Light green squares and white arrows denote results of particle filters (position, angle, size of rectangle, and velocity)

.

Table 5 Example of hypotheses for a car (Fig. 3(1))

	θ_{17}	θ_{18}	θ_{21}	θ_{24}	θ_{28}	θ_{29}	θ_{31}
$\boldsymbol{x}_1^k$	z_1, z_2, z_4	z_1, z_2, z_5	z_1, z_4, z_5	z_2, z_4, z_5	z_1, z_2, z_4, z_5	z_1, z_3, z_4, z_5	z_1, z_2, z_3, z_4, z_5
$P(\theta\|Z^k)$	0.011	0.019	0.022	0.0071	0.63	0.048	0.24

Table 6 Example of hypotheses for a car (Fig. 3(2))

	θ_7	θ_9	θ_{10}	θ_{12}	θ_{13}	θ_{14}	θ_{15}
$\boldsymbol{x}_1^k$	z_1, z_4	z_2, z_4	z_3, z_4	z_1, z_2, z_4	z_1, z_3, z_4	z_2, z_3, z_4	z_1, z_2, z_3, z_4
$P(\theta\|Z^k)$	3.7e-4	3.1e-4	2.7e-4	0.016	0.018	1.5e-4	0.96

of the car) is a false positive. The most reliable hypothesis in Table 6, on the other hand, is θ_{14}, in which all segments are obtained from the car.

4.4 Merging and Splitting of Clusters

If a set of particle filters no longer shares corresponding segments with the remaining particle filters in the same cluster, it is split as a new cluster. In contrast, if two or more particle filters belonging to different clusters share a corresponding segment or grouped segment, these clusters are merged. The proposed method uses two types of merging: temporary merging and permanent merging.

If one or both of the clusters is classified as a car or they have different velocity vectors, temporary merging is used (Fig. 4). In this case, the hypotheses for SJPDAFs are created using all LRF segments and particle filters that belong to both clusters. In the current implementation, the $P(c_n|Z_f^k)$ values for clusters that are temporarily merged are updated individually. Corresponding LRF scan segments are selected based on the most reliable SJPDAF hypothesis. If one cluster departs from the other again and the "Foreground region" is divided into two, temporary merging is terminated. Using this framework, the proposed methods can treat clusters that contain both cars and pedestrians.

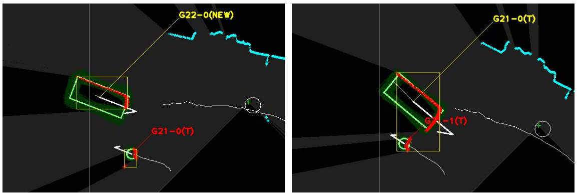


Fig. 4 Example of temporary merging: G21 (a pedestrian) and G22 (a car) were temporarily merged

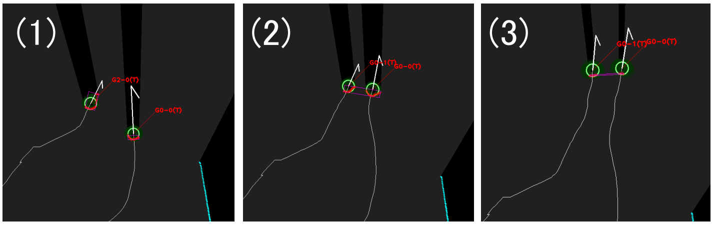


Fig. 5 Example of permanent merging: G0 (a pedestrian) and G2 (a pedestrian) were merged permanently because they had almost same velocity vectors

Table 7 Numbers and frames of training data set

class	c_0	c_1	c_2	c_3	c_4	c_5	c_6
	false positive	car	1 person	2 persons	3 persons	4 persons	5 or more persons
numbers	212	18	318	54	9	5	3
frames	868	899	20462	3631	352	267	58

Table 8 Results of tracking in an outdoor environment: Per-object accuracies

Target	Succeeded	Disrupted in occ.	Failed	Mismatch	FP
Pedestrian	89	16	2	4	41
Car	7	0	0	0	2

If both approaching clusters are classified as pedestrian groups and the differences between their average speeds and the directions of their velocity vectors are below the thresholds, permanent merging is used (Fig. 5). In this case, the $P(c_n|Z_f^k)$ values of both clusters are integrated.

5 Classification and Tracking Experiments and Evaluations

5.1 Classification and Tracking Experiments in an Outdoor Environment

This section describes experimental results obtained using the proposed method in an outdoor environment. An outdoor mobile robot equipped with a LRF(Hokuyo Top-URG) mounted about 800 mm above the ground was allowed to move over a distance of about 250 m in the University of Tokyo Hongo campus. It traveled the course back and forth four times, and LRF scans were obtained every 100 ms. The LRF scans obtained in first three runs were used to train the SVM, and the LRF scans obtained in the last run are used to evaluate accuracy. The total numbers of each class and the total numbers of frames detected each class are listed in Table 7. No groups that consist of more than five pedestrians are appeared in the training data, thus the the maximum number in a group is limited five.

Table 9 Results of classification in an outdoor environment: Per-object accuracies

Target	TP	FP	FN	Precision	Recall
Pedestrian	111	41	11	73.0%	91.0%
Car	4(3)	2	1	77.8%	87.5%

Table 10 Results of classification and tracking in an outdoor environment: Per-frame accuracies

Target	TP	FP	FN	Precision	Recall
Pedestrian	11116	1609	456	87.4%	96.1%
Car	344	44	42	88.6%	89.1%

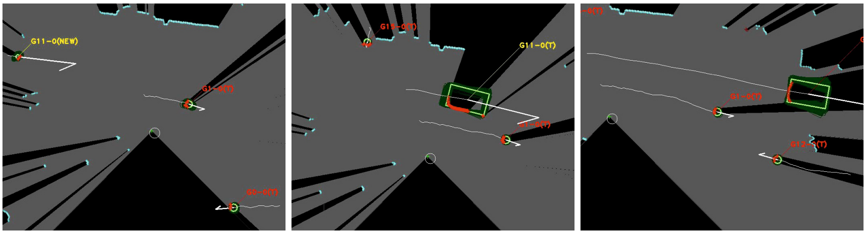
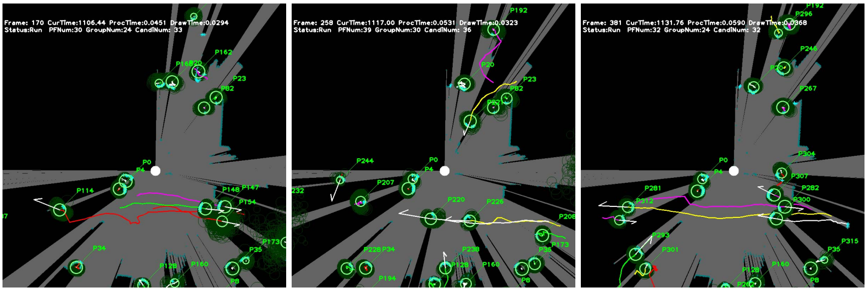


Fig. 6 Result of car tracking. G11 was initially classified as a pedestrian when it was far away from the robot because it initially produced a small LRF segment (left). When it approached the robot, however, the LRF segments became larger and it was correctly classified as a car (center). Tracking was terminated successfully (right).



```
Frame: 170 CurTime:1106.44 ProcTime:0.0451 DrawTime:0.0294
Status:Run  PFNum:30 GroupNum:24 CandiNum: 33
```

```
Frame: 258 CurTime:1117.00 ProcTime:0.0531 DrawTime:0.0323
Status:Run  PFNum:39 GroupNum:30 CandiNum: 36
```

```
Frame: 381 CurTime:1131.76 ProcTime:0.0590 DrawTime:0.0368
Status:Run  PFNum:32 GroupNum:24 CandiNum: 32
```

Fig. 7 Scenes of tracking experiment in a crowded environment. White circles indicate the position of the LRF.

Table 11 Tracking results: Per-object accuracies in a crowded environment

Target	Succeeded	Disrupted in occ.	Failed	Mismatch	FP
Pedestrian	625	114	12	37	2

The parameters defined in Section 3 were set to $t_1 = 1$ sec, $t_2 = 2$ sec, $L_1 = 300$ mm, and $L_2 = 500$ mm. These parameters are suitable for detecting objects moving at a velocity between 500 and 3000 mm/s. The number of particles per object was 500, and average processing time for one frame, including the processing time for the "Extended Trajectory" method described Section 3) was about 42.7 ms (Intel Core i7-940XM). Within the range of the LRF were 8 cars and 113 pedestrians.

The per-object accuracies of the tracking are listed in Table 8. The meaning of the states are the following.

1. **Succeeded**: Tracking completed successfully.
2. **Disrupted in occ.**: Tracking terminated one in an occluded area, and the same object was detected again
3. **Failed** Tracking failed in non-occluded area.
4. **Mismatch** The particle filter migrated another object.

Table 9 and Table 10 shows The per-objects accuracies of the classification and the per-frame accuracies of the tracking are listed in Table 9 and Table 10, where 'TP", "FP," and "FN" mean "True positive." "False positive," and "False Negative.". The number in parentheses in Table 9 is the number of cars first mistakenly classified as a group of pedestrians but later reclassified correctly. Fig. 6 shows an example of such a case. Initially erroneous classification is also the reason that the car Recall value in Table 10 is lower than the pedestrian Recall value there. In these experiments, many false positives were the result of a care being classified as a pedestrian by the "Extended Trajectory" initializing method.

5.2 Classification and Tracking Experiments in a Crowded Indoor Environment

This section describes experimental results obtained using the proposed method in a more crowded environment. One SICK LMS200 was placed on a booth at an exhibition and the real-time performance of the proposed system was demonstrated. The exhibition lasted three days, and we extracted a subset of the whole data for the evaluation (about 10 minutes at the most crowded time). As the LRF was fixed to the ground, the initialize method was the occupancy grid with a polar coordinate.

The SVM data for estimation of numbers of pedestrians was the same as that in the outdoor experiment described in the preceding section. As this experiment was performed in an indoor environment without cars, the class "Car" was removed.

The average numbers of groups and individual pedestrians were 18.3 and 23.8 respectively. The number of particle per object was 500, and average processing time for one frame was about 53.4 ms (Intel Core i7-940XM). Although the number of particles was much greater than that in the experiment described in the preceding section, the initialization method using a grid has a much smaller computation cost than the "Extended Trajectory" method does. As a result, processing time did not become worse. The LRF scans were obtained every 120 ms, so the proposed method could run in real time.

Fig. 7 shows the scenes of the tracking, and the per-object accuracies are listed in Table 11. As the grid map did not update during the evaluation, many stopping persons were detected. Therefore only the numbers of pedestrians that moved more than 1 m are listed in Table 11. As the LRF was fixed, there were far fewer false positives than there were in the outdoor results. The number of pedestrians "Disrupted in occ." was large, however, because there were several persons were standing near the LRF and they caused large occluded areas.

6 Conclusion

This paper described a method for classifying and tracking multiple moving objects with a laser range finder. The experimental results obtained when we implemented the proposed method using a personal mobility robot demonstrated its effectiveness in the real world. The number of false positives in outdoor environments is a major problem to overcome. We are now implementing initializing test algorithms that use several time-series LRF scans to determine whether or not tracking should start.

A major theoretical shortcoming of this method is that it performs classification and estimation of the number of pedestrians independent of the results of SJPDAFs. In future work, we are planning to integrate classification and tracking using MHT frameworks.

References

1. Bar-Shalom, Y.: Extension of the probabilistic data association filter to multi-target tracking. In: Proc. of the 5th Symposium on Nonlinear Estimation, pp. 16–21 (1974)
2. Reid, D.B.: An algorithm for tracking multiple targets. IEEE Transactions on Automatic Control AC-24(6), 843–854 (1979)
3. Lau, B., Arras, K.O., Burgard, W.: Multi-model hypothesis group tracking and group size estimation. International Journal of Social Robotics 2(1), 19–30 (2010)
4. Ata ur Rehman, Naqvi, S.M., Mihaylovay, L., Chambers, J.A.: Clustering and a joint probabilistic data association filter for dealing with occlusions in multi-target tracking. In: Proc. of 16th International Conference on Information Fusion, FUSION 2013 (2013)
5. Song, X., Cui, J., Zhao, H., Zha, H., Shibasaki, R.: Laser-based tracking of multiple interacting pedestrians via on-line learning, pp. 92–105 (2013)
6. Vermaak, J., Doucet, A.: Maintaining multi-modality through mixture. In: Proc. of 9th IEEE International Conference on Computer Vision (ICCV 2003), pp. 1110–1116 (2003)
7. Kurazume, R., Yamada, H., Murakami, K., Iwashita, Y., Hasegawa, T.: Target tracking using SIR and MCMC particle filters by multiple cameras and laser range finders. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008), pp. 3838–3844 (2008)
8. Schulz, D., Burgard, W., Fox, D., Cremers, A.B.: People tracking with a mobile robot using sample-based joint probabilistic data association filters. International Journal of Robotics Research 22, 99–117 (2003)
9. Dietmayer, K., Sparbert, J., Streller, D.: Model based object classification and object tracking in traffic scenes from range images. In: Proc. of the IEEE Intelligent Vehicle Symposium IV (2001)
10. Zhao, H., Zhan, Q., Chiba, M., Shibasaki, R., Cu, J., Zha, H.: Moving object classification using horizontal laser scan data. In: Proc. of the IEEE International Conference on Robotics and Automation (ICRA 2009), pp. 2424–2430 (2009)
11. Mori, T., Sato, T., Noguchi, H., Shimosaka, M., Fukui, R., Sato, T.: Moving objects detection and classification based on trajectories of LRF scan data on a grid map. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010), pp. 2606–2611 (2010)

GPmap: A Unified Framework for Robotic Mapping Based on Sparse Gaussian Processes

Soohwan Kim and Jonghyuk Kim

Abstract. This paper proposes a unified framework called *GPmap* for reconstructing surface meshes and building continuous occupancy maps using sparse Gaussian processes. Previously, Gaussian processes have been separately applied for surface reconstruction and occupancy mapping with different function definitions. However, by adopting the signed distance function as the latent function and applying the probabilistic least square classification, we solve two different problems in a single framework. Thus, two different map representations can be obtained at a single cast, for instance, an object shape for grasping and an occupancy map for obstacle avoidance. Another contribution of this paper is reduction of computational complexity for scalability. The cubic computational complexity of Gaussian processes is a well-known issue limiting its applications for large-scale data. We address this by applying the sparse covariance function which makes distant data independent and thus divides both training and test data into grid blocks of manageable sizes. In contrast to previous work, the size of grid blocks is determined in a principled way by learning the characteristic length-scale of the sparse covariance function from the training data. We compare theoretical complexity with previous work and demonstrate our method with structured indoor and unstructured outdoor datasets.

1 Introduction

Learning maps from observations is a fundamental problem in robotics. As representations of the world, accurate maps are essential for robots to perform tasks successfully while interacting with the world. For instance, mobile robots navigate through environments using occupancy maps, while manipulators grasp unknown objects based on their shape models. In addition to accuracy, scalability is another important issue in robotic mapping. With the emergence of high definition range sensors such as Velodyne laser scanners and Microsoft Kinects, millions of data

Soohwan Kim · Jonghyuk Kim
The Australian National University, Canberra, Australia
e-mail: {soohwan.kim,jonghyuk.kim}@anu.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_22, © Springer International Publishing Switzerland 2015

points are acquired even from a single snapshot. In large-scale environments such as university campuses and cities, this scalability issue becomes more serious.

In this paper, we use machine learning techniques to generate accurate and scalable robotic maps. Particularly, we combine Gaussian process implicit surfaces (GPIS) [31] and Gaussian process occupancy mapping (GPOM) [19] to build two different kinds of map representations, surface meshes and occupancy maps in a unified framework. This is made possible by applying Gaussian process regression even to GPOM with a signed distance latent function and applying the probabilistic least square classification [21].

In addition, we apply the sparse covariance function [16] to address the cubic computational complexity of Gaussian processes. Most of approximation methods use a subset of training data to reduce the computational complexity [5, 25], but the data selection relies on heuristics. Instead, we partition both training and test data in a principled way by training the characteristic length-scale of the sparse covariance function and discretizing the world into grid blocks with the hyperparameter. Then, the test positions within a block are correlated only with the training data in itself and its neighboring blocks. Therefore, active data selection and data clustering based on heuristics in previous approaches is substituted with training hyperparameters of the sparse covariance function.

Moreover, in order to handle arbitrary and complex shapes, we use derivative observations for training data based on the fact that the derivative of a Gaussian process is still a Gaussian process [28]. Because the gradient of the signed distance function is the surface normal vector, we estimate surface normals from the hit points and the robot positions and use them as derivative observations. As a result, we extract zero-valued iso-surfaces from the signed distance scalar fields using marching cubes [15] and build continuous occupancy maps by applying the probabilistic least square classification. We compare theoretic complexity with previous work and demonstrate our method with structured indoor and unstructured outdoor datasets. The map accuracy and runtime of occupancy maps is compared with octomaps [8].

The rest of the paper is organized as follows. We summarize related work in Section 2 and describe the overview of our method in Section 3. The mathematical foundation of our method, Gaussian processes, is explained in Section 4, while the post processing for surface reconstruction and occupancy mapping follows in Section 5. We demonstrate our method with real datasets and compare with octomaps in Section 6 and conclude the paper with future work in Section 7.

2 Related Work

Occupancy grid maps [17] have been widely used for mobile robot navigation. The world is discretized into disjoint grid cells, and the occupancy of each cell is updated incrementally and independently. Thanks to the independence assumption between observations, it runs fast and produces accurate maps, but requires huge amount of memory to store every single occupancy value of the environment. To address this problem, octomaps [32, 8] apply a memory-efficient data structure, octrees which maintain updated cells only and support multi-resolution map queries. By updating the occupancy maps scan-by-scan rather than ray-by-ray, the map accuracy is further enhanced in octomaps.

As an alternative map representation, elevation maps [4] have also been applied for legged robot locomotion. Because it stores height of every point on the ground, less memory is consumed, compared with occupancy grid maps. However, it is technically a 2.5 dimensional representation and thus suitable for terrain maps, not for arbitrary and complex environments. Recently, Gaussian processes have been applied to elevation maps [30]. Non-stationary covariance functions were adopted to reflect local characteristics [14, 20], and visibility information was exploited to enhance the map accuracy [3].

Meanwhile, Gaussian processes have also been applied to occupancy mapping [19]. Because a Gaussian process prior assumes a joint Gaussian distribution between outputs of a latent function, more accurate and continuous occupancy maps have been obtained compared with conventional occupancy grid maps. Another benefit of this approach is that map uncertainties are also provided, which can be used for exploration and path planning. Since Gaussian processes require discrete data points for training data, rays acquired from range sensors need to be discretized into several knot points, which increases the size of training data dramatically. Integral kernels [18] addressed this problem by integrating output values along the rays.

Interestingly, Gaussian process implicit surfaces [31] have been proposed for surface reconstruction. A signed distance function was adopted as a latent function from which zero-valued iso-surfaces were extracted. In the robotics community, it was applied for estimating unknown object shapes for grasping [2] by combining visual, haptic and laser data and for assessing mesh uncertainties to plan underwater inspection paths [6]. Gaussian process implicit surfaces were also represented with a beam-space parameterization and further used for adaptive compression of 3D laser data [27].

However, the cubic computational complexity of Gaussian processes limits its applications for large-scale environment mapping. Thus, many approximation methods have been proposed to speed up the runtime of Gaussian processes. A *kd*-tree was used to search for nearest training data of a test position, but it requires to build a new covariance matrix for each test position [26, 30, 19]. Based on the divide-and-conquer strategy, training data was clustered into manageable sizes, and Gaussian process experts were applied to each clusters [9, 10, 11]. In order to take the large size of test data into account, both training and test data was partitioned together, and local Gaussian processes were applied [12, 13]. However, those approaches heuristically determine the size of clusters or threshold the correlation between data. In this paper, we apply the sparse covariance function [16] of which hyperparameters are trained from data and partition both training and test data in a principled way.

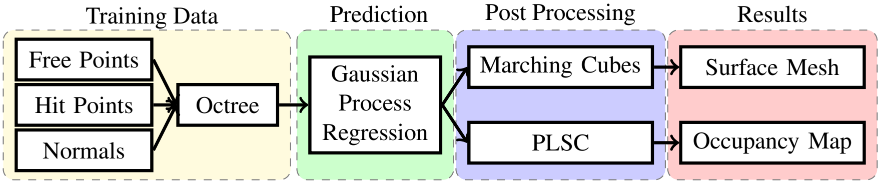


Fig. 1 Flow chart of our unified framework, *GPmap* for surface reconstruction and occupancy mapping using sparse Gaussian processes. PLSC stands for probabilistic least square classification.

3 Overview of Our Unified Approach

Fig. 1 describes the overview of our unified framework called *GPmap* for occupancy mapping and surface reconstruction. As training data, we use hit and free points for function observations and surface normal vectors for derivative observations. For free points, we sample points on the rays just before the hit points, while surface normals are estimated from nearest neighbors of the hit points. In order to address the scalability problem of Gaussian processes, we partition the space with grid blocks and store the training data in an octree. For the test positions of each grid block, we locally apply a Gaussian process regression with the training data in itself and its neighboring grid blocks. Given the means and variations of test positions, we produce surface meshes by extracting zero-valued iso-surfaces using marching cubes and build continuous occupancy maps by applying the probabilistic least square classification. Detailed explanation of each step will be followed in the next sections.

4 Gaussian Process Regression

Let us begin with the mathematical foundation of our method, Gaussian processes. A Gaussian process is a non-parametric Bayesian approach to regression and classification. It is a distribution over functions and assumes a joint Gaussian distribution between function outputs. Particularly, Gaussian process implicit surfaces (GPIS) are designed to predict the signed distance function which maps every point on the surface to zero and other points inside/outside of the surface to positive/negative distances to the surface, respectively.

Suppose that we have point clouds acquired from range sensors. Since sensors are not perfect, we assume that the output y of the signed distance function f at a position $\mathbf{x}$ is corrupted by addictive Gaussian noise ε,

$$y = f(\mathbf{x}) + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma_n^2)\,. \tag{1}$$

Given n noisy observations $(\mathbf{X},\mathbf{y}) = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ and a test data $(\mathbf{x}_*, f_*)$, a Gaussian process assumes a joint Gaussian distribution between outputs,

$$\begin{aligned} \begin{bmatrix} \mathbf{y} \\ f_* \end{bmatrix} &\sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}+\sigma_n^2\mathbf{I} & \mathbf{k}_* \\ \mathbf{k}_*^{\mathrm{T}} & k_{**} \end{bmatrix}\right), \\ \mathbf{K} \in \mathbb{R}^{n\times n},\ & [\mathbf{K}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)\,, \\ \mathbf{k}_* \in \mathbb{R}^{n},\quad & [\mathbf{k}_*]_i = k(\mathbf{x}_i, \mathbf{x}_*)\,, \\ k_{**} \in \mathbb{R},\qquad & k_{**} = k(\mathbf{x}_*, \mathbf{x}_*)\,, \end{aligned} \tag{2}$$

where $k(\mathbf{x},\mathbf{x}')$ denotes the covariance function of outputs between two input positions $\mathbf{x}$ and $\mathbf{x}'$, and a zero mean function is selected. See [23] for details.

Therefore, we can infer the predictive distribution of the test output f_* which is also a Gaussian distribution,

$$\begin{aligned} f_* \mid \mathbf{x}_*, \mathbf{X}, \mathbf{y} &\sim \mathcal{N}\left(\mu_*, \sigma_*^2\right), \\ \mu_* &= \mathbf{k}_*^{\mathrm{T}}\left(\mathbf{K}+\sigma_n^2\mathbf{I}\right)^{-1}\mathbf{y}\,, \\ \sigma_*^2 &= k_{**} - \mathbf{k}_*^{\mathrm{T}}\left(\mathbf{K}+\sigma_n^2\mathbf{I}\right)^{-1}\mathbf{k}_*\,. \end{aligned} \tag{3}$$

Thus, from the estimated signed distances of test positions μ_*, a scalar field in the three dimensional space, we can reconstruct surface meshes by extracting zero-valued iso-surfaces and estimate continuous occupancy maps by squashing the means μ_* with variances σ_*^2 into an S-shaped curve, a cumulative Gaussian density function. Details will be explained in Section 5.

The hyperparameters of covariance functions and σ_n are determined by maximizing the log marginal likelihood,

$$\log p(\mathbf{y}|\mathbf{X},\theta) = -\frac{1}{2}\mathbf{y}^{\mathrm{T}}(\mathbf{K}+\sigma_n^2\mathbf{I})^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}+\sigma_n^2\mathbf{I}| - \frac{n}{2}\log 2\pi\,, \tag{4}$$

where $|\cdot|$ denotes the matrix determinant.

4.1 Sparse Mátern Covariance Function

The covariance function defines the correlation between two data points and thus plays a pivotal role in Gaussian processes. The squared exponential covariance function is a common choice, but it is too smooth to handle complex and arbitrary object shapes. Thus, we employ the Mátern covariance function, instead. However, its correlation does not vanish to zero even though two data points are very far away, which makes the covariance matrix dense in Eq. (2) and inverting it computationally expensive. Therefore, we combine the Mátern covariance function and the sparse covariance function to cover sudden changes in outputs and avoid the high computational complexity. The independency of the sparse covariance function enables us to partition both training and test data with grid blocks and to apply local Gaussian processes to each block separately.

4.1.1 Mátern Covariance Function

We use the Mátern covariance function with $\nu = 3/2$,

$$k_{\text{Mátern}}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \left(1 + \frac{\sqrt{3}r}{l_M}\right) \exp\left(\frac{-\sqrt{3}r}{l_M}\right), \tag{5}$$

where $r = |\mathbf{x} - \mathbf{x}'|$, and the hyperparameters σ_f and $l_M > 0$ are called the signal variance and the characteristic length-scale, respectively. Note that as the distance between two points grows, the correlation decreases quickly. However, even if it is huge enough, the correlation is still non-zero because of the exponential term.

4.1.2 Sparse Covariance Function

The sparse covariance function [16] is designed to make the covariance to be sparse and thus to make the exact inference possible even with large datasets without any approximation. It is defined as

$$k_{\text{Sparse}}(\mathbf{x}, \mathbf{x}') = \begin{cases} \sigma_f^2 \left(\frac{2+cos(2\pi r_S)}{3}(1 - r_S) + \frac{1}{2\pi}\sin(2\pi r_S)\right) & \text{if } r_S < 1 \\ 0 & \text{if } r_S \geq 1, \end{cases} \tag{6}$$

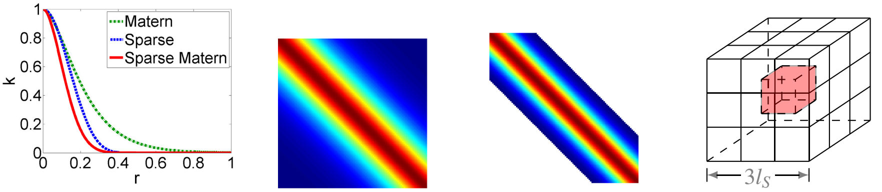
where $r_S = r/l_S$, and the characteristic length-scale $l_S > 0$.

4.1.3 Sparse Mátern Covariance Function

Since a product of two kernels is also a kernel, we multiply them together to take advantages of both covariance functions,

$$k_{\text{SparseMátern}}(\mathbf{x}, \mathbf{x}') = k_{\text{Sparse}}(\mathbf{x}, \mathbf{x}')\, k_{\text{Mátern}}(\mathbf{x}, \mathbf{x}') \,. \tag{7}$$

Fig. 2(a) compares different covariance functions. Recognize that when the distance between two points is greater than l_S, the sparse and the sparse Mátern covariance functions vanish, while the Mátern covariance function still has non-zero values. This is more evident when visualizing the covariance matrices. Notice that all the coefficients of the Mátern covariance matrix in Fig. 2(b) is filled with non-zero values, while the off-diagonal terms of the sparse Mátern covariance matrix in Fig. 2(c) are set to zero, hence it is named the sparse covariance function.



(a) Covariance, $k(r)$ (b) $\mathbf{K}_{\text{Mátern}}$ (c) $\mathbf{K}_{\text{SparseMátern}}$ (d) Data Partioning

Fig. 2 The effects of the sparse covariance function. (1) Comparison between several covariance functions ($\sigma_f = 1$, $l_S = 0.5$, and $l_M = 0.2$), (2) and (3) Covariance matrices of the Mátern and sparse Mátern covariance functions with linearly spaced points between 0 and 1 (red: high, blue: small, and white: zero), and (4) Data partitioning with grid blocks. The data in the red cube is independent of others outside of the extended cube.

4.1.4 Data Partitioning with Grid Blocks

The hyperparameter l_S in the sparse covariance function also naturally controls the data partitioning. Suppose that the space is divided into grid blocks of size l_S as shown in Fig. 2(d). Then, the distances between test positions in the red cube and any training data out of the extended $3l_S \times 3l_S \times 3l_S$ cube are greater than l_S. In other words, all the test points of the red cube in the center are totally independent of the training data outside of the extended cube. Therefore, we apply local Gaussian processes to the test positions in each grid block with the training data in itself and its surrounding blocks. Note that because there exists correlation between the training data inside and outside of the extended cube, this is not an exact inference but a reasonable approximation of Gaussian processes.

To be more precise, we partition the training data with an octree of a resolution l_S and infer the predictive distribution of the test positions of each grid block, only if there exist training data in the extended cube. This seems similar to the Gaussian process approximation with a *kd*-tree [26, 30, 19] in that the training data is clustered. However, our method predicts the posteriors for blocked test positions, not for every single test position. In addition, our method also applies local Gaussian processes as in [12, 13]. However, the size of the grid block is determined in a principled way by learning the hyperparameter l_S from the training data to maximize the marginal likelihood in Eq. (4).

Now, let us focus on how much computational complexity is reduced with this data partitioning. According to Eq. (3), for predicting the mean and variance of a test position given n training data, we need to invert the $n \times n$ matrix, which costs $O(n^3)$, and to multiply the $n \times n$ inverted matrix with the $n \times 1$ vector, which costs another $O(n^2)$. Therefore, for m test positions, the total computational complexity is $O(n^3 + n^2 m)$.

Suppose that training and test data is evenly partitioned with k grid blocks. Then, the number of training data would drop to $9n/k$ in the extended cube, and the number of test data in the red cube is m/k. Thus, the computational complexity decreases significantly to $O\left(n^3/k^2 + n^2 m/k^2\right)$. Note that predictions are repeated k times by

Table 1 Comparison of computational complexity between various approximation methods for Gaussian processes. ([methods] GP: Gaussian process, MGP: mixture of Gaussian processes, and LGP (our method): local Gaussian processes, [parameters] n: number of training data, m: number of test positions, and k: number of equally-divided grid blocks.) Note that $n \approx m$ in large-scale environments.

GP	MGP	LGP (our method)
$O(n^3 + n^2 m)$	$O\left(\frac{n^3}{k^2} + \frac{n^2 m}{k}\right)$	$O\left(\frac{n^3 + n^2 m}{k^2}\right)$

individual Gaussian processes in each grid block. This is even faster than the previous work [9, 11] which clusters training data only and applies a mixture of Gaussian processes. In that case, the predictions are repeated k times over the whole m test positions by k experts with n/k training data per cluster, and the results are merged into one by a weighted sum. Thus, the computation complexity is $O\left(n^3/k^2 + n^2 m/k\right)$. Table 1 summarizes the computational complexities of approximations for Gaussian processes.

4.2 Derivative Observations

Since a Gaussian process is a linear estimator, the derivative of a Gaussian process is still a Gaussian process. Thus, the covariances between function values and partial derivatives, and between partial derivatives are well-defined as

$$\operatorname{cov}\left(f, \frac{\partial f'}{\partial x_j}\right) = \frac{\partial k(\mathbf{x}, \mathbf{x}')}{\partial x_j}, \quad \operatorname{cov}\left(\frac{\partial f}{\partial x_i}, \frac{\partial f'}{\partial x_j}\right) = \frac{\partial^2 k(\mathbf{x}, \mathbf{x}')}{\partial x_i \partial x_j}. \tag{8}$$

For details, refer to [28]. Note that by the definition of the signed distance function, its partial derivative is the surface normal vector. Therefore, for each hit point, we estimate the surface normal from nearest neighbors and flip to the robot positions. However, the estimated surface normals are generally very noisy. Therefore, we assume a different level of measurement noise σ_{dn} for the derivative observations,

$$\frac{\partial y}{\partial x_i} = \frac{\partial f(\mathbf{x})}{\partial x_i} + \varepsilon_{dn}, \qquad \varepsilon_{dn} \sim \mathcal{N}(0, \sigma_{dn}^2). \tag{9}$$

5 Marching Cubes and Probabilistic Least Square Classification

In this section, we describe the post processing steps, marching cubes and probabilistic least square classification in Fig. 1.

5.1 Marching Cubes

Reconstructing surfaces from points clouds is a well-known problem in computer graphics. Among various approaches, the marching cubes [15] is an algorithm to extract iso-surfaces from a scalar field with a target value. For each cube, it determines intersecting vertices of the target value on each edge by interpolating two values of the end points. In the previous section, we predicted the means of the signed distances at each grid test position. Therefore, we can reconstruct surfaces by extracting iso-surfaces from the means using marching cubes.

5.2 Probabilistic Least Square Classification

Now, we turn our focus to building occupancy maps. Since occupancy mapping is a binary problem, applying Gaussian process classification might look more reasonable. However, because the output values should be class labels, we can not define a continuous latent function directly or use derivative observations. Therefore, in order to combine surface reconstruction and occupancy mapping in a single framework, we take a two-step approach using Gaussian process regression and probabilistic least square classification as shown in Fig. 1. Because the predictive distribution is a Gaussian, with a choice of a probit likelihood we can obtain class probabilities by squashing the means μ_* with variances σ_*^2 in Eq. (3) through a cumulative Gaussian density function $\varPhi$,

$$p(o_* = 1 \mid \mathbf{x}_*) = \varPhi\left(\frac{\alpha\mu_* + \beta}{\sqrt{1+\alpha^2\sigma_*^2}}\right), \tag{10}$$

where the binary random variable $o_* = 1$ (occupied) or 0 (empty), and the parameters α and β are optimized by performing leave-one-out cross-validation on the training set. Refer to [21] for details.

6 Experimental Results

Table 2 summarizes the benchmark datasets to be used for demonstration. Both structured indoor and unstructured outdoor environments are included. Our method was from 1.5 to 17 times slower than octomap. Recognize that the running time ratio of our method to octomap is decreasing in general as the size of the environment grows. This is because octomap updates free cells as well as occupied cells, while our method only considers grid blocks which have training data. Thus, one of our disadvantages is that our method is not suitable for dynamic environments. Instead, our method provides continuous occupancy maps with map uncertainties which can be used for exploration and path planning. We trained the hyperparameters of the sparse Mátern covariance function and noise variances of function values and partial derivatives with sampled datasets. ($\sigma_f = 3.6847 \times 10^{-3}$, $l_S = 3.4349 \times 10^{-1}$, $l_M = 1.8907 \times 10^{-1}$, $\sigma_n = 2.2063 \times 10^{-3}$, and $\sigma_{dn} = 5.2856 \times 10^{-1}$)

Table 2 Datasets used for demonstration and corresponding running time of octomap and our method (resolution = $0.1\,m$ and ratio = ours/octomap)

Dataset	Size (m^3)	Scans	Points/Scan	Ours	Octomap	Ratio
FR_079[1]	$44 \times 18 \times 4$	66	89,446	7.7 min	27.8 sec	16.6
Winter[2]	$72 \times 70 \times 19$	31	138,205	10.1 min	38.1 sec	15.9
Hauptgebaude[2]	$62 \times 65 \times 18$	36	189,163	9.5 min	98.6 sec	5.8
Gauss[3]	$204 \times 237 \times 44$	27	854,404	31.9 min	22.1 min	1.4

[1] The octomap 3D scan datasets of University of Freiburg [7]

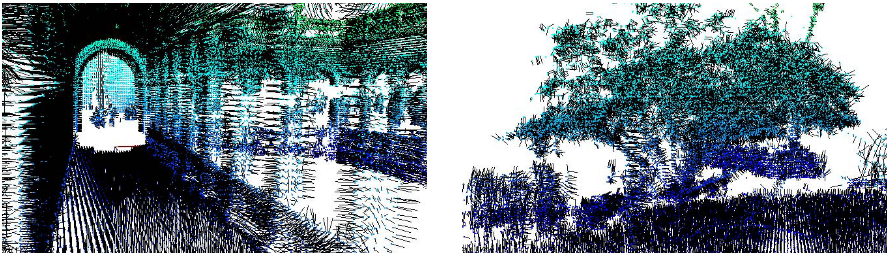
[2] ETHZ ASL datasets [22]

[3] The robotic 3D scan repository of Jacobs University [1]

6.1 Estimated Surface Normals

We first subsampled the hit points using a voxel grid filter [24] for fast surface normal estimation. The point clouds were partitioned with octrees of size $0.05\,m$ and approximated with their centroids. This approach is known to compress the size of point clouds with less loss of the details of the structure. Then, we stored the subsampled point clouds into *kd*-trees and estimated normal vectors with the eigenvectors associated with the smallest eigenvalues of the covariance matrices created from the nearest neighbors within a radius of $0.1\,m$. Finally, to make sure that surface normals are oriented towards the robot positions, we flipped them if they made obtuse angles with corresponding rays. The surface normal vectors were subsampled again with octrees of size $0.1m$ to remove redundant training data. The free points were sampled on the rays just $0.1\,m$ before the hit points.

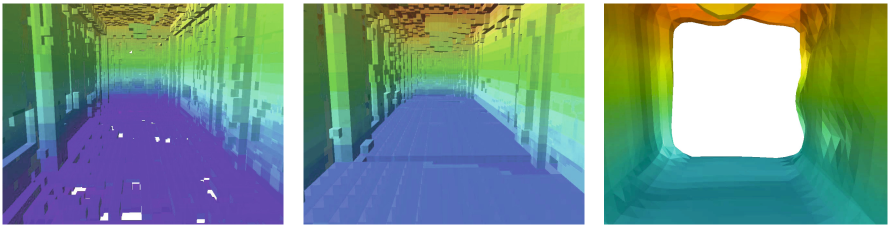
Fig. 3 shows exemplary results of our method. In the structured environment the normal vectors are estimated regularly towards inside of the cloister. On the other hand, the estimated normal vectors on the tree leaves are very noisy and pointing in almost every direction. This is why we introduced another noise variance for derivative observations, σ_{dn} in Eq. (9). As expected, σ_{dn} was trained much higher than the noise variance of function values σ_n.



(a) Structured environment (cloister)

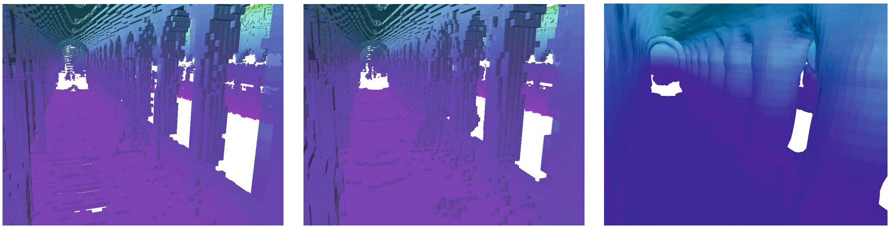
(b) Unstructured environment (tree)

Fig. 3 Estimated surface normal vectors



(a) Octomap (b) Occupancy Map (c) Surface Mesh

Fig. 4 Corridor (dataset: FR_079 [29])



(a) Octomap (b) Occupancy Map (c) Surface Mesh

Fig. 5 Cloister (dataset: Hauptgebaude [22])

6.2 Occupancy Maps and Surface Meshes

Fig. 4 and 5 depict octomaps and our results of a corridor and a cloister, respectively. In both results, octomaps reflect the structures very well but have some errors such as holes on the ground and noise on the walls. Our method, on the other hand, generated more accurate and consistent occupancy maps. Surface meshes are also well reconstructed, but they are a little bit smoothed. For example, the arches at the cloister look thicker than the reality. This is because of noise in observations. Octomaps and our continuous occupancy maps were thresholded at the probability of 0.5. Particularly, we removed uncertain occupied cells with variances greater than a threshold of 3.0×10^{-6}.

We also demonstrated our method with unstructured outdoor environments. Fig. 6 and 7 show occupancy maps and reconstructed surfaces of a forest and a city, respectively. Again, our method produced more accurate occupancy maps compared with octomaps as well as surface meshes at the same time. Notice that our occupancy maps have less holes on the ground and on the building walls. Also, the sharp objects such as the electric cables in the air of the city are represented more clearly in our occupancy map and surface mesh.

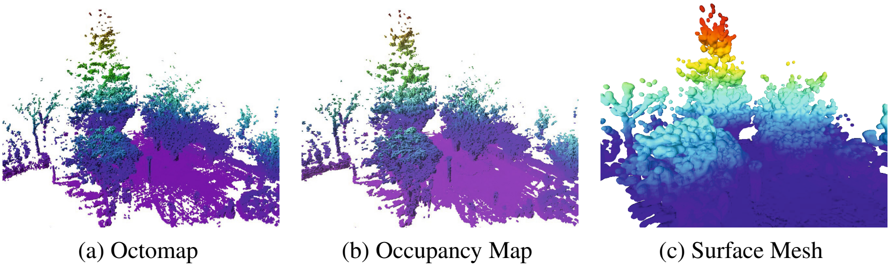

(a) Octomap (b) Occupancy Map (c) Surface Mesh

Fig. 6 Forest (dataset: Winter [22])

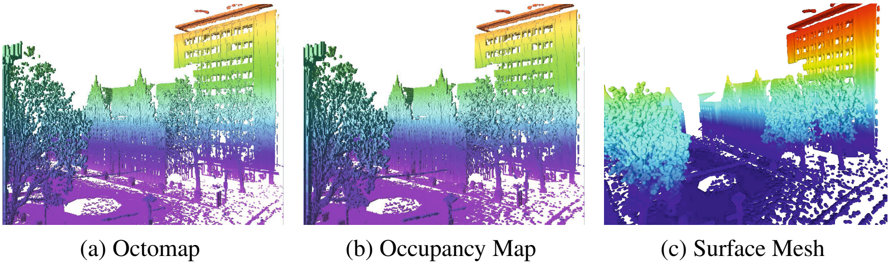

(a) Octomap (b) Occupancy Map (c) Surface Mesh

Fig. 7 City (dataset: Gauss [1])

7 Conclusions

In this paper, we proposed a unified framework for surface reconstruction and occupancy mapping using sparse Gaussian processes, called *GPmap*. Therefore, two different map representations, surface meshes and continuous occupancy maps were obtained at a single cast. Free and hit points, and estimated surface normals were used as training data to predict signed distances of test positions. From predicted means and variances, we extracted iso-surfaces using marching cubes and generated continuous occupancy maps using the the probabilistic least square classification.

Another contribution of this paper is our approximation method to reduce the computational complexity of Gaussian processes by applying the sparse covariance function. We partitioned both training and test data with blocks of which size is determined by learning the characteristic length-scale of the sparse covariance function from training data. We demonstrated our method with indoor and outdoor real datasets. Compared with octomaps, our method took more time for prediction, but provided more accurate occupancy maps as well as surface meshes.

The limitation of our method is that it is designed for static environments and same hyperparameters are used across the whole input space. In other words, the algorithm runs in batch processing. Our future work will include adapting hyperparameters to local training data and updating the maps incrementally for dynamic environments for online processing.

Acknowledgements. The authors would like to thank Tim Barfoot at University of Toronto and James Underwood at University of Sydney for their valuable comments.

References

1. Borrmann, D., Nüchter, A.: The robotic 3d scan repository of Jacobs University, `http://kos.informatik.uni-osnabrueck.de/3Dscans/`
2. Dragiev, S., Toussaint, M., Gienger, M.: Gaussian process implicit surfaces for shape estimation and grasping. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2845–2850 (2011)
3. Hadsell, R., Bagnell, J., Huber, D., Hebert, M.: Space-carving kernels for accurate rough terrain estimation. The International Journal of Robotics Research 29(8), 981–996 (2010)
4. Herbert, M., Caillas, C., Krotkov, E., Kweon, I.S., Kanade, T.: Terrain mapping for a roving planetary explorer. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 997–1002 (1989)
5. Herbrich, R., Lawrence, N.D., Seeger, M.: Fast sparse Gaussian process methods: The informative vector machine. In: Advances in Neural Information Processing Systems 14, pp. 609–616. MIT Press (2002)
6. Hollinger, G., Englot, B., Hover, F., Mitra, U., Sukhatme, G.: Active planning for underwater inspection and the benefit of adaptivity. The International Journal of Robotics Research 32(1), 3–18 (2013)
7. Hornung, A.: Octomap 3d scan dataset, `http://ais.informatik.uni-freiburg.de/projects/datasets/octomap/`
8. Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W.: Octomap: an efficient probabilistic 3d mapping framework based on octrees. In: Autonomous Robots, pp. 1–18 (2013)
9. Kim, S., Kim, J.: Towards large-scale occupancy map building using Dirichlet and Gaussian processes. In: Proceedings of the Australasian Conference on Robotics and Automation (2011)
10. Kim, S., Kim, J.: Building large-scale occupancy maps using an infinite mixture of Gaussian process experts. In: Proceedings of the Australasian Conference on Robotics and Automation (2012)
11. Kim, S., Kim, J.: Building occupancy maps with a mixture of Gaussian processes. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 4756–4761 (2012)
12. Kim, S., Kim, J.: Continuous occupancy maps using overlapping local gaussian processes. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (2013)
13. Kim, S., Kim, J.: Occupancy mapping and surface reconstruction using local gaussian processes with kinect sensors. IEEE Transactions on Cybernetics 43(5), 1335–1346 (2013)
14. Lang, T., Plagemann, C., Burgard, W.: Adaptive non-stationary kernel regression for terrain modeling. In: Proceedings of Robotics: Science and Systems (2007)
15. Lorensen, W., Cline, H.: Marching cubes: A high resolution 3D surface construction algorithm. ACM SIGGRAPH Computer Graphics 21, 163–169 (1987)
16. Melkumyan, A., Ramos, F.: A sparse covariance function for exact Gaussian process inference in large datasets. In: Proceedings of International Joint Conference on Artificial Intelligence, pp. 1936–1942 (2009)

17. Moravec, H., Elfes, A.: High resolution maps from wide angle sonar. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 2, pp. 116–121 (1985)
18. O'Callaghan, S., Ramos, F.: Continuous occupancy mapping with integral kernels. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 1494–1500 (2011)
19. O'Callaghan, S., Ramos, F.: Gaussian process occupancy maps. The International Journal of Robotics Research 31(1), 42–62 (2012)
20. Plagemann, C., Mischke, S., Prentice, S., Kersting, K., Roy, N., Burgard, W.: Learning predictive terrain models for legged robot locomotion. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3545–3552 (2008)
21. Platt, J.C.: Probabilities for SV Machines. In: Advances in Large Margin Classifiers, pp. 61–74. MIT Press (2000)
22. Pomerleau, F., Liu, M., Colas, F., Siegwart, R.: Challenging data sets for point cloud registration algorithms. The International Journal of Robotics Research 31(14), 1705–1711 (2012)
23. Rasmussen, C., Williams, C.: Gaussian Processes for Machine Learning. MIT Press (2006)
24. Rusu, R.B., Cousins, S.: 3d is here: Point cloud library (pcl). In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1–4 (2011)
25. Seeger, M., Williams, C.K., Lawrence, N.D.: Fast forward selection to speed up sparse Gaussian process regression. In: Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics (2003)
26. Shen, Y., Ng, A., Seeger, M.: Fast Gaussian process regression using *kd*-trees. In: Advances in Neural Information Processing Systems 18, pp. 1225–1232. MIT Press (2006)
27. Smith, M., Posner, I., Newman, P.: Adaptive compression for 3d laser data. The International Journal of Robotics Research 30(7), 914–935 (2011)
28. Solak, E., Murray-Smith, R., Leithead, W.E., Leith, D.J., Rasmussen, C.E.: Derivative observations in Gaussian process models of dynamic systems. In: Advances in Neural Information Processing Systems 13, pp. 1057–1064. MIT Press (2003)
29. Steder, B., Kümmerle, R.: The outdoor dataset of the University of Freiburg, `http://ais.informatik.uni-freiburg.de/projects/datasets/fr360/`
30. Vasudevan, S., Ramos, F., Nettleton, E., Durrant-Whyte, H.: Gaussian process modeling of large-scale terrain. Journal of Field Robotics 26(10), 812–840 (2009)
31. Williams, O., Fitzgibbon, A.: Gaussian process implicit surfaces. In: Proceedings of the Workshop on Gaussian Processes in Practice (2006)
32. Wurm, K.M., Hornung, A., Bennewitz, M., Stachniss, C., Burgard, W.: Octomap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In: Proceedings of the ICRA Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation (2010)

Part VII
Vision

Purposive Sample Consensus: A Paradigm for Model Fitting with Application to Visual Odometry

Jianguo Wang and Xiang Luo

Abstract. ANSAC (random sample consensus) is a robust algorithm for model fitting and outliers' removal, however, it is neither efficient nor reliable enough to meet the requirement of many applications where time and precision is critical. Various algorithms have been developed to improve its performance for model fitting.

A new algorithm named PURSAC (purposive sample consensus) is introduced in this paper, which has three major steps to address the limitations of RANSAC and its variants. Firstly, instead of assuming all the samples have a same probability to be inliers, PURSAC seeks their differences and purposively selects sample sets. Secondly, as sampling noise always exists; the selection is also according to the sensitivity analysis of a model against the noise. The final step is to apply a local optimization for further improving its model fitting performance. Tests show that PURSAC can achieve very high model fitting certainty with a small number of iterations.

Two cases are investigated for PURSAC implementation. It is applied to line fitting to explain its principles, and then to feature based visual odometry, which requires efficient, robust and precise model fitting. Experimental results demonstrate that PURSAC improves the accuracy and efficiency of fundamental matrix estimation dramatically, resulting in a precise and fast visual odometry.

Keywords: robust model fitting, visual odometry, samples' reliability, samples' geometry, sampling noise.

1 Introduction

Introduced in 1981, RANSAC is a popular algorithm for a variety of robust model fitting problems - particularly in computer vision for recovering epipolar geometry

Jianguo Wang · Xiang Luo
University of Technology Sydney (UTS), Sydney, Australia
e-mail: xiang.luo@student.uts.edu.au, jianguo.wang@uts.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_23, © Springer International Publishing Switzerland 2015

and 3D motion estimation [2]. It estimates a model that fits the provided data, while simultaneously classifies the data into inliers (samples consistent with the relation) and outliers (samples not consistent with the relation). It is a simple yet powerful technique that can estimate a model using data contaminated by a large fraction of outliers. RANSAC can be briefly summarized as a hypothesize-and-verify framework: a minimal subset of samples for model fitting is randomly selected from the entire dataset. The subset is then used to fit model hypotheses which are evaluated on the entire dataset by computing the distance of all other samples to this model and constructing an inliers' set with a threshold. This hypothesize-and-verify loop is repeated until the probability of finding a model with better consensus than the current best model falls below a predefined threshold. Then all the inliers are used for model parameter estimation [1],[4].

For generality and simplicity, RANSAC is based on a set of assumptions which are not true in many real situations. This leaves large room for improvement. In this paper we analyse potentials for improvement, and propose a purposive sampling algorithm named PURSAC (purposive sample consensus) to substitute random sampling. Comparing to RANSAC and MLESAC (maximum likelihood estimation sample consensus), PURSAC can detect more inliers with much fewer number of iterations and in turn can improve both the efficiency and reliability of model fitting. This is very important for applications where speed and precision is critical, such as visual odometry (VO). With analysis about a model's sensitivity against sampling noise (MSASN) and the pattern of samples' validity, PURSAC is designed to efficiently handle both sampling noise and outliers for model fitting.

The rest of the paper is organized as follows: Section II gives a detailed review of RANSAC and various approaches for its improvement at different aspects, and then introduces the principles of the proposed PURSAC. Section III describes a scheme for line fitting to justify the methodology of PURSAC. Section IV explains the details of applying PURSAC to VO, especially a method for purposive sample subsets selection considering both features' matching score and their geometry. The experimental results of VO on several scenarios are presented in Section V, which demonstrate the effectiveness and robustness of the proposed algorithm. Finally, the conclusion and discussions appear in the last section.

2 RANSAC and Its Variants

RANSAC can often find a correct solution even for seriously contaminated data; however, in order to achieve a high confidence level the required number of sample subsets increases exponentially, and associated computational cost is substantial. Many algorithms have been developed for increasing the efficiency of the basic RANSAC algorithm, some aiming to optimize the process of model verification while some others seeking to preferentially generate more useful hypotheses [3-7]. A comparative analysis of the state-of-the-art RANSAC algorithms and their categorization can be found in [1] and [4]. Here we review them from the aspects they targeted for improvement.

2.1 RANSAC

Assuming all the samples have same *outlier possibility* **ε**, and ignoring the impact of sampling noise, RANSAC follows a random sampling paradigm. Fundamentally it is a stochastic algorithm without deterministic guarantees of finding the global maximum of the likelihood. A *success rate* ***p*** is the level of confidence of finding a consensus subset, which is a function of **ε,** *the number of iterations to be conducted* ***N*** and *the number of samples in a subset* ***s*** [2].

$$N = \frac{\log(1-p)}{\log(1-(1-\epsilon)^s)} \tag{1}$$

For the sake of robustness, in many practical implementations ***N*** is usually multiplied by a factor of ten, which increases much computational costs [1]. Without prior knowledge of **ε,** commonly the implementations of RANSAC estimate **ε** adaptively, iteration after iteration.

In practice, sampling always has noise and ε may be different for each sample. By analysing the difference of ε, it has large potential for optimizing sample subsets selection and improving model fitting performance. As an example, assuming a required success rate ***p*** is 99% and a dataset with outlier rate **ε** = 50%, according to (1), the number of iterations ***N*** is 16 for ***s*** = 2 (line fitting), 145 and 587 for ***s*** = 5 and 7 (visual odometry). If a special part of the dataset can be found having a lower outlier rate ε = 20%, and sample subsets are selected only from this part, then N is just 5, 12 and 20 for s = 2, 5 and 7 respectively. This leads to one of the strategies in PURSAC, which will be detailed in the line fitting example.

The following two subsections review the model fitting approaches applied in computer vision utilizing the difference of samples' **ε** and model's sensitivity against sampling noise (MSASN) respectively.

2.2 Samples Reliability Analysis

There are many cases that sampling process can provide some information about the samples' reliability and accuracy. In feature based image matching, the feature detection and matching stages produce a set of matches where each feature is only matched once, but some of the matches may be in error [21]. In addition to pair the features in two images, feature matching also gives similarity measures (matching scores). It has been found that matches with higher scores have higher reliability of being inliers [9],[21]. The scores or the ranking of the scores provide useful information for selecting reliable sample subsets and then improving the efficiency of model parameter estimation, which RANSAC does not utilize.

Several sampling consensus algorithms have been proposed considering this important information from features' matching. Pre-emptive RANSAC by Nistér is powered by random sample consensus with pre-emptive scoring of the motion hypotheses and enable real-time ego-motion estimation [3]. PROSAC (Progressive Sample Consensus) developed by Chum and Matas tentatively and progressively selects samples from a set of higher ranked features to reduce the computational costs [9]. Uncertainty RANSAC [13] incorporates feature uncertainty and shows that this determines a decrease in the number of potential outliers, thus enforcing a reduction in the number of iterations. A deterministic RANSAC approach [14] also estimates the probability of a match to be correct. Tordoff and Murray [21] proposed guided sampling and consensus for motion estimation based on MLESAC. While MLESAC assumes a uniform prior for the validity of a match, the guided-sampling approach uses the quality function of a feature matching algorithm to derive the probability of matches' validity. Similar to PROSAC, guided sampling and consensus is also based on the evidence that a valid match is likely to have a higher matching score.

Above approaches often achieve significant computational savings in practice, since good hypotheses tend to be generated early on during the sampling process. However, it is observed in many cases, features with high matching scores often lie on a same spatial structure, such as a rich texture section or object in an image, and are potentially in a degenerate geometric configuration [4]. Thus, utilizing above approaches alone has the danger of selecting feature close to each other; therefor other strategies are needed to avoid the degenerate geometric configurations.

2.3 Sampling Noise Analysis

For most model fitting tasks, two types of sampling errors must be considered: small errors (noise) and blunders (outlier) [15]. Even if a consensus subset is found after $\boldsymbol{N}$ iterations, due to the sampling noise and degenerate configurations, the model and inliers decided by this subset may be largely wrong. The reason will be explained with a line fitting example in Fig.1.

There are several algorithms have been proposed to address sampling noise with different strategies. GroupSAC [6] proposed by Kai aims to handle sample subset degenerate configuration. It performs well in the cases of high outlier ratios by image segmentation for group sampling; however, it is inefficient as increasing the computational cost. Assuming inlier samples with Gaussian noise, MLESAC adopts the same sampling strategy as RANSAC to generate putative solutions, but chooses the solution to maximize the likelihood rather than the number of inliers [8]. MLESAC is a generalization of RANSAC. Capel [10] proposed a statistical bail-out test for RANSAC that permits the scoring process be terminated early and achieve computational savings. Tordoff and Murray [21] also mentioned spatial grouping of samples as a cue for further speed the search.

Chum et al. define a locally optimized RANSAC variant to deal with sampling noise and to purposively select sample subset [22]. Observing that a good model tends to find a significant fraction of the inliers, an inner RANSAC strategy is devised using only the set of inliers to the current best model. The inner RANSAC technique has the effect of improving sample consensus more rapidly than standard RANSAC, which causes the termination criterion (1) to be met earlier.

Above methods can make improvement from RANSAC in some aspect, but the improvement is usually limited and unstable. Further improvement can be achieved by strategically fusing a group of selected strategies together.

2.4 PURSAC

By analysing the principle and effectiveness of various approaches, we design a new algorithm PURSAC, aiming to induce better model fitting results with a smaller number of iterations. It takes three major steps to address the limitations of RANSAC and it variants.

a) Instead of assuming all the samples have same probability to be inliers, PURSAC seeks their difference and purposively selects sample sets.
b) As sampling noise always exists, PURSAC purposely selects the subsets according to the analysis of a model's sensitivity against the sampling noise, causing a selective geometric consideration in VO and line fitting.
c) The final step is to apply local optimization algorithm iteratively with all the inliers so as to further improve model fitting performance.

PURSAC can achieve results close to optimal theoretical estimation. Being a qualitative guidance in theory, PURSAC's implementation needs a quantitative analysis to design executable rules for purposive sample consensus. Two examples will be investigated. Line fitting is used as an example to describe the scheme of PURSAC and to justify its methodology. Then it is applied to feature based VO, to conform to its prominent requirement of efficient, robust and precise model fitting.

3 PURSAC for Line Fitting

Let us investigate the line fitting example in the original RANSAC paper [2]. One of the assumptions inherent in the standard termination criterion (1) is that a model computed from an uncontaminated sample subset is consistent with all inliers. In practice, this is often not the case, particularly when the data points are noisy. As showing in Fig.1, two types of sampling errors (noise and outlier) exist within the sample points. Due to sampling noise, model hypotheses selected by RANSAC with limited number of iterations usually do not fit a model well, as Line 1 in the figure. By randomly selecting a set of samples (two points for line fitting) from all the samples without considering MSASN, RANSAC likely misses some inliers and consequentially reduces the accuracy of model fitting. Original RANSAC is only effective in removing outliers but is inadequate of handling sampling noise.

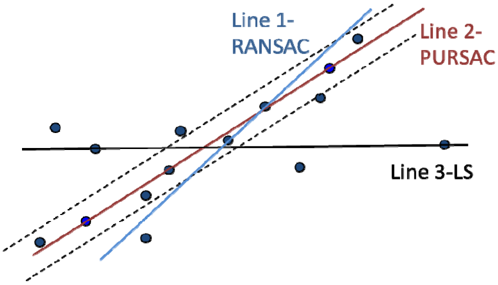


Fig. 1 Line fitting results by RANSAC (Line 1) , PURSAC (Line 2) and least square (Line 3)

Observation1: Samples Geometry

Two-point form of a linear equation is expressed as (2), where ($x_{1,}$ y_1) and ($x_{2,}$ y_2) are the coordinates of the two points that decide the line model.

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) \tag{2}$$

If the noise terms of the two points position are denoted as (δx_1, δy_1) and (δx_2, δy_2), and let $y_2 - y_1$ written as dy, $x_2 - x_1$ as dx, $\delta y_2 - \delta y_1$ as δy and $\delta x_2 - \delta x_1$ as δx, Then the slope of the line can be expressed as

$$\begin{aligned}(y_2 - y_1)/(x_2 - x_1) &= (y_2 - y_1 + \delta y_2 - \delta y_1)/(x_2 - x_1 + \delta x_2 - \delta x_1)\\ &= (\mathrm{d}\,y + \delta y)/(\mathrm{d}\,x + \delta x)\\ &= \frac{\mathrm{d}\,y}{\mathrm{d}\,x} \cdot \frac{1 + \delta y/\mathrm{d}\,y}{1 + \delta x/\mathrm{d}\,x}\end{aligned} \tag{3}$$

The items δx and δy are solely decided by sampling noise, while dx and dy are directly related to the distance of the two points. It can be concluded from (3) that the smaller the distance of two points (dx, dy) is, the more the estimated line slope to be affected by sampling noise (δ x, δ y). This can be evidenced by the results of a Monte Carlo line fitting test.

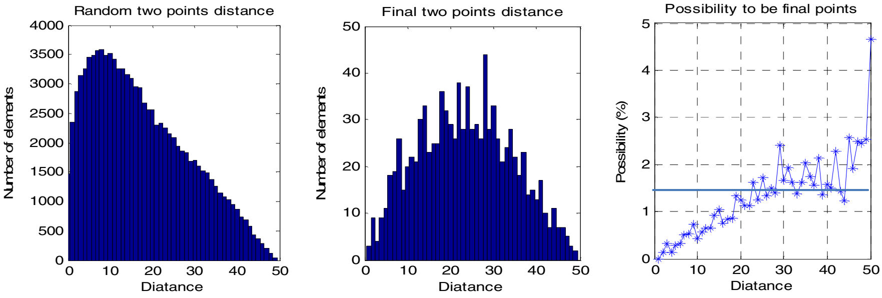


Fig. 2 Monte Carlo test of the distsnce distribution for line fitting with RANSAC

Fig.2 shows the results of 1,000 runs' line fitting tests to verify the possibility distribution of two final points with the best model fitting against their distance. The two points are randomly selected from 100 points, so the average possibility of a point to be selected as the second point is about 1%. However, as shown in the right figure, the possibility of two final points with the best model fitting increases dramatically with their distance, from almost 0% for two close points to about 3% for two far apart points. Therefore it is wise to purposely select two sample points apart, instead of random selection, so as to dilute the effect of sampling noise for model fitting.

The left figure shows that the distance between two random selected points tends to be close; while the middle figure indicates that the distance of the final two points selected by RANSAC has a Gaussian distribution with the peak at middle. The difference of these two distributions causes the result in right figure.

Observation2: Samples Validity

The sample points themselves do not have any information available about their validity for line fitting. It is impossible to rank or score the points without any prior knowledge about them. However, we do know for a line model some of the points are inliers and other are outliers, and their validity will be assessed during the process of classifying them. This validity information can be used to purposively select points afterward that have higher possibility to be inliers so as to speed up the model fitting process.

Similar to locally optimized RANSAC [22], by observing that a good model tends to find a significant fraction of the inliers, a strategic inner iteration is used. After a very small number of iterations ***k***, the inliers of the current best model are generated with RANSAC or MLESAC. Then next sample subset is selected only form the current inliers but verified against the entire dataset. As the sampling is running on only inliers, the size of sample subset can be all the inliers. This can mitigate sampling noise and minimize the error of model fitting.

This optimization technique has the effect of improving the consensus score more rapidly and causes the iteration termination criterion (***N*** in (1)) to be met earlier by selecting samples from current inliers set, which has a lower outlier rate than the entire dataset has. In addition it can also provide more robust and precise model fitting by minimizing the error of model estimation with proper sample size.

Line Fitting with PURSAC

Considering the two observations about samples geometry and validity for the line fitting tests, PURSAC is designed to purposely select two points far apart, instead of random selection. The first point is randomly selected; then the distances from it to all the other points are calculated. The second point is selected according to the statistical distribution shown in Fig.2. Inner iteration is then applied by selecting samples only from the current inliers until reaching iteration termination criterion (1). Finally local optimization is implemented and all the inliers are used iteratively to compute the final model parameters.

Table 1 shows the results of 1,000 runs line fitting tests with RANSAC and PURSAC. Within 100 points, 55 are inliers. The number of iteration N is set to 20, which means the success rate p in (1) is 99.97%. ***Fitting_error*** is a model fitting error against the ground truth measured by the area between the two lines. ***Inliers_miss*** is the number of points that should be counted as inliers but miscounted as outliers; similarly is ***Outliers_miss***. ***STD*** and ***mean*** is the standard deviation and mean of the 1,000 runs' results.

Table 1 Line Fitting Monte Carlo Test Results

Line fitting Method	***Number of inliers*** *mean/STD*	***Fitting_error*** *mean/STD*	***Inliers_miss*** *mean/STD*	***Outliers_miss*** *mean/STD*
PURSAC	46.07 / 0.92	117.58 / 7.22	11.69 / 1.80	1.76 / 1.46
RANSAC	43.67 / 3.41	122.99 / 22.23	14.53 / 4.19	2.20 / 1.69

The result shows that under exactly the same condition, PURSAC can achieve better performance than RANSAC, with less miscounted inliers and outliers, and is closer to true model. The final line fitting performance is affected by the miscounted inliers and outliers. As shown in Table 1 and Fig.3, all the STDs of PURSAC are smaller than that of RANSAC, indicating that PURSAC has better reliability.

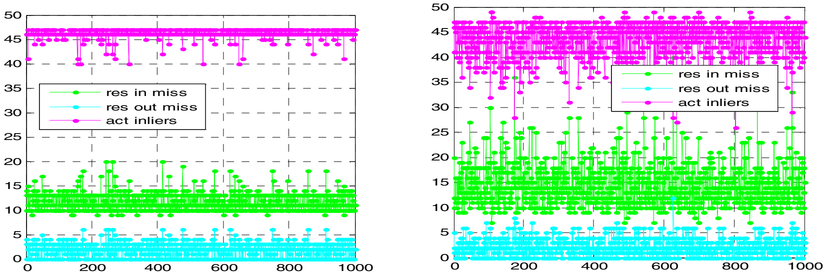


Fig. 3 1000 runs Monte Carlo test results with PURSAC and RANSAC

The key idea behind PURSAC is to purposely select sample sets according to the sensitivity analysis of a model to be fitted against sampling noise; and also to the measures of samples' possibility to be inliers. It is worth to mention that the way to implement PURSAC is based on the analysis of each model fitting case and it is open to find an optimal way for different cases.

4 PURSAC for Visual Odometry

RANSAC and MLESAC has been widely used in computer vision, particularly in the areas of recovering epipolar geometry and 3D motion estimation, such as image registration, structure from motion and feature based VO. The motion of an agent (vehicle, human or robot etc.) can be estimated by incrementally estimating the pose of the agent through examination of the movement induced on the images of its on board camera(s) [1]. Feature based VO uses salient and repeatable features extracted and matched across the captured images. No matter which algorithm is used, the matched features are usually contaminated by outliers (wrong data associations). How to efficiently remove the outliers is the most delicate task in VO and still has large room for improvement.

This section introduces how to apply PURSAC to VO. The relations of features matching scores to their possibility to be outliers and to their location accuracy in images are investigated first. Then the sensitivity analysis of the egomotion model against samples noise is conducted. Based on these analyses PURSAC is elaborated aiming to design a purposive sample set selection procedure for the fundamental matrix estimation, and to improve the results of outlier removal and model fitting and in turn the VO performance.

Feature Matching Analysis

First let us analyse the correspondence of features matching score and the features' possibility to be inliers. Similar analysis has been done previously for SIFT [16] and some other descriptors [21]. In all experiments, regardless of which similarity function used, the fraction of inliers decreased almost monotonically as a function of the number of tentative matches [9]. This is verified by our test results for 169 pairs of images using SURF features[17]. All the results show that the features with lower matching scores have higher possibility of being outliers.

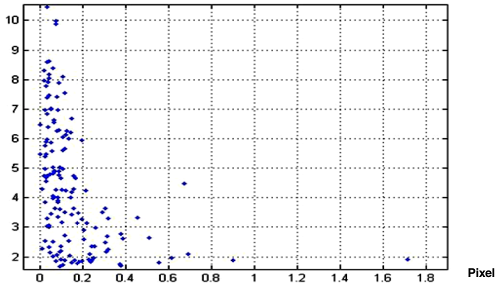


Fig. 4 Features' location accuracy and matching score correspondence

Another test was conducted to evaluate the robustness and accuracy of SIFT and SURF features' location against feature matching scores. Fig.4 is a test result for SIFT features. The horizontal axis is SIFT features' location accuracy in pixels, and vertical axis is the features' matching scores, which is the 'confidence level' of a matching is correct. It shows that as the confidence increases, so does the location accuracy (to an extent). This location error can be treated as sampling noise that is related to feature matching score. This observation will be counted in designing PURSAC rules for purposive sample subset selection.

Model Noise Sensitivity Analysis

Model noise sensitivity analysis is to find a strategic way to dilute the effect of sampling noise for model fitting. It has been found that the geometry of selected features in images affects VO result remarkably [10, 11]. The more evenly features are distributed in images, the less sensitive a model is to noise, and the more stable is VO results. The fact that the geometry of the features in images affects the VO results reflects the relation between sampling noise and the model to be fitted.

The fundamental matrix estimation with RANSAC etc. method takes geometric constraints introduced by a motion model. After feature detection and matching, matched features are nominated for subsequent procedure. In RANSAC, sample data sets are randomly chosen from entire matched features. However, the selected feature points may be close to each other (causing degenerate configuration), which will induce uncertainty of model fitting and affect the accuracy of estimation.

PURSAC Rules for Visual Odometry

Feature based VO requires efficient and robust model fitting. According to the statistical analysis of outlier possibility and features' location accuracy against the matching scores, rules for PURSAC implementation are innovated by considering both scores ranking and feature geometry.

a) All the matched features are ranked by their matching scores. The one with the highest rank is selected and the features close to it within a threshold $\boldsymbol{\rho}$ are excluded in following samples selection. This process iterates until all the matched features are either selected or excluded.
b) Only the selected features are used for searching the sample set for the consensus of model hypothesis but it is verified against the entire dataset. Sample subsets are purposely selected according to their ranking until reaching an initial iteration number $\boldsymbol{k}$.
c) Same as line fitting case, local optimization is then implemented to further increase the speed and certainty of model fitting. By improving accuracy and efficiency of fundamental matrix estimation, a precise and fast visual odometry can be achieved.

This set of rules is combined with both standard RANSAC and MLESAC for fundamental matrix estimation and visual odometry computation. The threshold $\boldsymbol{\rho}$ is set as 10 times of the inlier threshold in RANSAC (10x1 pixels), and the initial iteration number $\boldsymbol{k}$ is set to 20.

5 Experiment Results

Málaga 2009 Robotic Dataset Collection PARKING-6L is used in this paper for featured based VO testing [20]. A section of the dataset is selected from the images captured by a camera mounted on a test vehicle which runs a closed loop in a car park. The test runs 100 times to evaluate the precision and consistency of the VO results from different methods. Test results from PURSAC are compared with RANSAC and MLESAC, which were implemented in our previous approaches for VO [19] and monocular SLAM [18] respectively.

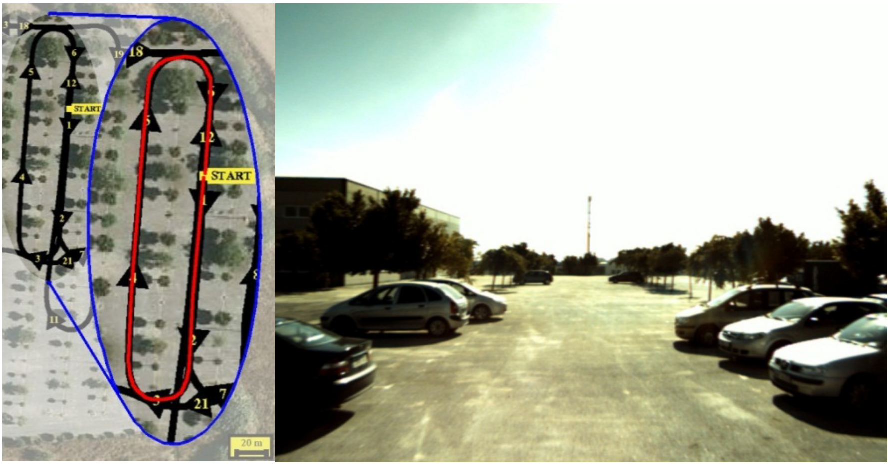


Fig. 5 Test field, trajectory (left) and a sample image for visual odometry

The red line in left image of Fig.5 is trajectory of the test in a car park. The right image is one of the images captured by an onboard camera for our visual odometry test.

Table 2 is the test results of matched features' inlier rate detected by different methods in 100 runs. Five pairs of images have the inlier rate from less than 50% to over 85%. The tests set two different success rate p_1 = 99% and p_2 = 99.99%. The number of iterations is calculated by (1) dynamically.

The results show that combining with either RANSAC or MLESAC, PURSAC achieves much better results than the original algorithms. The inlier rate is higher and the standard deviation of the number of detected inliers is much lower, which means PURSAC has much better and more consistent sample consensus. It is noted that for the images with high inlier rate (Image 58&59), the STD of both RP and MP reaches zero, entailing a complete certainty. Results show that the two different success rates p do not impact much on the inlier rate μ of PURSAC, but do for MLESAC and RANSAC. Even if a higher success rate p2 is selected for MLESAC and RANSAC, PURSAC still performs better with a low success rate, as the bold data indicated. Therefore, the number of iterations needed for PURSAC (RP and MP) is much lower than the original algorithms, leading to a faster process.

Table 2 Image Matching Monte Carlo Test Results

Image pair (Number of matches)	Inlier rate μ & number of iteration *N*	RO (RANSAC Original)		RP (RANSAC PURSAC)		MO (MLESAC Original)		MP (MLESAC PURSAC)	
		p_1	p_2	p_1	p_2	p_1	p_2	p_1	p_2
Image 2&3 (357)	Mean μ	68.74%	**70.03%**	**75.92%**	75.83%	68.10%	**69.37%**	**75.90%**	75.93%
	Inliers STD	10.39	**9.06**	**0.98**	1.16	12.78	**10.30**	**0.96**	1.0714
	Mean *N*	68.29	**116.83**	**47.66**	81.31	79.58	**125.68**	**49.61**	85.98
Image 27 &28 (390)	Mean μ	52.78%	**54.32%**	**58.39%**	58.46%	52.72%	**54.1%**	**58.45%**	58.36%
	Inliers STD	8.12	**6.82**	**3.12**	2.90	10.23	**9.27**	**3.09**	3.1527
	Mean *N*	447.92	**689.88**	**315.82**	585.35	475.55	**713.77**	**349.3**	610.29
Image 58 &59 (1019)	Mean μ	77.03%	**78.71%**	**85.57%**	85.57%	77.3%	**78.9%**	**85.57%**	85.57%
	Inliers STD	38.96	**31.92**	**0**	0	36.85	**32.05**	**0**	0
	Mean *N*	31.72	**50.60**	**21.02**	39.05	30.89	**51.68**	**23**	39.56
Image 2&4 (186)	Mean μ	55.22%	**56.81%**	**59.47%**	59.70%	55.35%	**56.23%**	**59.78%**	59.77%
	Inliers STD	4.97	**3.66**	**2.40**	1.72	4.56	**5.44**	**1.53**	0.6257
	Mean *N*	335.70	**507.45**	**230.12**	390.37	346.98	**582.5**	**228.08**	422.55
Image 2&6 (129)	Mean μ	44.32%	**44.99%**	**47.16%**	47.22%	43.81%	**43.95%**	**47.28%**	47.38%
	Inliers STD	1.96	**1.87**	**1.73**	1.52	2.25	**2.46**	**1.47**	1.6161
	Mean *N*	1471.0	**2567.2**	**1127.7**	2090.6	1640.6	**3193.4**	**1338.5**	2615.3

The test results on 103 pair of images for 100 runs are plotted in Fig.6. The top figure shows the number of matches N_m, the number of inliers N_{in} and inlier rate μ in each pair of images. The middle one is the mean difference of N_{in} detected by MO and MP. It indicates that MP can always detect more inliers than MO, especially in the case that N_{in} is low (image number 76). The bottom figure shows the standard deviation of N_{in} for 100 runs. MO has higher STD than MP in all the 103 pair of images. This proves that MP has better consistency than original MLESAC, which is also critical for model fitting.

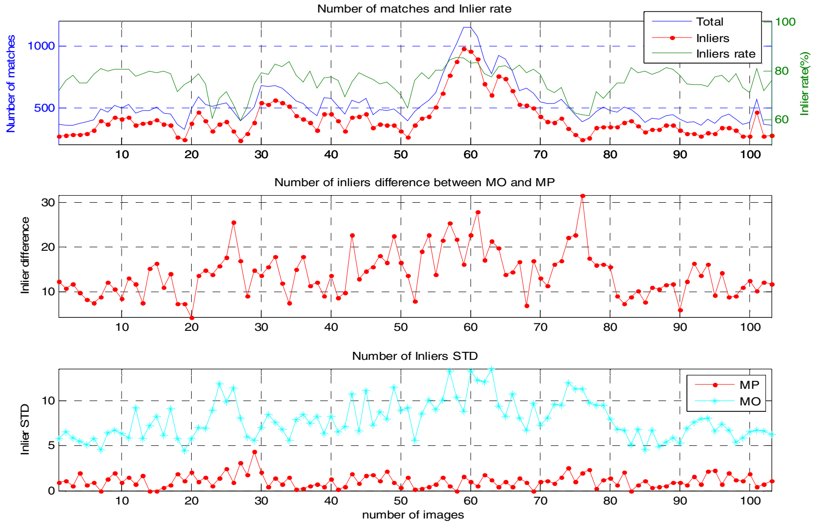


Fig. 6 Number of inliers in visual odometry tests

The trajectories of 100 runs VO using MLESAC (MO) and proposed PURSAC (MP) are plotted for analysis and comparison. As shown in Fig.7, PURSAC has significantly improved the performance of VO in terms of standard deviation. The final position of the true trajectory returns to the start position, however, due to the camera calibration uncertainty, there is a bias in both cases. The mean and standard deviation of final camera positions with different methods are listed in Table 3.

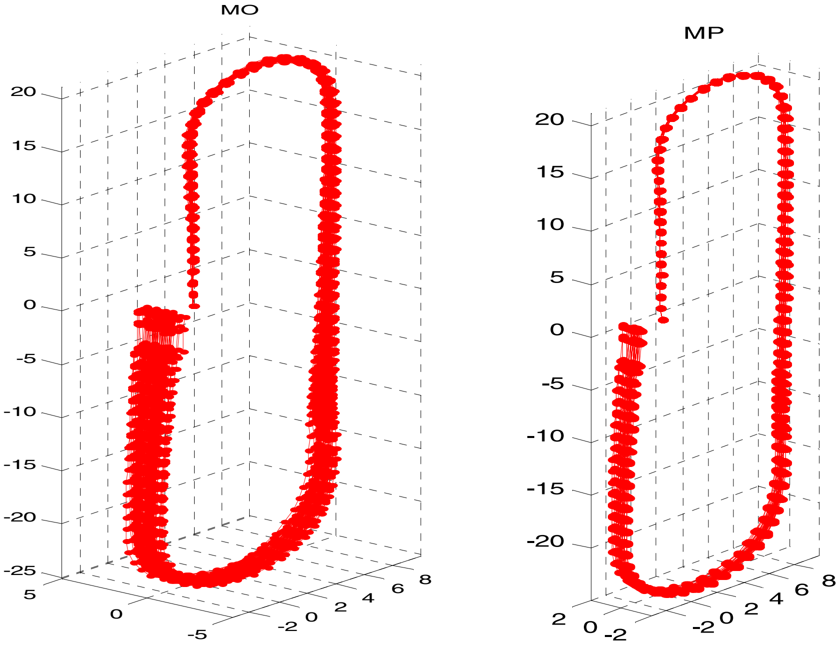


Fig. 7 The 100 trajectories of VO using MO and MP

Table 3 VO Final Position STD and Number of Iterations

Methods	**X STD/mean**	**Y STD/mean**	**Z STD/mean**	**Mean iterations**
RO (*p_2* = 99.99%)	0.215/ 1.247	2.220/ 1.349	0.217/ 0.319	147.51
RP (*p_1* = 99%)	**0.0840/** 1.207	**0.252/** 0.782	**0.0588/** 0.296	**65.87**
MO (*p_2* = 99.99%)	0.184/ 1.208	0.533/ 1.005	0.151/ 0.292	151.75
MP (*p_1* = 99%)	**0.0625/** 1.204	**0.236/** 0.791	**0.0415/** 0.295	**67.59**

Table 3 shows that the mean final camera positions with different methods are similar, which is a systemic bias to the ground truth. While for standard deviation and number of iterations, PURSAC achieves much better results than RANSAC and MLESAC. With less than half of the number of iterations, the final positions of the two PURSAC methods (RP and MP) have much smaller STD than that of the original RANSAC (RO) and MLESAC (MO).

6 Conclusion

This paper introduces PURSAC, a purposive sample set selection paradigm for model fitting. It has three major steps to implement. Firstly, instead of assuming all the samples have a same probability to be inliers, PURSAC seeks their differences and purposively selects sample sets. Secondly, as sampling noise always exists; the selection is also according to the sensitivity analysis of a model against the noise. The final step is to apply a local optimization for further improving its model fitting performance. PURSAC can combine with any model fitting algorithm that uses random sample sets selection, and achieve better outcomes with a smaller number of iterations.

Being a qualitative guidance in principle, PURSAC's implementation needs quantitative analysis to design executable rules for purposive sample consensus. Two examples are investigated in this paper. PURSAC is applied to line fitting to explain its principles, and then to visual odometry, which requires efficient, robust and precise model fitting.

Experimental results in the two examples show that PURSAC can achieve very high model fitting certainty using only a small number of iterations. It demonstrates much better performance than RANSAC and MLESAC. Applied in VO, concerning both features' geometry and matching score ranking, PURSAC improves the accuracy and efficiency of fundamental matrix estimation dramatically, resulting in a precise and fast visual odometry.

References

[1] Fraundorfer, F., Scaramuzza, D.: Visual Odometry Part II: Matching, Robustness, Optimization, and Applications. IEEE Robotics & Automation Magazine 19, 78–90 (2012)

[2] Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24, 381–395 (1981)

[3] Nister, D.: Preemptive RANSAC for live structure and motion estimation. In: Ninth IEEE International Conference on Computer Vision, 2003. Proceedings., vol. 1, pp. 199–206 (2003)

[4] Raguram, R., Frahm, J.-M., Pollefeys, M.: A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 500–513. Springer, Heidelberg (2008)

[5] Frahm, J.M., Pollefeys, M.: RANSAC for (quasi-) degenerate data (QDEGSAC). In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 453–460 (2006)

[6] Kai, N., et al.: GroupSAC: Efficient consensus in the presence of groupings. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 2193–2200 (2009)

[7] Rosten, E., et al.: Improved RANSAC performance using simple, iterative minimal-set solvers, arXiv preprint arXiv:1007.1432 (2010)

[8] Torr, P.H.S., Zisserman, A.: MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. Computer Vision and Image Understanding 78, 138–156 (2000)

[9] Chum, O., Matas, J.: Matching with PROSAC - progressive sample consensus. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, pp. 220–226 (2005)

[10] Capel, D.: An effective bail-out test for RANSAC consensus scoring. In: Proc. BMVC 2005, pp. 629–638 (2005)

[11] Scaramuzza, D., Fraundorfer, F., Siegwart, R.: Real-time monocular visual odometry for on-road vehicles with 1-point RANSAC. In: Proc. IEEE Int. Conf. Robotics and Automation (ICRA), pp. 4293–4299 (2009)

[12] Nister, D., Naroditsky, O., Bergen, J.: Visual odometry. In: Proc. Int. Conf. Computer Vision and Pattern Recognition, pp. 652–659 (2004)

[13] Raguram, R., Frahm, J., Pollefeys, M.: Exploiting uncertainty in random sample consensus. In: Proc. ICCV, pp. 2074–2081 (2009)

[14] McIlroy, P., Rosten, E., Taylor, S., Drummond, T.: Deterministic sample consensus with multiple match hypotheses. In: Proc. British Machine Vision Conf., pp. 1–11 (2010)

[15] Rodehorst, V., Hellwich, O.: Genetic Algorithm SAmple Consensus (GASAC) - A Parallel Strategy for Robust Parameter Estimation. In: Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW 2006), June 17-22, pp. 103–111 (2006)

[16] Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 20(2), 91–110 (2003)

[17] Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded up robust features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part I. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)

[18] Zhao, L., Huang, S., Yan, L., Wang, J.J., Hu, G., Dissanayake, G.: Large-Scale Monocular SLAM by Local Bundle Adjustment and Map Joining. In: Proc. of the 11th. Int. Conf. Control, Automation, Robotics and Vision (ICARCV 2010), pp. 431–436. IEEE Technical Committee, Singapore (2010)

[19] Wang, J.J., Kodagoda, S., Dissanayake, G.: Vision Aided GPS/INS System for Robust Land Vehicle Navigation. In: Proceedings of the 22nd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2009), Savannah, Georgia, USA, September 22-25, pp. 600–609 (2009)

[20] Blanco, J.-L., Moreno, F.-A., Gonzalez, J.: A Collection of Outdoor Robotic Datasets with centimeter-accuracy Ground Truth. Autonomous Robots 27(4), 327–351 (2009)

[21] Tordoff, B., Murray, D.W.: Guided sampling and consensus for motion estimation. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002, Part I. LNCS, vol. 2350, pp. 82–96. Springer, Heidelberg (2002)

[22] Chum, O., Matas, J., Kittler, J.: Locally optimized RANSAC. In: Proc. DAGM-Symposium, pp. 236–243 (2003)

Cooperative Targeting: Detection and Tracking of Small Objects with a Dual Camera System

Moein Shakeri and Hong Zhang

Abstract. Surveillance of a scene with computer vision faces the challenge of meeting two competing design objectives simultaneously: maintain a sufficient field-of-view coverage and provide adequate details of a target object if and when it appears in the scene. In this paper we propose a dual-camera system for tracking small objects based on a stationary camera and a pan-tilt-zoom (PTZ) camera. The utilization of two cameras enables us to detect a small object with the stationary camera while tracking it with the second moving camera. We present a method for modeling the dual-camera system and demonstrate how the model can be used in object detection and tracking applications. The main contribution of this paper is that we provide a model for explicitly computing the extrinsic parameters of the PTZ camera with respect to the stationary camera in order to target the moving camera at an object that has been detected by the stationary camera. Our mathematical model combines stereo calibration and hand-eye calibration algorithms as well as the kinematics of the pan-tilt unit, in order for the two cameras to collaborate. We present experimental results of our model in indoor and outdoor applications. The results prove that our dual camera system is an effective solution to the problem of detecting and tracking a small object with both excellent scene coverage and object details.

1 Introduction

In recent years visual surveillance research has attracted considerable attention due to its importance in both civilian and military applications [1, 2].

Moein Shakeri · Hong Zhang
Department of Computing Science, University of Alberta, Canada, Edmonton
e-mail: {shakeri,hzhang}@ualberta.ca

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_24, © Springer International Publishing Switzerland 2015

However, such a system faces the challenge of meeting two competing design objectives simultaneously: maintain a sufficient field-of-view coverage and provide adequate details of a target object if and when it appears in the scene. On one hand, a wide-angle camera can provide a large field of view, but it can capture only a small number of pixels of the target object. On the other hand, a PTZ camera can pan, tilt and zoom to improve the resolution of a small object in the scene; however, tracking small objects with a PTZ camera is still an open problem in computer vision. To overcome the limitation of either type of camera, there have been some surveillance systems that involve the collaboration between a moving camera with a large zoom range and a wide-angle stationary camera (see Section 2).

This paper deals with the problem of capturing a moving object in high resolution with two collaborating cameras, one stationary and the other on a PTU with a variable focal length (zoom). The role of the stationary camera is to detect the object of interest in the scene, and that of the moving camera is to place the object of interest in the center of its field of view at a high resolution. For capturing the close-up of a moving object, the moving camera must move and change its the position and orientation with respect to the stationary camera. To handle this change and map the location of the target object in the field of view of the stationary camera to the center of the field of view of the moving camera with an appropriate scale, we introduce a mathematical model for computing the pan and tilt angles and the focal length of the moving camera. Note that we make no assumption about the placement of the two cameras and that, if we choose the world reference frame to be that of the stationary camera, then the problem becomes one of computing the extrinsic parameters of the moving camera in order to target it at the object of interest.

The reminder of this paper is organized as follows. In Section 2, we discuss related works. In Section 3, we introduce our model for controlling the moving camera, based on the results of offline camera calibration and the object location detected online by the stationary camera. Experimental results in indoor and outdoor environments are described in Section 4. Finally Section 5 summarizes our method and concludes the paper.

2 Related Works

Tracking moving objects using moving cameras and specially PTZ cameras is found in many applications. In general, the existing methods can be divided in two approaches in terms of the camera configuration. In the first approach, there is a PTZ camera for both detection and tracking of moving objects [2, 5, 6], while in the second approach multiple cameras are involved [3, 7, 9, 16]. Methods using the first approach work by extracting features from the moving object, and this is often not robust in cluttered and dynamic backgrounds, especially when the object is small, due to the fact that the background is moving when the camera moves. Methods following the second approach is more robust to track small objects, with the fixed camera extracting the moving object by using background subtraction methods and the second camera tracking it. Some of the previous works use a master-slave camera configuration [3, 9, 16] where the master fixed camera detects and tracks the moving object in the scene while the PTZ camera is employed to acquire a close-up. This approach relies on offline calibration between the stationary and the PTZ camera; however, the basic assumption in existing methods is that either the offset between the fixed and the moving camera is ignored or the moving camera is placed in a special configuration with respect to the fixed camera with well aligned optical axes. Xu *et al.* [16] reported an autonomous surveillance system with multiple PTZ cameras assisted by a fixed wide-angle camera. The wide-angle camera provides large but low resolution coverage and detects and tracks all moving objects in the scene. Based on the output of the wide-angle camera, the system generates spatio-temporal observation requests for each moving object, which are candidates for a close-up using PTZ cameras. The basic assumption in their work is all cameras are already calibrated and all transformation between cameras are available.

Recently, Choi *et al.* [3] proposed a camera system consisting of one PTZ camera and two stationary cameras to acquire high resolution face images. They used a camera calibration method based on the coaxial configuration among the three cameras. Specifically, they configured two stationary cameras, one above (horizontal camera), and one beside (vertical camera) the PTZ camera, so that the x coordinate (y coordinate) of the horizontal (vertical) camera's focal point coincides with the x coordinate (y coordinate) of the PTZ camera's center of rotation. The limitation of their system is that they have assumed all cameras are configured to have parallel camera axes.

Marchesotti *et al.* [9] used two cameras including one stationary and one PTZ camera to capture high resolution face images. The first camera monitors the entire area of interest and detects the moving objects using change detection techniques, while the second camera tracks the objects at high resolution. They employed the standard calibration procedure to obtain the intrinsics of the individual cameras. The extrinsics between the cameras however are assumed to consist of simple translations in a 2D plane. No general calibration method is employed to obtain these translations.

Hampapur *et al.* [7] used stationary and PTZ cameras to estimate the 3D world coordinates of the objects and then zoom to capture a high resolution image. Stillman in [14] used multiple stationary cameras to estimate the location of objects in a calibrated scene and then applied the PTZ camera for tracking them. Bodor *et al.* [1] proposed a dual-camera system which is

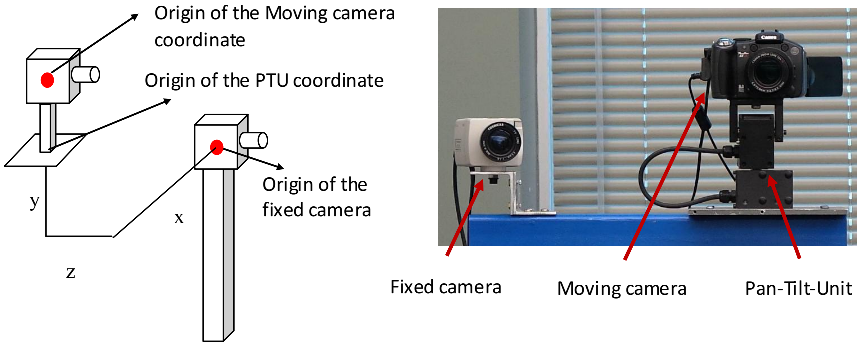

Origin of the Moving camera coordinate
Origin of the PTU coordinate
Origin of the fixed camera
y
x
z
Fixed camera
Moving camera
Pan-Tilt-Unit

Fig. 1 (a)Schematic of a system with two cameras and a PTU supporting one of the cameras and (b) view of our experimental platform

comprised of a wide-angle fixed field of view camera coupled with a PTZ camera to make detailed measurements of activity recognition applications. Zhou *et al.* [17] proposed a different kind of master/slave system. This system collaborates based on one fixed camera and one camera that can zoom in and out as well as pan and tilt using a Pan Tilt Unit (PTU). Geometric models were proposed for calibrating between a camera and a PTU [4, 10] and active calibration of a multi-camera system consists of zoom cameras mounted on PTU [15]. After a coarse initial calibration, the probability of each relative pose is determined using a probability distribution based on the camera image. Works in [1, 8, 5] described the mathematical model of the collaboration between two cameras, but made the same limiting assumption as in [9] of simple 2D translational change between the camera positions. The methods in [7, 14, 18] assumed that known crude relationship among the cameras, and no formal treatment for the spatial relationship among the collaborating cameras was attempted.

In this paper, we introduce a dual-camera detection and tracking system in which there is no limitation for configuration of the locations of the two cameras. We derive a mathematical model for controlling the moving camera based on the location of the target object in the view of the stationary camera.

3 Modeling of a General Active Dual-Camera System

In this section we present a summary of our system which contains one fixed wide-angle camera and one camera with a changeable focal length mounted on a PTU. We will also describe the mathematical model for our system. An example of such a system is shown in Fig. 1.

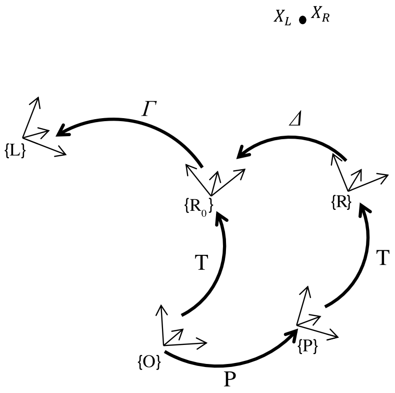


Fig. 2 Platform of active dual-camera coordinate system

In this system when PTU moves not only does the camera rotate but it also translates to a new position. As a result, the extrinsic parameters of the moving camera are changed with respect to the stationary camera. In all mentioned methods in Section 2, authors a simple spatial relation between the active camera and the stationary camera. This assumption could be acceptable with some errors when the active camera has a small zoom range or if the two cameras are mounted in a special configuration. Alternatively and more appropriately, we can build an exact mathematical model for the relationship between the fixed camera and the moving camera on the PTU to compute the variable extrinsic parameters from the moving camera to the stationary camera online.

To establish this mathematical model, we first define the coordinate frames associated with the various objects of importance in the system as shown in Fig. 2. Assume that the left camera is stationary and right camera is mounted on the PTU. Let X be a space point of interest, and X_L and X_R be its 3D coordinates in the left camera coordinate frame $\{L\}$ and right camera coordinate frame $\{R\}$, respectively. $\{R\}$ is initially at $\{R_0\}$ before PTU motion. We further define Γ to be the transformation of $\{L\}$ with respect to $\{R_0\}$, T the transformation of $\{R_0\}$ with respect to the PTU's coordinate frame $\{O\}$. Finally, when the PTU moves, its new position $\{P\}$ with respect to the original $\{O\}$, is defined by the transformation P, which can be computed from PTU's direct kinematics. Note that after the PTU moves to $\{P\}$, the right camera's coordinate frame $\{R\}$ with respect to the PTU's initial position $\{O\}$ can be obtained through transformation P and the known transformation T (see below).

Upon detection of a point of interest X by the left camera, the goal is move the right camera by way of the PTU from $\{R_0\}$ to $\{R\}$ in such a way that

X is centered in the field of view of the right camera. X in the left camera and in the right camera are related by

$$X_R = \mathit{\Delta} \mathit{\Gamma} X_L \tag{1}$$

where $\mathit{\Delta} = T^{-1}P^{-1}T$, so that

$$X_R = T^{-1}P^{-1}T\mathit{\Gamma} X_L \quad \text{or} \quad TX_R = P^{-1}T\mathit{\Gamma} X_L \tag{2}$$

From Eq. (2), we are now able to compute the control variables of the PTU given X_L. Specifically, X_L and X_R are projected to $\{L\}$ and$\{R\}$ via $x_L = C_L X_L$ and $x_R = C_R X_R$ where C_L and C_R are the camera matrices and they are available through camera calibration. For x_L that defines an object of interest in the left camera's image plane, its 3D coordinates X_L are estimated, using prior knowledge about the object size (e.g., the rough height of a person or dimensions of a bird) and the camera's intrinsics. In addition, $\mathit{\Gamma}$ is known through the stereo calibration between $\{L\}$ and $\{R_0\}$. The only remaining unknown is the transformation T, which can be obtained with hand eye calibration as described in the next section.

3.1 Hand-Eye Calibration

The unknown transformation T can be obtained by one of the existing algorithms in robot hand-eye calibration. This is the common task of computing the relative 3D position and orientation between the camera mounted on the end effector of a robot (such as a PTU) and the coordinate frame of the robot wrist in a so called eye-on-hand configuration [13]. In our model PTU works the same as robot. Therefore, hand-eye calibration is defined between coordinate frames $\{O\}$ and $\{R_0\}$ in Fig. 2, and it is redrawn in Fig. 3. Different algorithms have been proposed for hand-eye calibration and they all attempt to solve a system of linear equations of the form: $AT = TB$ where A and B are known from robot forward kinematics and from stereo camera calibration, respectively [13, 11]. To solve for T, it is necessary to make at least two independently movements and form a system of equations such as Eq. (3).

$$A_iT = TB_i \qquad i = 1, ..., n \tag{3}$$

For each configuration of the robot (PTU), A_i is available through direct kinematics and B_i can be found by using a stereo calibration algorithm with the camera observing a calibration target. T can be obtained optimally by minimizing the error:

$$\eta = \sum_{i=1}^{n} E(A_iT,\ TB_i) \tag{4}$$

where E is a distance metric on the Euclidean group [11].

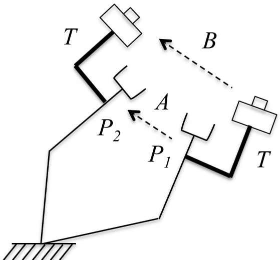


Fig. 3 Schematic of the hand-eye calibration for finding T by solving $AT = TB$

3.2 Computation of PTU Variables

The final task is to obtain the pan and tilt angles, θ_1 and θ_2, in order to center X in the right camera $\{R\}$. This is equivalent to requiring $X_R = (0, 0, z)$ where z is the depth of X which has been estimated with the camera intrinsics and known object size, as mentioned. To begin, we rewrite Eq. (2) as follows:

$$X_R^{'} = P^{-1}X_L^{'} \quad or \quad PX_R^{'} = X_L^{'} \tag{5}$$

where

$$X_R^{'} = TX_R \text{ and } X_L^{'} = T\Gamma X_L \tag{6}$$

For the PTU, P is a rotation transform and Eq. (5) can be expanded as

$$PX_R^{'} = R(\theta_1, \theta_2)\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} = X_L^{'} \tag{7}$$

where from the forward kinematics of the PTU, P is given by

$$P = R(\theta_1, \theta_2) = Rot(Y, \theta_1)Rot(X, \theta_2) = \begin{bmatrix} c_1 & s_1s_2 & s_1c_2 \\ 0 & c_2 & -s_2 \\ -s_1 & c_1s_2 & c_1c_2 \end{bmatrix} \tag{8}$$

where s_i and c_i are sine and cosine functions of the two joint angles. From Eq. (8) and the second row of Eq. (7) we have

$$c_2y - s_2z = b \tag{9}$$

Eq. (9) may be solved by making the following trigonometric substitution [12]:

$$r_1cos\varphi = z \text{ and } r_1sin\varphi = y \tag{10}$$

where

$$r_1 = +\sqrt{(y^2 + z^2)} \text{ and } \varphi = tan^{-1}(y/z) \tag{11}$$

Substituting Eq. (11) into Eq. (9) we obtain

$$sin(\theta_2 - \varphi) = -b/r_1 \tag{12}$$

Therefore, we obtain two solutions to θ_2

$$\theta_2 = tan^{-1}\frac{y}{z} + sin^{-1}\frac{-b}{r_1} \quad \text{and} \quad \theta_2 = tan^{-1}\frac{y}{z} + \pi - sin^{-1}\frac{-b}{r_1} \tag{13}$$

With Eq. (7) and Eq. (13), we have

$$Rot(Y, \theta_1) \begin{bmatrix} x^{'} \\ y^{'} \\ z^{'} \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{14}$$

where $\begin{bmatrix} x^{'} \\ y^{'} \\ z^{'} \end{bmatrix} = Rot(X, \theta_2) \begin{bmatrix} x \\ y \\ z \end{bmatrix}$. Eq. (14) expands into

$$\begin{bmatrix} c_1 & 0 & s_1 \\ 0 & 1 & 0 \\ -s_1 & 0 & c_1 \end{bmatrix} \begin{bmatrix} x^{'} \\ y^{'} \\ z^{'} \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{15}$$

From the first row of Eq. (14), we have $c_1 x^{'} + s_1 z^{'} = a$. This equation can be solved the same as Eq. (9), and the solutions are

$$\theta_1 = tan^{-1}\frac{z^{'}}{x^{'}} + cos^{-1}\frac{a}{r_2} \quad \text{and} \quad \theta_1 = tan^{-1}\frac{z^{'}}{x^{'}} - cos^{-1}\frac{a}{r_2} \tag{16}$$

where $r_2 = \sqrt{(x^{'2} + z^{'2})}$ and $\varphi = tan^{-1}(z^{'}/x^{'})$

Once the object is centered in the right camera, we can vary its size in the camera by controlling the focal length. Since the camera is calibrated, for a given desired object size u_d in pixels, we can easily find the desired focal length of the right camera using trigonometry and the pinhole camera model and, it is, in pixels and millimeters (f_p and f_{mm}), given by:

$$f_p = u_d\frac{z}{x} \quad \text{or} \quad f_{mm} = pu_d\frac{z}{x} \tag{17}$$

where z and x are the depth and the size of the object, and p is the pixel size of the camera in *mm*. Note that that f_{mm} is limited by the maximum focal length of the zoom camera.

The model for controlling the PTU is summarized in Table 1, and it shows how the image coordinates of the target in the left view are mapped to the pan and tilt angles and the focal length of the moving camera.

Table 1 Cooperative Targeting Algorithm

Offline Calibration

- Calibrate each camera independently to determine their intrinsics C_R, C_L, and pixel size p
- Identify the transformation Γ by stereo calibration between the fixed and the moving camera with pan/tilt angles = 0
- Perform hand-eye calibration to find T by Eqs. (3) and (4)

Online Detection and Tracking

- In the fixed camera, detect the object of interest centered at x_L in image
- Estimate X_L using known object size and camera model $x_L = C_L X_L$
- Set $X_R = [0, 0, z]^T$ where z is the depth of the object in X_L
- Compute $X_L^{'}$, $X_R^{'}$ with Eq. (6), and pan and tilt angles by Eqs. (13) and (16),
- Find the focal length f_{mm} by Eq. (17)
- Command the PTZ with θ_1, θ_2 and f_{mm} accordingly

4 Experimental Results

We are interested in the use of a multi-camera system for object detection and tracking in indoor and outdoor environments. In this section, we will present experimental results to demonstrate the utility of our proposed model in this application. Our system (see Fig. 1) consists of a stationary camera with a fixed lens (a Philips LTC 0600) and a zoom camera (a Canon PowerShot S5 IS with 12x optical zoom). The Philips camera is interfaced to a Linux PC via an Axis video server (241Q) using http, and the Canon camera is connected to the PC through a USB serial port. The communication between the Canon camera and the PC is established with the camera's publicly available API that allows the remote capture of its images and the selection of its focal length. Finally the pan tilt motion of the Canon camera is achieved through a PTU by Directed Perception (D46), which has a pan range of −180° to 180°, and a tilt range of −80° to +30°. While target detection is performed with images from the stationary camera, object tracking and zooming into the region of interest are handling by the moving camera by the algorithm described in Table 1. The control software is developed in Matlab.

4.1 Offline Calibration

As described in Table 1, our model has an offline calibration step, followed by an online detection and tracking step. In the first step we calibrate each camera in turn to determine its intrinsics. The camera intrinsic parameters are shown in Table 2. Also, we identify the transformation by the stereo calibration algorithm between the fixed camera and the initial position of the moving camera. Furthermore, we need to compute the transformation between the PTU and the moving camera so as to determine the extrinsic parameters of the moving camera when the PTU moves. For this we apply hand eye calibration algorithm described in Section 3.1. Table 3 shows the numerical values of transformation Γ and T.

Table 2 Intrinsic parameters of the cameras

Camera model	Focal length (pixels/mm)	Camera center (pixels)	CCD Size (mm)	Pixel Size p (mm)
Canon PowerShot* (zoom camera)	[693, 692] [6.237, 6.228]	[235, 326]	[5.74, 4.31]	0.009
Philips Camera (stationary camera)	[684, 686] [6.840, 6.860]	[239, 325]	[6.40, 4.80]	0.01

*: The focal length for the Canon camera was calibrated at its minimum.

4.2 Online Detection and Tracking

In the second step we perform online detection and tracking. Figure 4 shows an example of target detection and tracking based on our model. The target (a book) is small and far from the camera, and Fig. 4(a) and Fig. 4(b) show its initial view in the stationary (left) and moving camera (right). Using a color-based appearance template, we first detect the target, and then find its pixel coordinates to be $[529, 275]$, in the 640×480 left image. We can estimate the depth of the object with respect to the stationary camera knowing the real size of the object using (17) and Table 2. For illustration purposes, we assume the estimated depth of the target to be $z = 10000$ mm, and we compute its 3D coordinates with respect to stationary camera using Table 2. The estimated position of the object, $X_L = [2973, 519, 10000, 1]^T$ in mm in homogeneous coordinates. So, the 3D coordinates of the point with respect to the moving camera before any pan-tilt motion can be computed, also in homogeneous coordinates, as follows:

$$X_R = \Gamma X_L = [811, 563, 10464, 1]^T \tag{18}$$

Table 3 Transformation matrices for our dual camera system

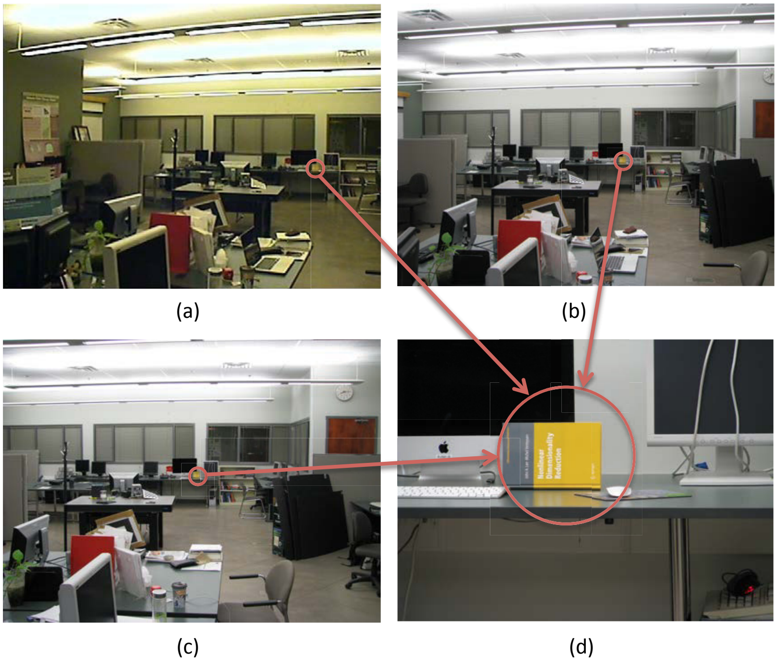
$$\varGamma = \begin{bmatrix} 0.97 & 0.02 & -0.24 & 356.18 \\ -0.02 & 1.00 & 0.00 & 93.59 \\ 0.24 & -0.00 & 0.97 & 38.30 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T = \begin{bmatrix} 0.99 & 0.0032 & 0.00 & -159.97 \\ 0.00 & 1.00 & 0.00 & -90.01 \\ 0.00 & 0.0011 & 1.00 & -39.98 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For the object to be at the center of the moving camera, we need to control the PTU it is centered with respect to the moving camera. We compute its coordinates according to Eq. (6) and Table 2. Setting $X_R = [0,\ 0,\ 10464,\ 1]^T$, we have

$$X_R^{'} = TX_R = [-143.26,\ -79.54,\ 10424,\ 1]^T \tag{19}$$

$$X_L^{'} = T\varGamma X_L = [661.92,\ 481.35,\ 10426,\ 1]^T \tag{20}$$

Finally, from Eqs. (16) and (13), pan and tilt angles are $\theta_1 = -3.08°$ and $\theta_2 = 4.44°$.



(a)

(b)

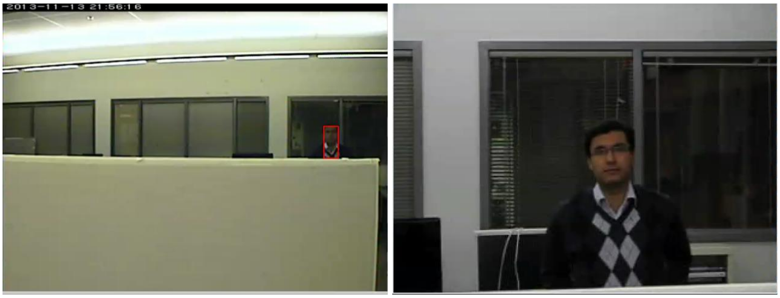
(c)

(d)

Fig. 4 Result of an indoor experiment (a) and (b) show position of the target before tracking in both cameras, (c) shows the detected target after tracking, and (d) shows the close-up of the target in the moving camera

After applying the rotation of $[\theta_1, \theta_2]$ to the PTU, the object of interest moves to the center of the image as shown in Fig. 4(c). In addition, we use Eq. (17) to compute the focal length of the camera zoom in order to capture a high resolution view of the object. The object (book) has a size of $x = 234$ mm in this case. If its desired size in the image is $u_d = 160$ pixels, then the focal length should be 64.14 mm or we need a zoom factor of 10.3, which our camera is able to provide with its 12x zoom. Fig. 4(d) shows the close-up of the target object in the moving camera.

We have also applied our algorithm in a background subtraction application. Fig. 5 shows the results from one experiment in the indoor environment where a person moves across the scene and is detected by the stationary camera using the standard adaptive Gaussian mixture model (GMM) for extracting moving pixels and simple connected-component tracking to identify the object. The pixel coordinates of the object are used to compute the pan and tilt angles of the moving camera, with the close-up shown in Fig. 5(b).



(a) (b)

Fig. 5 Result of another indoor experiment (a) shows position of the detected moving object by stationary camera, and (b) shows position and close-up of the moving object in moving camera after PTU motion

We have also conducted experiments with our system in video surveillance in the outdoor environment. Fig. 6 shows some example results where the left column displays the detecting foreground moving objects in the scene, the middle column wide-angle view of the monitored scene from the stationary camera, and the right column the close-up of the detected objects in the PTZ camera. The detection is once again performed with the straightforward Gaussian mixture model background subtraction, and the control of the camera is done through our proposed model. The detection and tracking is conducted in real time. In case of multiple objects in the view, we choose to track the first object in the scene until it disappears, before the system switches to the next object of interest in the stationary camera. The advantage of our system to target an object through cooperative control of the two cameras is clearly demonstrated, with the stationary camera maximizing the coverage of the scene and the moving camera providing a clear view of the object of interest to all its identification and further analysis.

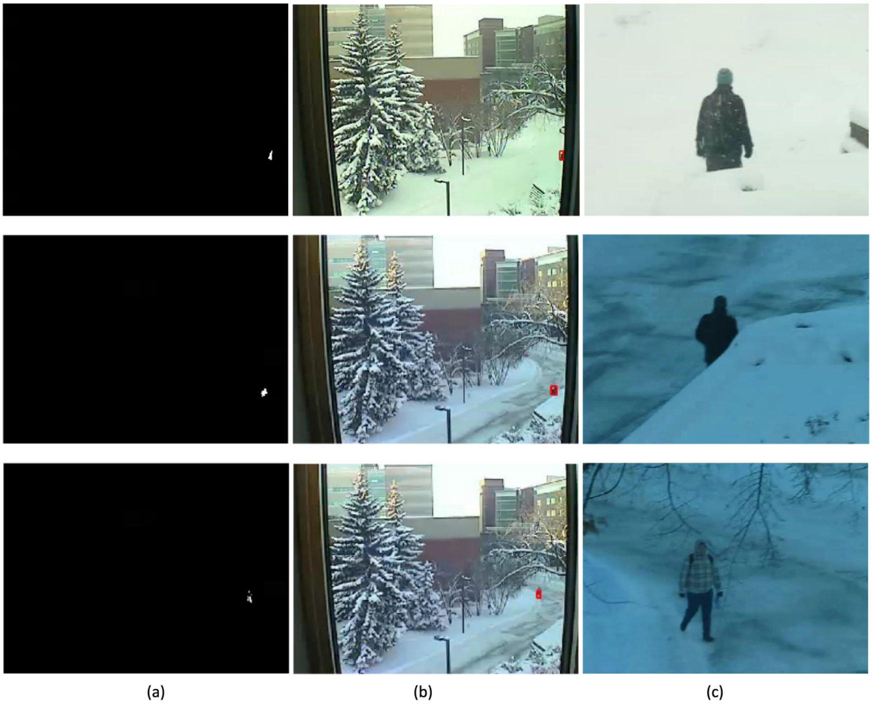


Fig. 6 Result of an outdoor experiment (a) shows results of adaptive GMM by stationary camera, (b) shows the position of the detected moving object by stationary camera, and (c) illustrates the close-up of the detected object.

5 Conclusion

We have presented a novel dual-camera system in which a stationary wide-angle camera detects an object of interest in its view, and the moving PTZ camera is directed at the moving object and zooms in appropriately to acquire a high-resolution view of the object. The transformation between the two cameras is explicitly modeled mathematically and used in real time to control the moving camera. Our model does not ignore the offset between the two cameras, a common practice in previous systems when an active camera was involved. We have validated our approach through extensive real experiments in both indoor and outdoor environments.

References

1. Bodor, R., Morlok, R., Papanikolopoulos, N.: Dual-camera system for multi-level activity recognition. Intelligent Robots and Systems (IROS) 1, 643–648 (2004)

2. Cai, Y., Medioni, G.: Towards a practical PTZ face detection and tracking system. In: IEEE Workshop on Applications of Computer Vision (WACV), USA, pp. 31–38 (2013)
3. Choi, H.C., Park, U., Jain, A.K.: PTZ camera assisted face acquisition, tracking and recognition. In: Fourth IEEE International Conference on Biometrics: Theory Applications and Systems (BTAS), Washington DC (2010)
4. Detesan, O., Arghir, M., Solea, G.: The Mathematical Model of the Pan-Tilt Unit Used in Noise Measurements in Urban Traffic. In: Fitt, A.D., Norbury, J., Ockendon, H., Wilson, E. (eds.) Progress in Industrial Mathematics at ECMI 2008. Mathematics in Industry, pp. 791–796. Springer, Heidelberg (2010)
5. Dinh, T., Vo, N., Medioni, G.: High resolution face sequences from a PTZ network camera. In: IEEE International Conference on Automatic Face and Gesture Recognition and Workshops, USA (2011), doi:10.1109/FG.2011.5771454
6. Dinh, T., Yu, Q., Medioni, G.: Real time tracking using an active pan-tilt-zoom network camera. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), USA (2009), doi:10.1109/IROS.2009.5353915
7. Hampapur, A., et al.: Face cataloger: Multi-scale imaging for relating identity to location. In: Advanced Video and Signal Based Surveillance (2003)
8. Lalonde, M., et al.: A system to automatically track humans and vehicles with a PTZ camera. In: Defense and Security Symposium. International Society for Optics and Photonics (2007)
9. Marchesotti, L., et al.: Cooperative multi-sensor system for real-time face detection and tracking in uncontrolled conditions. In: Image and Video Communications and Processing, pp. 100–114 (2005)
10. Ngan, P., Valkenburg, R.: Calibrating a pan-tilt camera head. In: Image and Vision Computing Workshop (1995)
11. Park, F., Bryan, J.: Robot sensor calibration: solving AX=XB on the Euclidean group. IEEE Transactions on Robotics and Automation 10(5), 717–721 (1994)
12. Paul, R., Zhang, H.: Computationally efficient kinematics for manipulators with spherical wrists. Int. J. Robot. Res. 5(2), 32–44 (1986)
13. Shiu, Y., Ahmad, S.: Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form AX= XB. IEEE Transactions on Robotics and Automation 5(1), 16–29 (1989)
14. Stillman, S., Tanawongsuwan, R., Essa, I.: Tracking multiple people with multiple cameras. In: Proc. Int. Conf. Audio and Video based Biometric Person Authentication (1999)
15. Wang, T., et al.: Active stereo vision for improving long range hearing using a laser Doppler vibrometer. In: IEEE Workshop on Applications of Computer Vision, WACV (2011)
16. Xu, Y., Song, D.: Systems and algorithms for autonomously simultaneous observation of multiple objects using robotic PTZ cameras assisted by a wide-angle camera. In: IROS, pp. 3802–3807 (2009)
17. Zhou, F., et al.: A Remote Scene Binocular Video Surveillance System with Small Object Tracking. In: Zhou, F. (ed.) 2nd International Congress on IEEE Image and Signal Processing, CISP (2009)
18. Zhou, X., et al.: A Master-Slave System to Acquire Biometric Imagery of Humans at a Distance. In: Proc. ACM SIGGM Intl. Workshop Video Surveillance, pp. 113–120 (2003)

Experiments on Stereo Visual Odometry in Feature-Less Volcanic Fields

Kyohei Otsu, Masatsugu Otsuki, and Takashi Kubota

Abstract. This paper describes a stereo visual odometry system for volcanic fields which lack visual features on the ground. There are several technical problems in untextured terrain including the diversity of terrain appearance, the lack of well-tracked features on surfaces, and the limited computational resources of onboard computers. This paper tries to address these problems and enable efficient and accurate visual localization independently of terrain appearance. Several key techniques are presented including a framework for terrain adaptive feature detection and a motion estimation method using fewer feature points. Field experiments have been conducted in volcanic fields for validation and evaluation of the system effectiveness and efficiency.

1 Introduction

The robotics in unstructured and dynamic environments has been studied extensively for these decades. The robots applied to such challenging areas are often referred as 'field robots', which should be more important in the future. The applications of field robots spread widely, including construction, forestry, agriculture, mining, subsea, intelligent highways, search and rescue, military, and space. Especially, hazardous areas where humans cannot work are the important fields for such robots, represented by active volcanic fields. The robotic exploration of volcanic fields are highly desired since detailed observations on site can provide valuable re-

Kyohei Otsu
The University of Tokyo, 7-3-1 Hongo, Bunkyo, Tokyo, Japan
e-mail: kyon@ac.jaxa.jp

Masatsugu Otsuki · Takashi Kubota
Institute of Space and Astronautical Science, Japan Aerospace Exploration Agency, 3-1-1 Yoshinodai, Chuo, Sagamihara, Kanagawa, Japan
e-mail: otsuki.masatsugu@jaxa.jp, kubota@isas.jaxa.jp

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_25, © Springer International Publishing Switzerland 2015

sults for geoscientists to figure out the mechanism of eruptions, and for rescues to reduce the impact of disasters [20].

In order to deploy robots into the hazardous areas without human intervention, robotic autonomy is required not only to accomplish tasks but also to keep their safety. One of the important techniques is localization, i.e., to identify their orientation and position. Accurate localization enables a robot to explore efficiently, obtain precise data about the environment, and keep safety from collapsed obstacles and dangerous places. GPS (Ground Positioning System) is widely used for this purpose, while it can provide only position (not orientation) and it cannot be used if the satellites are not visible such as in a valley.

Recently, localization approaches using vision sensors are extensively studied. It is called 'visual odometry (VO)' in contrast with conventional wheel odometry (WO) that uses wheel encoders to estimate the motion and direction. The accuracy of WO is harmfully affected by wheel slips which a robot frequently experiences in unstructured natural terrain. In contrast, VO is not degraded by slips, hence it can provide accurate pose estimation even in slippery sandy terrain or steep slopes.

Meanwhile, the vision-based method may suffer in several specific cases. One of the crucial problems is the failure of visual feature tracker. Recent VO systems extract visual features from terrain and track them frame by frame [4, 15]. However, in some feature-less terrain where fewer features found on the ground surface, stable feature tracking can occasionally be difficult. The lack of well-tracked features results in poor accuracy or failure of motion estimation. Moreover, the performance of feature tracker depends on the parameters such as thresholds and window size. A field robot encounters various types of terrain while exploring natural scenes and thus, it is hard to find the best parameters that fit to a new terrain. Another challenge is the high computational cost for dealing with 2D images. In order to perform real-time operations using limited onboard processors and memories, the efficiency of algorithms must not be neglected. One will find the tradeoff problem between the computational efficiency and the accuracy of estimation.

This paper introduces a VO system for a wheeled robot equipped with a stereo camera rig. It employs an algorithm adaptive to variable terrain in order to exhibit higher robustness for terrain diversity. Furthermore, for real-time operations, a novel efficient algorithm for estimating motion is adopted. The proposed system is evaluated through field experiments in volcanic fields with a testbed robot developed for future volcanic or planetary exploration missions.

2 Related Works

The research on VO in volcanic fields may be relevant to the one that visually estimates poses of mobile vehicles in untextured environments. The environments span volcanic fields, planetary surfaces, arctic regions, and paved roads. Broadly speaking, the approaches of related works can be divided into three types.

The first one is to preprocess images so that a feature tracker performs properly. An image enhancement method using adaptive histogram equalization was demonstrated to be effective for feature limited arctic images [26]. The second way is to select a better feature detector. Several detectors finding scale-space features are stable across scale changes of images, such as SIFT [13] or CenSurE [1]. Edge points are another option due to their sufficient availability even in untextured environments [25]. The third method is to adapt grids on a image. It divides an image into several blocks and selects best features in each block [24].

While the conventional techniques have enabled robust motion estimation in feature-poor terrain, this paper employs other approaches to improve it further. First, a terrain-adaptive feature detector is employed. Natural environments exhibit various appearance, which makes it hard to find a universally good feature detector. The proposed method selects a detector from several prepared detectors in accordance with the terrain appearance. Secondly, a novel motion estimation method is employed to reduce the number of required feature points. Compared to the conventional three-point algorithms for a stereo rig, the proposed method merely uses one near point and one far point. It also mitigates the computational burden in combination with an iterative outlier rejection scheme.

3 Technical Approaches

An image pair grabbed with a rig is processed along the pipeline shown in Fig. 1. This section mentions a detector adaptive to terrain appearance, and a method to recover rigid motion from the set of corresponding 3D points.

3.1 Feature Extraction

Feature extraction is the first step of VO and deeply related to the later tracking process. One crucial problem in feature extraction is the lack of well-tracked distinctive features in untextured scenes. Fig. 2 shows a typical scene in a volcanic fields with FAST corners [19] extracted. The low-textured terrain with a small portion of high-contrast objects imposes difficulty in feature extraction. Furthermore, the diversity

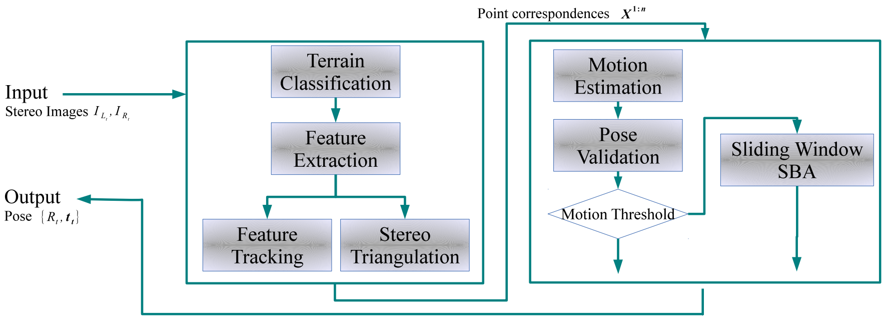


Fig. 1 Block diagram of feature-based VO algorithm

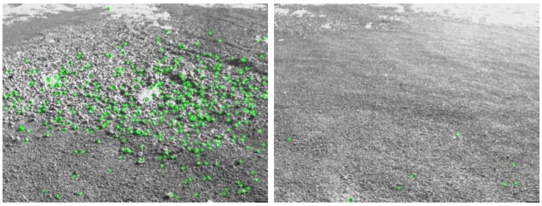


Fig. 2 Examples of a volcanic field with FAST corners extracted. Left: textured terrain with rich features. Right: untextured terrain with poor features.

of terrain appearance is identified as a challenging problem since the parameters used in a detector cannot be universal to all terrains. The properties required for a feature detector are summarized as follows:

- **Repeatability** to re-appear a feature in different images of same scenes.
- **Distinctiveness** to be distinguishable from other features.
- **Flexibility** to be adaptive to many types of terrain.
- **Invariance** to be invariant to the motion of a robot or illumination changes.
- **Computational efficiency** to perform the algorithm in real time.

As of today, numerous number of feature detectors have been proposed including [10, 13, 19, 1]. A survey was conducted to evaluate the performance of these detectors in the context of visual tracking [9]. The comprehensive evaluation under various conditions provides the insights about the characteristics of each detector. In conclusion, they mentioned the difficulty to derive a universally appropriate decision.

A straightforward answer for this problem is to use the combination of several detectors. The previous work of authors introduced the hybrid of several detectors which selects the appropriate detector for current terrain [17]. Using the hybrid detector, features can be extracted in wide variety of appearance in natural terrain.

The hybrid detector is composed of two parts: *texture assessment* and *detector switching*. The texture assessment part analyzes the texture of current terrain using a machine learning technique. In Fig. 3, the terrain is classified into three classes (ROUGH, SMOOTH, and SHADOW) so that each segments can be characterized by the impact on feature extraction. The AdaBoost [7] is employed to classify image patches. In order to describe the image patches, four statistical values are computed for the probability distribution function P of image intensity:

$$\mu = \sum_m mP(m) \tag{1}$$

$$\sigma^2 = \sum_m (m-\mu)^2 P(m) \tag{2}$$

$$E = \sum_m P(m)^2 \tag{3}$$

$$p_o = \sum_{\substack{m<\mu-3\sigma,\\ \mu+3\sigma<m}} P(m) \tag{4}$$

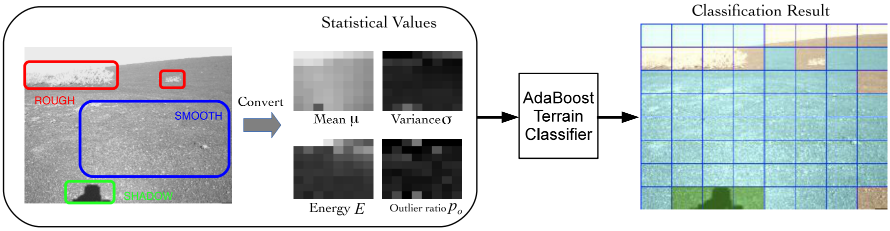


Fig. 3 Terrain classification using a machine learning technique. ROUGH(red): feature-rich terrain. SMOOTH(blue): feature-poor terrain. SHADOW(green): the shadow of robots which can be outliers for motion estimation.

where m denotes pixel brightness (ranging 0 to 255 for 8-bit depth images). Finally, the concatenated AdaBoost classifiers enables $\mathbb{R}^4 \mapsto \mathbb{R}^3$ mapping by the hand-labeled supervised learning.

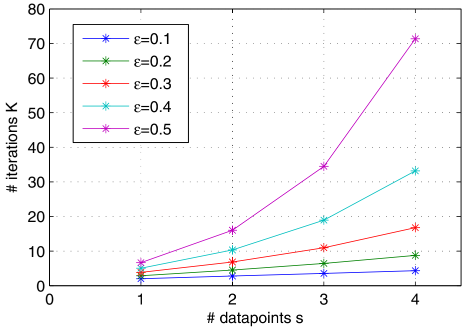
Based on the terrain classification, an appropriate detector is selected from a pre-defined detector set. Through the preliminary experiments, parameter-tuned Harris [10] for SMOOTH and FAST [19] for ROUGH are selected. Note that scale-invariant detectors such as SIFT [13] can be selected if the motion of a robot is assumed to be large, while it sacrifices the computational efficiency. A detector set should be determined in consideration of specific conditions.

In order to avoid excessive switching between detectors, which imposes extra computation, a branch prediction using a saturating counter is employed. The n-bit saturating counter is composed of a finite state machine with 2^n states. The counter increments its count if the current terrain is classified as ROUGH, and decrements if it is classified as SMOOTH. The saturating property means the further increment (decrement) has no effect when the counter reaches to its maximum (minimum) value. The detector is selected depending on the highest bit of the counter (0/1). This technique is simple but known to generate good estimation with less time and memory usage. Based on the experiments, a two-bit counter which has four states produces the best estimate and efficiency.

3.2 Motion Estimation

In VO systems, camera motion between frames is estimated with a set of associated feature points. Data association is one of the challenging problems, and the associated features between frames inevitably suffer from wrong matches. In order to reduce the impact of wrong matches and estimate motion accurately, outlier rejection schemes are commonly employed. Among several known schemes, the RANSAC [5] is used in many VO systems [16, 12]. It is a hypothesis-and-test framework which produces hypotheses with randomly sampled data sets and selects the hypothesis that acquires highest consensus with other data.

Fig. 4 The number of required RANSAC iterations w.r.t. the number of data points $(p = 99\%, \varepsilon = 50\%)$



According to [5, 21], the number of RANSAC iterations that assure a certain level of correct solutions can be found is computed by

$$K_s = \frac{\log(1-p)}{\log(1-(1-\varepsilon)^s)} \tag{5}$$

where p is the requested probability of success, ε is the percentage of outliers in the dataset, and s is the number of data points which can be used for model estimation. Fig. 4 shows the number of iterations increases exponentially with the number of data points s. In the other words, if the number of required data can be reduced the computational efficiency will drastically be improved.

Theoretically, to solve the relative pose problem of a stereo camera rig, the availability of three point correspondences is required. The proposed algorithm, however, uses only two point correspondences. Reducing the number of data points requires another cue to fully recover pose parameters. The main approach is to use knowledge about the common direction which is estimated from inertia measurements or vanishing points [14, 11, 6]. However, those measurements are not easily obtained by mobile robots in uneven terrain. The proposed method employes an alternative obtained from the correspondence of a feature point in the distance.

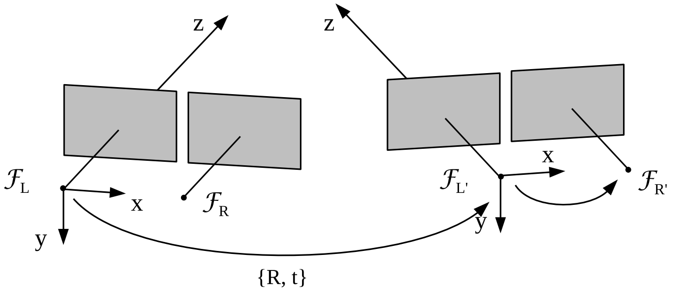


Fig. 5 Coordinate frames with two poses of a stereo camera

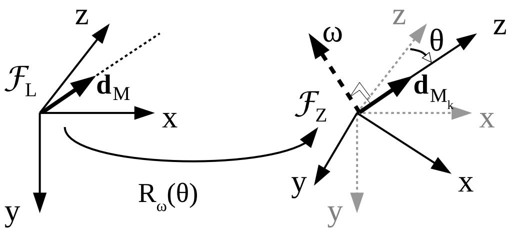


Fig. 6 Transformation to intermediate frame

At first, a distant point correspondence recovers two rotation parameters. In Fig. 5, two poses are associated with a rigid transformation

$$\mathbf{X}' = \mathsf{R}\,(\mathbf{X} - \mathbf{t}) + \mathbf{N}(z) \tag{6}$$

with a distance-dependent noise vector $\mathbf{N}(z)$. Let $\mathbf{d}_M$ be the unit vector toward the corresponding distant point. Introducing the intermediate frame $\underset{\longrightarrow}{\mathscr{F}_Z}$ where $\mathbf{d}_M = \mathbf{e}_z$ (see Fig. 6), the axes of frame $\underset{\longrightarrow}{\mathscr{F}_L}$ is aligned to $\underset{\longrightarrow}{\mathscr{F}_Z}$ by the following rotation

$$\mathsf{R}_\omega(\theta) = \mathbf{1} + [\omega]_\times \sin\theta + (1 - \cos\theta)[\omega]_\times^2 \tag{7}$$

where

$$\omega = \mathbf{e}_z \times \mathbf{d}_M \tag{8}$$

$$\theta = \arccos(\mathbf{e}_z \cdot \mathbf{d}_M)\,. \tag{9}$$

If the point is sufficiently far, the z-axis of $\underset{\longrightarrow}{\mathscr{F}_Z}$ and $\underset{\longrightarrow}{\mathscr{F}_{Z'}}$ are approximately aligned. The rotation matrix between two poses can then be expressed as

$$\mathsf{R} = \mathsf{R}_{\omega'}(\theta')^\top \mathsf{R}_{\mathbf{e}_z}(\phi)\mathsf{R}_\omega(\theta) \tag{10}$$

where ϕ is the rotation angle for solving the rotational ambiguity around z-axis.

Next, the remaining parameters are solved by a near point correspondence. In (6), $\mathbf{N}(z)$ can be negligible where the point is close to the camera. Thus, the translation is computed as

$$\mathbf{t} \approx \mathbf{X} - \mathsf{R}^\top \mathbf{X}'. \tag{11}$$

From (10) and (11), the following equations

$$\mathbf{t}_Z = \mathbf{X}_Z - \mathsf{R}_{\mathbf{e}_z}(\phi)^\top \mathbf{X}'_Z \tag{12}$$

can be deduced, which are three equations with four unknown parameters. Notice that the vectors are transformed to the intermediate frames. By introducing the epipolar constraint

$$\mathbf{m}_Z'^\top [\mathbf{t}_Z]_\times \mathsf{R}_{\mathbf{e}_z}(\psi)\mathbf{m}_Z = 0 \tag{13}$$

all unknown parameters can be recovered as one of two roots of polynomial systems.

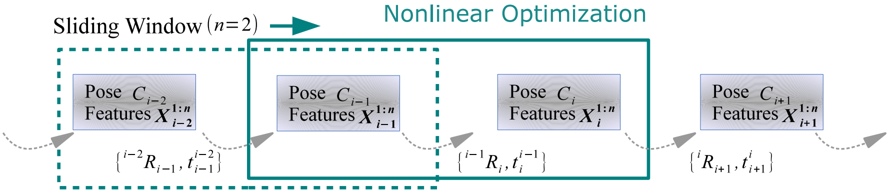


Fig. 7 Sliding window SBA (window size= 2)

3.3 *Local Optimization*

The motion parameters recovered with the two-point algorithm contain small errors. A dominant reason is the approximated alignment between the vectors towards a point at a finite distance, which is introduced in (10). If a distant point is sufficiently far, the approximation almost meets. However, points at infinity are hard to be found in real outdoor environments, and the small directional difference generates an estimation error. Additionally, improper camera calibration or wrong matches may also be the causes of inaccurate estimation. As VO estimates the current pose by concatenating transformations at each time, the small error accumulates quickly.

The accumulated error can be mitigated by optimization techniques. The sparse bundle adjustment (SBA) is an efficient solution for the pose graph optimization [23]. In the developed system, the sliding window SBA shown in Fig. 7 is employed. It optimizes the parameters in a fixed-size image window and slides it as the new pose is added.

The procedure of employed sliding window SBA is as follows. Firstly, the pose graph is generated. The new pose and feature measurements are added to the graph if the motion from the previous pose exceeds a certain threshold. Then, nonlinear optimization such as Levenberg-Marquardt is performed for the last w poses, where w denotes the window size. The window is shifted while the graph grows, and the poses are locally optimized in the whole graph.

3.4 *Algorithm Summary*

A summary of the complete algorithm is given as follows:

1. Capture images with a stereo camera rig.
2. Apply terrain classification to select the appropriate detector.
3. Extract features and descriptors for both images.
4. Triangulate feature points in 3D space.
5. Match features between frames using feature description.
6. Perform the two-point algorithm with RANSAC to compute the relative pose from the previous pose.
7. Validate the estimated pose using the pose constraint of the robot (e.g., direction of gravity, ground height).

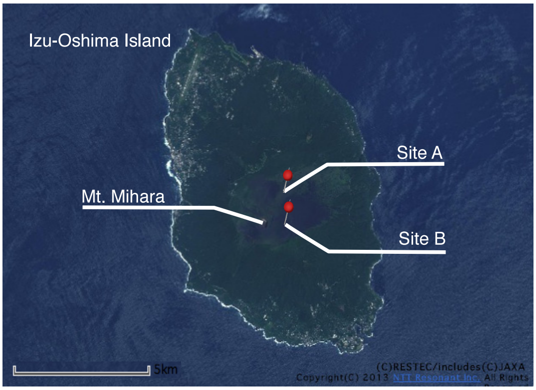
8. Concatenate poses. If the motion exceeds the predetermined threshold, perform sliding window SBA to reduce accumulated errors.

4 Experiments and Results

Field experiments have been conducted to evaluate the proposed VO method. The experimental field, Urasabaku Desert, is located at the center of Izu-Oshima Island in Tokyo, Japan (Fig. 8). The terrain is covered with scoriae, dark and bright colored volcanic rocks, and volcanic bombs on large uneven slopes.

The images are taken by the M6A rover, which is equipped with a stereo camera rig at the height of 1.5 m from the ground, a GPS receiver, and an AHRS (Attitude and Heading Reference System). Images are captured with Point Grey Dragonfly2 CCD cameras and Tamron M13VG308 wide-angle lenses. The intrinsic and extrinsic parameters of a stereo camera is preliminary calibrated using the method of Zhang [27].

The entire system has been integrated into ROS (Robot Operating System) and coded with C++. The visualization of a robot and a path is shown in Fig. 9. The analysis is performed on a laptop PC with Intel Core 2 Quad 2667MHz CPU using only a single core.


Izu-Oshima Island
Site A
Mt. Mihara
Site B
5km
(C)RESTEC/includes(C)JAXA
Copyright(C) 2013 NTT Resonant Inc. All Rights

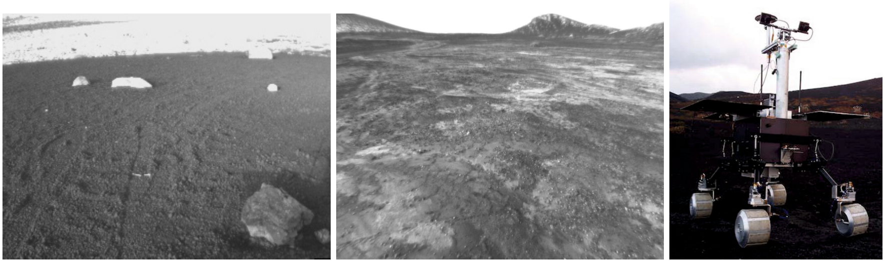
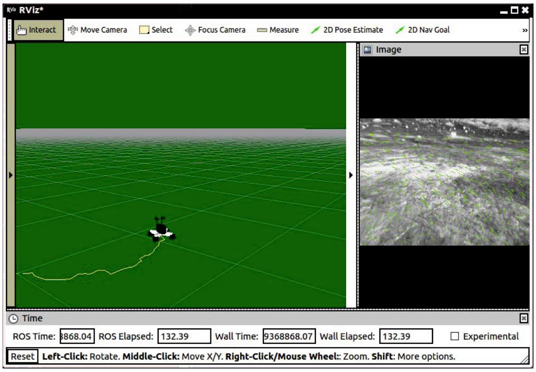


Fig. 8 Experimental areas. Top: Site map of Izu-Oshima Island. Bottom left: Sample Image for Site A. Bottom middle: Sample Image for Site B. Bottom right: Experimental robot M6A.



ROS Time: 868.04 ROS Elapsed: 132.39 Wall Time: 9368868.07 Wall Elapsed: 132.39 □ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift:** More options.

Fig. 9 The window image for visualizing path

4.1 Feature Extraction

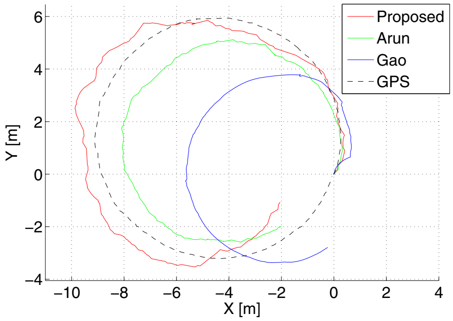
The VO systems in feature-less environments such as these volcanic fields encounter the first and most challenging problem at the extraction of features. The above mentioned hybrid detector selects the well-tuned appropriate detector for feature-rich/poor scenes (Refer to Fig. 2) so that enough features can be extracted from both environments. The scene classifier is trained with hundreds of labeled data for each class using the one-versus-rest technique. The patch classification accuracy is around 70–80%. The slightly low classification rate can be mitigated with the voting procedure in a whole image.

The proposed hybrid detector is compared with common detectors including Harris [10], Shi-Tomasi [22], FAST [18, 19], DoG (Difference of Gaussians) [13], Fast Hessian [3], and STAR based on CenSurE [1]. The result is summarized in Table 1. Two combinations for the hybrid detectors are shown: Harris+FAST and DoG+FAST. The notable improvement is the average number of successful feature tracking for the hybrid detector with Harris and FAST. It also shows good performance in extraction time due to the combination with high-speed FAST detector. It successfully benefitted from both Harris and FAST.

Although the hybrid detector additionally imposes the extra computational cost for terrain classification, it only takes 0.39 ms for 4x4 blocks in a 320x240 resolution image. This is due to the efficient AdaBoost algorithm and the design of terrain descriptor with four simple statistical values describing the terrain well.

Table 1 Performance comparison in feature extraction (961 frames, 320x240 grayscale images)

Detector	Inlier Count	Success Frames	Extract Time	Class. Time
Harris	47.27 (9.8%)	97.3%	11.87 ms	–
Shi-Tomashi	43.21 (4.6%)	93.1%	14.36 ms	–
FAST	33.73 (21.1%)	25.3%	1.32 ms	–
DoG	31.13 (10.4%)	75.5%	54.98 ms	–
Fast Hessian	31.16 (12.1%)	39.2%	27.90 ms	–
STAR	18.61 (23.5%)	28.3%	9.86 ms	–
Proposed (Harris+FAST)	56.93 (12.1%)	87.7%	9.24 ms	0.39 ms
Proposed (DoG+FAST)	45.43 (14.3%)	54.4%	39.34 ms	0.39 ms



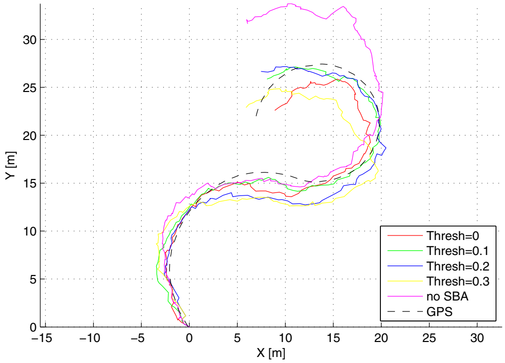
Method	Time	RMS Error
Arun [2]	0.18 ms	1.061 m (3.16%)
Gao [8]	1.32 ms	2.757 m (8.23%)
Proposed	0.12 ms	1.357 m (4.05%)

Fig. 10 Comparison of motion estimation methods. Left: Estimated trajectories. Right: Average time and RMS errors in meters. (Arun: 3D-to-3D least squares method [2] using four points. Gao: 3D-to-2D P3P method [8] using three points.)

4.2 *Pose Recovery*

Using the tracked features, motion is estimated between frames. The two-point algorithm is compared with conventional three-point algorithms [2] and [8], all of which are implemented in C++. The trajectories are estimated in Fig. 10 along a 40 m drive on a circular path. The motion is estimated within the RANSAC framework. The two-point algorithm successfully recovers a circular path while it contains small fluctuation in yaw direction. The fluctuation is derived from the approximation using points at finite distance. Even so, this formulation recovers trajectory with sufficiently small RMS error and time.

Moreover, the VO system was tested for longer trajectories. The sliding window SBA approach optimizes the poses within w-window ($w = 4$ in this experiment). If a motion threshold is set, the motion between adjacent images inside the window should exceed the threshold. Exceptions occur if the number of feature tracking drops less than a threshold due to a sudden view change or bad image conditions.



Motion Thresh.	RMS Error
0.0 m	2.262 m (3.98%)
0.1 m	2.438 m (4.29%)
0.2 m	2.505 m (4.41%)
0.3 m	2.483 m (4.36%)
no SBA	3.798 m (6.67%)

Fig. 11 Estimated trajectories for longer traverse at varying motion thresholds. Left: Estimated trajectories. Right: RMS errors in meters.

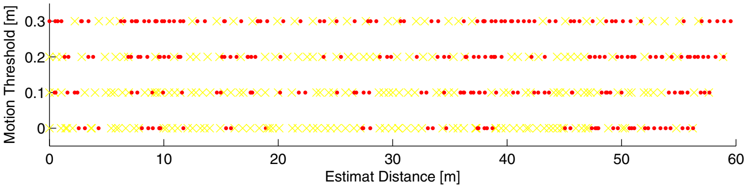


Fig. 12 The frequency of SBA process at varying motion thresholds. Each dot represents the execution of SBA. Yellow: Normal process. Red: Few feature tracking detected.

In Fig. 11, the trajectories for a 60 m path are estimated at varying motion thresholds for sliding window SBA. By introducing SBA, the error accumulation is reduced. Smaller motion thresholds show better performance in error reduction. However, determining the motion threshold is not a simple problem. It is not preferable to perform SBA in every frame since it consumes the resources. On the other hand, with larger thresholds the feature tracker may fall into failure. Fig. 12 presents the timing that SBA is performed for the motion thresholds 0.0, 0.1, 0.2, 0.3 m. Yellow dots represent normal process (reaching the threshold), while red dots show the case where the failure of feature tracking is detected.

5 Conclusion

Although estimating camera poses in natural environments is a difficult task, the proposed techniques enabled pose estimation by VO even in challenging feature-less scenes. One crucial problem of VO in such scenes is that the terrain does not exhibit rich visual features. The lack of feature tracking degrades the accuracy and in some case causes the failure of pose estimation. Considering there does not exist a ‘single’ method to extract interest points in ‘any’ situation, the proposed approach makes use of the different properties of multiple detecters. The hybrid detector selects the most appropriate detector in accordance with the terrain classification by an efficient machine learning technique.

Moreover, a novel motion estimation method was proposed to efficiently recover six motion parameters with smaller data points. In theory, the minimal number for recovering an unconstraint motion is three. It is reduced by the proposed algorithm which only requires one point from near region and another point from far region. Since it is difficult to extract feature points from near region in feature-less terrain, the strategy of point selection will be able to reduce missed frames. The two-point algorithm reduced the number of data points by exploiting common direction between poses. The common direction is computed by the direction to the point at a finite distance which is obtained by triangulation. Using the two-point algorithm with RANSAC accelerated the estimation process by reducing iterative operations.

The proposed VO algorithm was evaluated with real data, which was obtained by the developed testbed robot at the volcanic fields in Izu-Oshima, Japan. The algorithm was validated by the image sequences of several drives. The estimated motion contained a small yaw error caused by the approximation in formulation, while the error can be mitigated with SBA approaches. The future work involves the performance evaluation with longer (multi-km) paths in untextured environments.

References

1. Agrawal, M., Konolige, K., Blas, M.R.: CenSurE: Center surround extremas for realtime feature detection and matching. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 102–115. Springer, Heidelberg (2008)
2. Arun, K., Huang, T., Blostein, S.: Least-Squares Fitting of Two 3-D Point Sets. IEEE Transactions on Pattern Analysis and Machine Intelligence 9(5), 698–700 (1987)
3. Bay, H., Tuytelaars, T., Gool, L.V.: SURF: Speeded up Robust Features. Computer Vision and Image Understanding 110(3), 346–359 (2008)
4. Cheng, Y., Maimone, M.W., Matthies, L.H.: Visual odometry on the Mars Exploration Rovers. In: IEEE Robotics & Automation Magazine, pp. 54–62 (2006)
5. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24(6), 381–395 (1981)
6. Fraundorfer, F., Tanskanen, P., Pollefeys, M.: A minimal case solution to the calibrated relative pose problem for the case of two known orientation angles. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 269–282. Springer, Heidelberg (2010)
7. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer System Sciences 55(1), 119–139 (1997)
8. Gao, X., Hou, X., Tang, J., Cheng, H.: Complete solution classification for the perspective-three-point problem. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(8), 930–943 (2003)
9. Gauglitz, S., Höllerer, T., Turk, M.: Evaluation of interest point detectors and feature descriptors for visual tracking. International Journal of Computer Vision 94(3), 335–360 (2011)

10. Harris, C., Stephens, M.: A combined corner and edge detector. In: Alvey Vision Conference, Manchester, UK, vol. 15, pp. 147–151 (1988)
11. Kalantari, M., Hashemi, A., Jung, F., Guedon, J.P.: A New Solution to the Relative Orientation Problem Using Only 3 Points and the Vertical Direction. Journal of Mathematical Imaging and Vision 39(3), 259–268 (2011)
12. Konolige, K., Agrawal, M., Solà, J.: Large-scale visual odometry for rough terrain. In: Kaneko, M., Nakamura, Y. (eds.) Robotics Research. STAR, vol. 66, pp. 201–212. Springer, Heidelberg (2010)
13. Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
14. Naroditsky, O., Zhou, X.S., Gallier, J., Roumeliotis, S.I., Daniilidis, K.: Two efficient solutions for visual odometry using directional correspondence. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(4), 818–824 (2012)
15. Nistér, D., Naroditsky, O., Bergen, J.: Visual odometry. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 652–659 (2004)
16. Nistér, D., Naroditsky, O., Bergen, J.: Visual odometry for ground vehicle applications. Journal of Field Robotics 23(1), 3–20 (2006)
17. Otsu, K., Otsuki, M., Ishigami, G., Kubota, T.: Terrain adaptive detector selection for visual odometry in natural scenes. Advanced Robotics 27(18), 1465–1476 (2013)
18. Rosten, E., Drummond, T.: Fusing points and lines for high performance tracking. In: The 10th IEEE International Conference on Computer Vision, vol. 2, pp. 1508–1515 (2005)
19. Rosten, E., Drummond, T.W.: Machine learning for high-speed corner detection. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part I. LNCS, vol. 3951, pp. 430–443. Springer, Heidelberg (2006)
20. Saiki, K.: Selection and management of a test site for volcano-observation robots (in Japanese). Journal of the Japanese Society for Planetary Sciences 21(2), 94–102 (2012)
21. Scaramuzza, D.: Performance evaluation of 1-point-RANSAC visual odometry. Journal of Field Robotics 28(5), 792–811 (2011)
22. Shi, J., Tomasi, C.: Good features to track. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 593–600 (1994)
23. Sünderhauf, N., Konolige, K., Lacroix, S., Protzel, P.: Visual odometry using sparse bundle adjustment on an autonomous outdoor vehicle. In: Autonome Mobile Systeme, pp. 157–163 (2005)
24. Tamura, Y., Suzuki, M., Ishii, A., Kuroda, Y.: Visual odometry with effective feature sampling for untextured outdoor environment. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3492–3497 (2009)
25. Tomono, M.: Robust 3D SLAM with a stereo camera based on an edge-point ICP algorithm. In: IEEE International Conference on Robotics and Automation, pp. 4306–4311 (2009)
26. Williams, S., Howard, A.M.: Developing monocular visual pose estimation for arctic environments. Journal of Field Robotics 27(2), 145–157 (2009)
27. Zhang, Z.: A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(11), 1330–1334 (2000)

Eight Weeks of Episodic Visual Navigation Inside a Non-stationary Environment Using Adaptive Spherical Views[*]

Feras Dayoub, Grzegorz Cielniak, and Tom Duckett

Abstract. This paper presents a long-term experiment where a mobile robot uses adaptive spherical views to localize itself and navigate inside a non-stationary office environment. The office contains seven members of staff and experiences a continuous change in its appearance over time due to their daily activities. The experiment runs as an episodic navigation task in the office over a period of eight weeks. The spherical views are stored in the nodes of a pose graph and they are updated in response to the changes in the environment. The updating mechanism is inspired by the concepts of long- and short-term memories. The experimental evaluation is done using three performance metrics which evaluate the quality of both the adaptive spherical views and the navigation over time.

1 Introduction

Functional and useful mobile service robots require the ability to share physical spaces with humans, and need to deal with a dynamic and ever-changing world. These changes are mainly caused by human activities making them spontaneous, discontinuous and unpredictable. This includes changes in the structure of the environment as well as its appearance (e.g. rearrangement of the furniture or changing the colour of a curtain).

In order to maintain an up-to-date inner representation of the world, robots can use their continuous stream of sensory information, which reflects the momentary status of their surroundings. However, the amount of sensory information to be processed in a lifetime is vast; therefore, efficient methods are required for filtering, storing and updating this information over time.

Feras Dayoub
CyPhy lab, Queensland University of Technology, Brisbane, Australia
e-mail: feras.dayoub@qut.edu.au

Grzegorz Cielniak · Tom Duckett
University of Lincoln, Lincoln, UK

[*] An extended abstract of this paper appeared as [6]

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_26, © Springer International Publishing Switzerland 2015

One possible solution for handling this large amount of sensory information can be inspired by the concepts of short- and long-term stores of the human memory. While a robotic memory need not be constrained by the fallibilities of human memory nor the exact details of its biological implementation, we believe that the modal model of human memory provides a useful framework for the filtering and storage of perceptual information in artificial agents such as robots.

This paper uses a long-term map-updating mechanism inspired by the multi-store model of human memory [7] for the application of visual navigation. The map consists of an adjacency graph of nodes on a global level, and each node on the local level of the map represents a spherical view of image features extracted from an omnidirectional image of the node. The spherical views provide both an appearance signature for the nodes, which the robot uses to localise itself in the environment, and heading information when the robot uses the map for visual navigation. The paper presents an evaluation of the navigation performance within a typical office environment over a period of eight weeks. These metrics are used to evaluate the effect of the learning and forgetting processes on the quality of the map over time.

The rest of the paper is structured as follows. Section 2 discusses related work. Section 3 presents an overview of the proposed memory model, Section 4 describes our method for long-term adaptation and visual navigation. Section 5 discusses the performance evaluation metrics. Section 6 presents the experiments and results obtained. Finally we draw conclusions in Section 7.

2 Related Work

Although nearly every actual robot real-life environment is dynamic, the majority of previous work on robotic mapping assumes that the world is static. Whereas, most previous approaches that consider mapping dynamic environments assume that the underlying structure of the environment is static, and then try to separate moving objects from stationary parts by treating the dynamic effects as measurement outliers [8, 12, 10]. Alternatively, many authors try to improve the robustness of the mapping process by detecting and tracking moving objects separately [18, 15, 13].

Considering that the environment consists of static and dynamic objects, other approaches build two maps, one for the static parts and one for dynamic elements. The complete state of the environment is obtained by merging the two maps [19]. Other approaches try to maintain one map for both dynamic and static landmarks, by classifying landmarks as dynamic or stationary using a probabilistic framework. Movements of the dynamic landmarks are observed and included in the estimation process of the map [4].

Several authors have investigated mapping systems that incorporate simple forgetting mechanisms based on recency weighting. Andrade-Cetto and Sanfeliu [1] developed an EKF-based mapping system that is able to forget landmarks that have disappeared, where an existence state associated with each landmark measures how often it has been seen. However, none of these methods are general enough to handle environmental changes occurring at different rates, nor has the long-term robustness of these approaches been demonstrated in real world environments. Using a map learning and forgetting mechanism, Biber and Duckett [5] introduced an approach to update a laser based map which represents the environment at different timescales, with older memories fading at different rates. They used samples and robust statistics to handle noise and contradicting measurements produced by environment changes.

Very recently [17] proposed a method to update a laser-based metric map. The method uses a pose graph SLAM approach to optimise the trajectory of the robot and produce the map. In order to update the map, the robot compares its current laser scan view with scans stored from previous passes through the same sections of the environment. The author makes the assumption that the environment contains only low dynamic objects, i.e. objects that move only outside of the robot's view, which makes the environment static when the robot is passing through it. However, in a dynamic environment, changes can occur while the robot is operating inside the environment. As such, these changes need to be detected and filtered out.

The main aspect of the previous works on vision-based navigation that is superseded by our approach is the ability to adapt and maintain only one reference view for each place in the robot's map in response to environmental changes instead of keeping a history of multiple views to represent the same place over time.

3 An Overview of the Memory Model

According to the basic model of Atkinson and Shiffrin [2], human memory is divided into separate stores for sensory memory (SM), short term memory (STM) and long term memory (LTM). The sensory memory contains information perceived by the senses, and selective attention determines what information moves from sensory

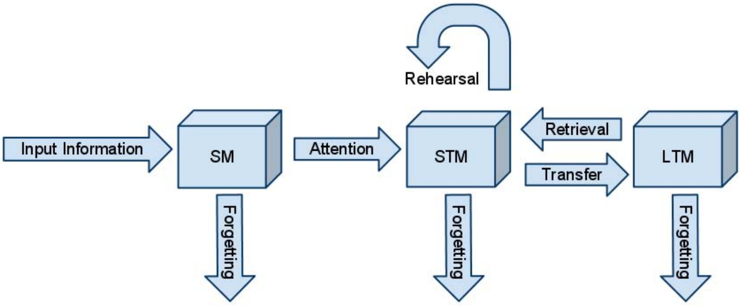


Fig. 1 Multi-store memory model. SM: Sensory memory. STM: Short-term memory. LTM: Long-term memory. Selective attention, which involves the LTM, determines what information moves from SM to STM. Through the process of rehearsal, information in STM can be transferred to LTM and be retained for longer periods of time. Information from the LTM store is retrieved using a process called "recall".

memory to short-term memory. Through the process of rehearsal, information in STM can be committed to LTM to be retained for longer periods of time. In return, the knowledge stored in LTM affects our perception of the world, and influences what information we attend to in the environment. In our approach, perceptual attention includes detection of local image features for subsequent processing in the memory model. (see Fig. 1).

The concepts of the above memory model are used to update the map, incrementally, by gradually adding information about new stable image features in the environment, while removing information about features that no longer exist. The sensory memory contains the features extracted from the current image. Then an attentional mechanism selects which information to move to STM, which is used as an intermediate store where new image features are kept for a short time. Over this time the system uses a rehearsal mechanism to select features that are more stable for transfer to LTM. In order to limit the overall storage requirements and adapt to changes in the environment, the system also contains a recall mechanism that forgets (i.e. removes) unused feature points in LTM. LTM is used in turn by the attentional mechanism for selecting the new image features to update the map.

3.1 Map Representation

The robot's world is represented as a hybrid map consisting of two levels, global and local. Fig. 2 illustrates the hybrid map. On the global level, the world is represented as n optimized pose-graph. On the local level of the map, each node stores a spherical view representation of image features. The spherical views contain the 3D location of the image features on a sphere, so only the directions of the features (but not their distance or depth) from the centre of the sphere are stored. The centre of the sphere in this case corresponds to the centre of that node.

Each spherical view is initialised from an omnidirectional image recorded from the centre of each node in the global map. The spherical representation of the nodes creates a connection between the global and local levels of the map, where the group of image features is used as a qualitative descriptor for localisation on the global level, and the 3D location of these features on the sphere is used for estimating the heading needed for the navigation system at the local level [7].

Localisation on the global level is achieved by using an image similarity score based on the number of matched feature points between the current view of the robot and the group of points stored in each node. The robot localises itself to the node which has the highest similarity score in the map. Navigation on the local level is done by using multiple view geometry for spherical views to estimate the robot's heading during autonomous navigation. This navigation method is described in Section 4.3. The same image features used for navigation are also used to update the spherical views stored inside each node over time, in response to changes in the appearance of the environment.

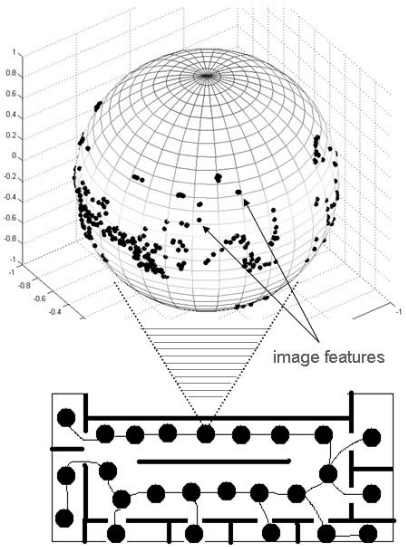


Fig. 2 The environment is represented as an adjacency graph of nodes on a topological level and each node on the metric level of the map represents the 3D location of image features on a unit sphere. Our method represents the direction of the features from the centre of the sphere, which corresponds to the centre of that node.

4 Methodology

4.1 Map Updating Mechanism

We represent STM and LTM as finite state machines (see Fig. 3), where each memory type consists of a set of states (S_i). There is one STM and one LTM associated with each node of the map that stores information about features. The LTM represents the recent stable configuration of features in the environment and these are the features that are used as reference views of the map. The rehearsal process for a stored feature in STM is the process of continually recalling information into the STM in order to memorise it. In order to transfer a feature point from STM to LTM the feature has to be seen frequently. Features enter STM from sensory memory and must progress through several intermediate states (S_1 to S_n) before transfer to LTM. Every time the robot finds the feature ("detect"), the state of the feature is moved closer to LTM. However if the feature is missing from the current view ("miss"), it is returned to the first state (S_1) or forgotten if it is already there. This policy means that spurious features should be quickly forgotten, while persistent features will be transferred to LTM. The recall process for a stored feature in LTM first involves updating the LTM by a process of feature matching. In order to remain in the LTM, a feature has to be seen occasionally. In contrast to rehearsal, features enter LTM from

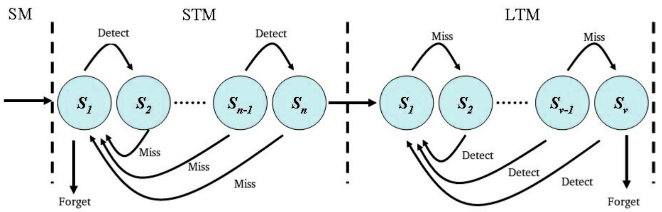


Fig. 3 The proposed multi-store memory model. SM: Sensory memory. STM: Short-term memory. LTM: Long-term memory.

STM and must progress through several intermediate states (S_1 to S_v) before being forgotten. Stored features which have been seen in the current view are reset to the first state (S_1), while the state of features which have not been seen is progressed, and a feature point that passes through all states without a "detect" is forgotten. Finally, recall returns the list of new features that were not already present in the LTM (i.e. the difference in appearance between the current and reference views). We use multiple view geometry to transfer the image features from the current view to the spherical views of the nodes. The geometric method by which we update the spherical views of the map is presented in full detail in [7].

4.2 Temporal Calibration for the Updating Mechanism

Temporal calibration means selecting the real-time unit in which the robot uses the memory system to update its map. In this work, it is assumed that the system is used by a mobile service robot working inside a house or public environment, where life is a series of daily episodes. This suggests that using days as a basic time unit would be a realistic choice. After each working day, during which it spends its time navigating inside the environment and visiting different nodes inside the map, the robot goes through what could be called a "sleep" period where it activates its memory system to update the map. However, in other situations where life and human activities do not follow daily cycles, the robot can adopt a time scale to update the map which reflects the natural cycle of activities in its surroundings.

Depending on the nature of the task, the robot may visit some nodes in the map multiple times during one day, whereas other nodes will be visited less frequently. This means that it is important to unify the rate at which the appearance of each node is updated. In order to achieve this aim, the robot selects, at the end of each working day and for each visited node in the map, one view only to use for map updating. The selected view is the one which has the highest number of matched points with the reference view of each visited node over the whole day.

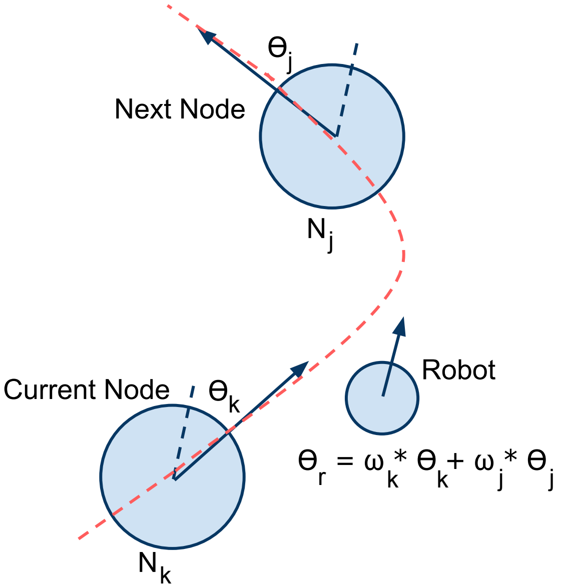


Fig. 4 The proposed navigation strategy using heading estimation. The robot is required to re-execute a path consisting of a number of key images which were recorded during a previous mapping stage. In the figure, N_k is the current node in the path and N_j is the next node. The red dashed line is the intended path. θ_k, θ_j are the relative orientations between the robot's heading and the reference orientation of the nodes N_k, N_j respectively. θ_r is the desired heading which results from a weighted sum of θ_k and θ_j.

4.3 Using the Map for Navigation

Every map can be judged by its usefulness for practical purposes. In our case the map is used for a daily path following routine inside a continually changing environment.

When robots work inside an indoor environment, their navigation generally is restricted to what the humans consider to be a path inside that environment, such as corridors and the areas between the furniture. These routes effectively simplify the task of navigation by limiting the robot to only one degree of freedom along the path. And by representing this path as a sequence of images, the following framework of the appearance-based approach for visual navigation is repeatedly used in the literature:

- The path is first built during a learning phase where the robot is controlled by a human operator. During this phase the robot captures a sequence of images along the path.
- A subset of the captured images is selected to represent the reference images along the path.

- During the replay phase, the robot starts near the first position and is required to repeat the same path.
- The robot extracts motion directions by comparing the currently observed image with the reference images along the path.

In this work we adopted a similar framework for visual path following using a sequence of nodes from the map. Fig. 4 illustrates the navigation strategy. First the robot localizes itself to one of the nodes in the path. This is done by selecting the node which has the highest similarity score with the currently observed view. Let S_k be the similarity score, i.e, the number of matched points. The similarity score is also computed between the current view and the next node in the sequence. Let S_j be the similarity score with the next node. After that the following ratio is computed:

$$\omega_k = \frac{S_k}{S_k + S_j}, \ \omega_j = \frac{S_j}{S_k + S_j}. \tag{1}$$

Then the heading angle θ_r is computed as a weighted sum:

$$\theta_r = \omega_k * \theta_k + \omega_j * \theta_j. \tag{2}$$

where θ_k and θ_j are the relative orientation between the current view and the nodes N_k and N_j respectively (see Fig. 4). By following this navigation strategy, the nodes in the path can be considered as directional signs which lead the robot toward its goal.

In order to estimate the relative orientation between two views, such θ_k and θ_j in the above case, we use epipolar geometry to estimates the essential matrix $\mathbf{E}$, which is factored to give a rotation matrix $\mathbf{R} \in SO(3)$ and the skew-symmetric matrix $[\mathbf{t}]_\times$ of a translation vector $\mathbf{t} \in \mathbb{R}^3$ [11] as follows:

$$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}. \tag{3}$$

After that, the relative orientation is extracted from the rotation matrix $\mathbf{R}$.

5 Performance Evaluation

5.1 Map Adaptability

The main goal of the proposed memory model is to make the reference views of the map adapt to the changes in the appearance of the environment over time. In order to measure this adaptability we use the similarity between the views of the nodes and the reference views as a metric. Higher similarity means better representation for the environment where the similarity is measured as the number of matched feature points between two views. We compare how the similarity changes over time when the memory is used to update the reference views and again when the reference views were left static.

5.2 Map Consistency

Map consistency in our case means that the updating process of the reference views, which involves removing and adding image features over time, does not cause the map to degrade. If the map degraded over time, the robot would have difficulties to use the map for tasks such as autonomous visual navigation. Therefore, measuring the performance of executing a visual navigation task over time would be a good indicator of the quality of the map. In other words, the map is considered to be consistent if the performance does not drop over time.

In order to evaluate the performance of the proposed navigation strategy presented in Section 4.3, we use two metrics. The first is the length of the trajectory. If the length of the trajectory increased over time this would mean that the robot took more steps to execute the path due to poor directional information from the map. The second metric is the curvature of the executed trajectory by the robot [14]: the lower the curvature the smoother the trajectory. The smoothness of the trajectory is a good indicator of the consistency of the decision-action relationship in the navigation system. Similar to the first metric, if the curvature of the trajectory increased over time this would indicate that the robot performance is degrading.

Representing the trajectory of the robot as a curve in a 2D plane:

$$y = f(x), \tag{4}$$

the length of this trajectory can be calculated as:

$$L = \sum_{i=1}^{n-1} \sqrt{(x_{i+1} - x_i)^2 + (f(x_{i+1}) - f(x_i))^2}, \tag{5}$$

where $(x_i, f(x_i))$, i=1...n, are the n points of the trajectory in Cartesian coordinates. The curvature of the trajectory at any point can be calculated as:

$$k(x_i) = \frac{f''(x_i)}{[1 + (f'(x_i))^2]^{\frac{3}{2}}}. \tag{6}$$

Using the above curvature factor, the smoothness of the trajectory can be measured as follows:

$$B_E = \frac{1}{n} \sum_{i=1}^{n} k(x_i)^2, \tag{7}$$

where B_E is called the bending energy [16]. The bending energy can be understood as the energy needed to bend a rod to the desired shape. The less the energy the smoother the trajectory.

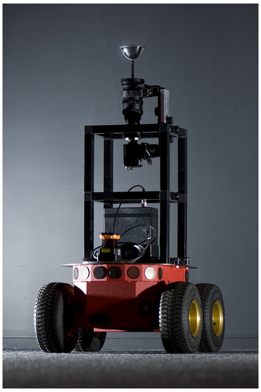


Fig. 5 An ActivMedia P3-AT robot equipped with an omnidirectional vision system.

6 Experiment and Results

Our experimental platform was an ActivMedia P3-AT robot equipped with a GigE progressive camera (Jai TMC-4100GE, 4.2 megapixels) with a curved mirror from 0-360.com. See Fig. 5

The experiment was carried out inside the faculty office at the School of Computer Science at the University of Lincoln. We choose this area to conduct a long-term experiment because the room is designed as open offices where seven members of staff perform their daily activities. These activities result in changes to the appearance of the room over time. On the first day the robot was driven in a loop and a map with 10 nodes was created. For each node in the map, a spherical view of SURF features [3] was built. Using these spherical views, the map was used by the robot to perform a visual navigation routine from node number 1 to node 10. The 10 node route was repeated 38 times over a period of 8 weeks and after each run the robot used the memory model to update the map. Fig. 7 shows three images taken by the robot from the same place but at different times.

In this experiment the robot uses 3 stages in STM and 7 stages in LTM (one week) as the parameters for the memory. In other words, the robot rehearses new information for 3 days before transferring it to the LTM. In the LTM the robot forgets any information which has not been used for a week, taking into account the weekly episodic nature of our daily life.

At the beginning of each run, the robot was placed in the vicinity of the first node. Then the robot performed global localization by matching the extracted feature points from its current view with all the reference views in the map and localized itself to the node with the highest number of matched points. Then the robot estimated its heading as described in Section 4.3. The obstacle avoidance procedure used in this work is as follows. When the robot receives a command to rotate, it checks its sonar range readings first. If the sonar ranges allow for movement, the robot simply executes the movement; otherwise, it turns 10^o in the opposite

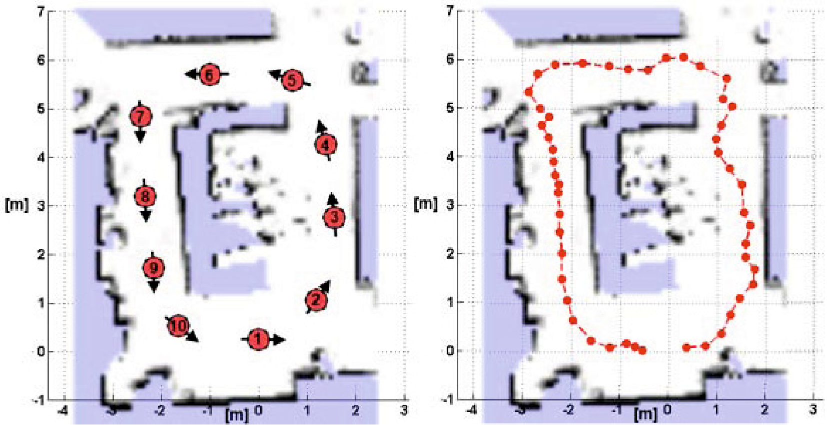


Fig. 6 Left:A laser based occupancy map for the area of the room where the experiments took place. Right: The trajectory of the path taken by the robot at one of the navigation episode during the experiment.

direction and then moves forward for a distance of 50 mm. After that it re-estimates the desired heading using the view from its new position. If both directions are blocked, the robot moves backward 100 mm and then re-estimates the desired heading from its new position. Finally, if the robot receives a command to move forward but there is no room for the movement based on sonar readings, the robot checks the sonar ranges on its right and left sides and then turns in the direction which has the most free space. This procedure is done in a recursive manner.

In order to obtain the ground truth data, we used Laser Range Finder (LRF) sensor with the GMapping library [9]. The GMapping algorithm provides a Simultaneous Localization and Mapping (SLAM) solution for static environments based on a Rao-Blackwellized particle filter. The output of the algorithm is an estimate of the robot trajectory along with an occupancy grid map of the environment. This data provides us with information about the total distance travelled by the robot and the smoothness of the trajectory. Fig. 6 shows a laser-based occupancy map for the area of the room where the experiments took place.

The robot was able to perform the path following task successfully during all runs. As mentioned earlier, we use the similarity metric as an indicator for the adaptability of the map. The mean number of matched points between the view which has the best number of matching points and the reference views of the map was 170.9 ± 84.6 when the static reference views were used for the map and 255.7 ± 92.6 when the adaptive map was used. This result shows the effect of using our memory model in increasing the similarity of the map to the environment. Fig. 8 shows how the similarity score changed over the 38 runs for node number 4 in the map.

The second metric is the change of the length of the trajectory over time. Fig. 9 shows that the length of the trajectory does not increase over time. The mean distance traveled over all runs was $19.9 \pm 0.8\,m$.

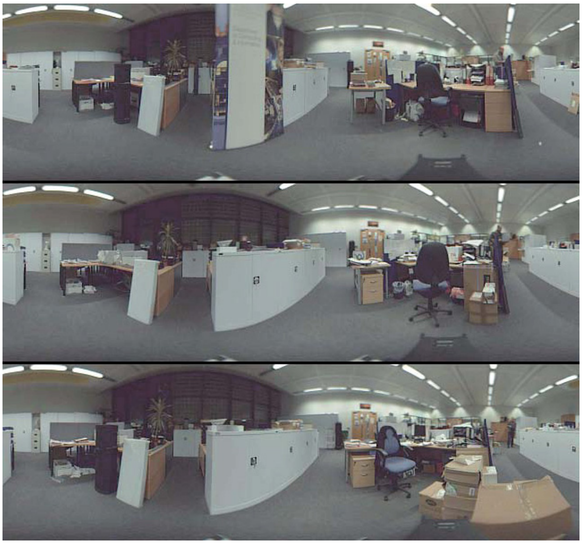


Fig. 7 Three images recorded from the same place at different times. The appearance changes through the existence of new objects in the arena and the disappearance of others.

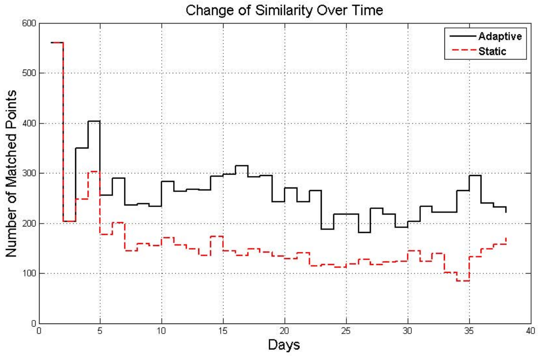


Fig. 8 A comparison between the static and the adaptive map showing the change of the similarity over the 38 runs for node number 4

The third metric used for the evaluation is the smoothness of the trajectory measured by the bending energy. Fig. 10 shows that the bending energy of the trajectory is not increasing over time but it is consistent. This means that the quality of the map is also consistent over time.

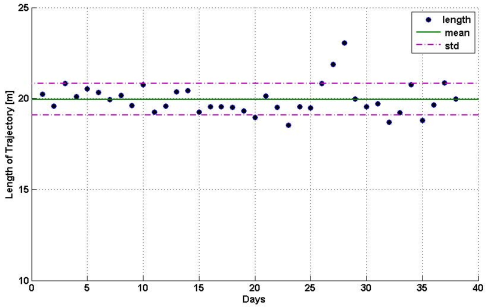


Fig. 9 The change of trajectory length over time. The mean distance travelled over all runs was $19.9 \pm 0.8\ m$. Between days 27 and 28 a big box was delivered into the office, taking up part of the robot's path and forcing it to take a longer trajectory.

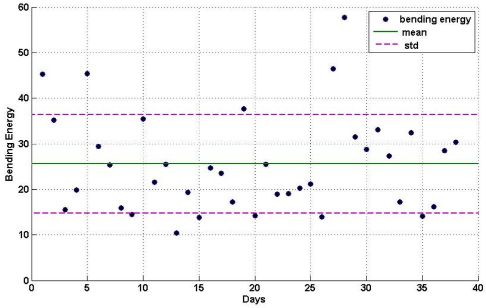


Fig. 10 The change of the smoothness of the trajectory measured by its bending energy. Between days 27 and 28 a big box was delivered into the office which affected the smoothness of the trajectory.

7 Conclusion

This paper presented am eight weeks episodic visual navigation experiment in a real office environment. An updating mechanism, based on short- and long-term memory concepts, incorporates a spherical view representation of image features, is used to keep robot's map up-to-date. The spherical views are used for navigation using multi-view geometry, as well as representing appearance signature of the environment. The results show that the proposed system has a persistent performance in such a real changing environments.

References

1. Andrade-Cetto, J., Sanfeliu, A.: Concurrent map building and localization in indoor dynamic environments. International Journal of Pattern Recognition and Artificial Intelligence 16(3), 361–374 (2002)

2. Atkinson, R., Shiffrin, R.: Human memory: A proposed system and its control processes. In: The Psychology of Learning and Motivation: Advances in Research and Theory, vol. 2, pp. 89–195 (1968)
3. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded Up Robust Features. Computer Vision and Image Understanding 110, 346–359 (2008)
4. Bibby, C., Reid, I.: Simultaneous Localisation and Mapping in Dynamic Environments (SLAMIDE) with Reversible Data Association. In: Proc. Robotics: Science and Systems, Atlanta, GA, USA (2007)
5. Biber, P., Duckett, T.: Dynamic Maps for Long-Term Operation of Mobile Service Robots. In: Proc. Robotics: Science and Systems, Cambridge, MA, USA, pp. 17–24 (2005)
6. Dayoub, F., Cielniak, G., Duckett, T.: Long-term experiment using an adaptive appearance-based map for visual navigation by mobile robots. In: Groß, R., Alboul, L., Melhuish, C., Witkowski, M., Prescott, T.J., Penders, J. (eds.) TAROS 2011. LNCS, vol. 6856, pp. 400–401. Springer, Heidelberg (2011)
7. Dayoub, F., Cielniak, G., Duckett, T.: Long-Term experiments with an adaptive spherical view representation for navigation in changing environments. Robotics and Autonomous Systems 5, 285–295 (2011)
8. Dong, J., Wijesoma, S., Shacklock, A.: Extended rao-blackwellised genetic algorithmic filter SLAM in dynamic environment with raw sensor measurement. In: Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Diego, USA (2007)
9. Grisetti, G., Stachniss, C., Burgard, W.: Improved techniques for grid mapping with rao-blackwellized particle filters. IEEE Transactions on Robotics 23(1), 34–46 (2007)
10. Hahnel, D., Triebel, R., Burgard, W., Thrun, S.: Map building with mobile robots in dynamic environments. In: Proc. IEEE International Conference on Robotics and Automation (ICRA), Taipei, Taiwan, vol. 2, pp. 1557–1563 (2003)
11. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2 edn. Cambridge University Press (2004)
12. Liu, Z., Chen, W., Wang, Y., Wang, J.: Localizability estimation for mobile robots based on probabilistic grid map and its applications to localization. In: 2012 IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), pp. 46–51 (2012), doi:10.1109/MFI.2012.6343051
13. Migliore, D., Rigamonti, R., Marzorati, D., Matteucci, M., Sorrenti, D.G.: Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments. In: Proc. IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan (2009)
14. Muoz Ceballos, N., Valencia, J., Ospina, N.: Performance metrics for robot navigation. In: Electronics, Robotics and Automotive Mechanics Conference (2007)
15. Ortega, S.: Towards visual localization, mapping and moving objects tracking by a mobile robot: a geometric and probabilistic approach. Ph.D. thesis, Institut National Polytechnique de Toulouse (2007)
16. Selekwa, M., Collins, E., Combey, J.: Multivalued Verus univalued Reactive Fuzzy Behavior Systems for Navigation Control of Autonomous Ground Vehicles. In: Proc. Florida Conference on the Recent Advances in Robotics (FCRAR), vol. 20 (2004)
17. Walcott-Bryant, A., Kaess, M., Johannsson, H., Leonard, J.: Dynamic pose graph slam: Long-term mapping in low dynamic environments. In: Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2012)
18. Wang, C.C., Thorpe, C., Thrun, S., Hebert, M., Durrant-Whyte, H.: Simultaneous localization, mapping and moving object tracking. The International Journal of Robotics Research 26(9), 889 (2007)
19. Wolf, D.F., Sukhatme, G.S.: Mobile robot simultaneous localization and mapping in dynamic environments. Autonomous Robots 19(1), 53–65 (2005)

Part VIII
Domestic Robots

Human Activity Recognition for Domestic Robots

Lasitha Piyathilaka and Sarath Kodagoda

Abstract. Capabilities of domestic service robots could be further improved, if the robot is equipped with an ability to recognize activities performed by humans in its sensory range. For example in a simple scenario a floor cleaning robot can vacuum the kitchen floor after recognizing human activity "cooking in the kitchen". Most of the complex human activities can be sub divided into simple activities which can later used for recognize complex activities. Activities like "take meditation" can be sub divided into simple activities like "opening pill container" and "drinking water". However, even recognizing simple activities are highly challenging due to the similarities between some inter activities and dissimilarities of intra activities which are performed by different people, body poses and orientations. Even a simple human activity like "drinking water" can be performed while the subject is in different body poses like sitting, standing or walking. Therefore building machine learning techniques to recognize human activities with such complexities is non trivial. To address this issue, we propose a human activity recognition technique that uses 3D skeleton features produced by a depth camera. The algorithm incorporates importance weights for skeleton 3D joints according to the activity being performed. This allows the algorithm to ignore the confusing or irrelevant features while relying on informative features. Later these joints were ensembled together to train Dynamic Bayesian Networks (DBN), which is then used to infer human activities based on likelihoods. The proposed activity recognition technique is tested on a publicly available dataset and UTS experiments with overall accuracies of 85% and 90%.

1 Introduction

Recent advancements in robotics technologies have introduced low cost domestic robots that can vacuum the floor while residents are away or provide company for

Lasitha Piyathilaka · Sarath Kodagoda
Centre for Autonomous Systems, University of Technology, Sydeny, Australia
e-mail: {Jayaweera.M.Piyathilaka,Sarath.Kodagoda}@uts.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_27, © Springer International Publishing Switzerland 2015

less mobile or elderly people. It is argued that the success of such domestic service robots can be significantly enhanced by the ability of robots to understand the human activities and to respond them accordingly. Such capabilities will enable robots to make more human like decisions without explicitly being ordered to carry out a certain task. It will also allow the robot to seamlessly integrate with human interactions.

Our research focus is to develop robotic technologies to help and promote independent living for elderly people. It is motivated by the growing number of older people around the world and difficulty of finding enough care staff. In general elderly people gradually lose their cognitive ability to keep track of daily activities. In this context, an assistive robot that can recognize human daily activities will be immensely helpful. For example, an elderly person could be reminded of taking medications in appropriate times and could follow it up until the activity has been completed. In addition, the robot may detect abnormal conditions such as someone laying on the floor or sleeping longer than usual and notify the appropriate personnel.

Detection of human activities is challenging due to several reasons. The first reason is related to noisy sensory inputs, and the second reason is related to the difficulty of modeling highly ambiguous actions. Moreover human activities are performed in different body poses and orientations with inter subject variations. Therefore, video-based human action recognition has unwarranted complexity and limited accuracy.

Recent trend in human activity recognition research is to use low cost RGB-D cameras like Microsoft Kinect^{TM}. These cameras are capable of generating skeleton model of a human with 15 body joints positions and their orientation. In this research our intention is to use these skeleton features to extract relatively unambiguous features to model human activities.

In our previous work [9], we have developed human activity recognition model that used Gaussian mixture based HMM. However, its recognition accuracy is severely compromised, if the actions are performed with different body poses. For example "drinking water" activity can be performed while the person is in different body poses such as sitting, standing or even while walking. This is due to the incorporation of all the features, including non informative and ambiguous ones. However, if we could devise a methodology for identifying the most informative features for a given activity, then it will be better positioned at handling actions done with different body poses.

This paper presents a novel human action recognition approach that uses only 3D skeleton features produced by a depth camera. Each activity was modelled as a Dynamic Bayesian Network (DBN) in which each joint node is probabilistically weighted according to the importance of that joint to the activity being modelled. These joint weights together with their observation probability ensembles, form a model for each activity. Joint weights are calculated by training HMM for each case of a given activity and estimating the dissimilarity measure between such trained models. The model is firstly evaluated on a publicly available benchmark dataset: Cornell activity Detection Dataset [11]. Then it was tested with our experiments which shows that proposed method is able to achieve higher recognition accuracies even with higher intra-activity variations of 3D skeleton features.

2 Related Works

Human activity detection is not a new research area that has been looked into by various researchers. In [12] human activities are classified as either normal or aggressive by using a mobile robot and a 3D sensory tracker system. Other researchers have utilized human activity detection to learn and imitate humans activities [2][5]. In [4], audio-based human activity recognition using non-markovian ensemble voting technique is presented. Applicability of this method is limited by the inherent distinguishable sounds associated with activities. Therefore such a system may only be used as a complement to the existing sensory systems.

It is common knowledge that knowing the 3D joint position is helpful for activity recognition. Multi-camera motion capture (MOCap) systems [13] has also been used for activity detection but requires markers attached to joints with a highly calibrated camera system. Therefore, such a system is infeasible to be used in practical robotic scenarios. With the invention of low cost depth cameras, several researchers have used RGB-D skeleton data to recognize activities. In [11] two-layered maximum entropy Markov model with a set of sub-activities is used to detect human activities. There, both the skeleton and 3D point cloud data are used extracting 715 features. However, the algorithm is heavily dependent on a particular sequence of sub activities to form human activities. This can have adverse influence on the generalization aspect due to the individual differences in carrying out activities.

In [13] actionlet ensemble model for human activity detection with depth cameras are proposed. The actionlets are proposed to compensate intra-class variations caused by human activities. This approach mainly differs form ours in many ways. Their actionlets only comprises of different combination of joints, whereas our approach assigns probabilistic weighting for each skeleton joint. Therefore our action ensembles contain more meaningful information than actionlets. Secondly our approach only relies on universal skeleton features whereas actionlet based approach uses depth data associated with each joint position, called Local Occupancy Pattern (LOP). But these LOP features would depend on the objects that the subject interacts with. Therefore it may have difficulty in dealing with a subject performing an activity using different object sizes and shapes.

Use of probabilistic graphical models is one of the most popular techniques that has been used by automatic human activity detection. In [1] researchers used coupled HMM to detect human two hand activities and some others utilized motion template together with HMM to recognize human activities [6]. But these researchers didn't incorporate all the joint information in their models. However most of the human daily activities are too complex to recognize by only observering few joint features. Therefore those techniques would fail to recognize human daily activities with high intra-activity variations.

The paper is organized as follows. Sect. 3 describes overall activity detection model which is the core of our proposed approach. It details the algorithm we used to calculate joint confidence weights followed by the Dynamic Bayesian Network (DBN) that incorporates joint ensembles. Sect. 4 describes the implementation of the proposed approach and training of DBNs for activity detection. Experimental results are discussed in Sect. 5 followed by the conclusions in Sect. 6.

3 Activity Recognition Model

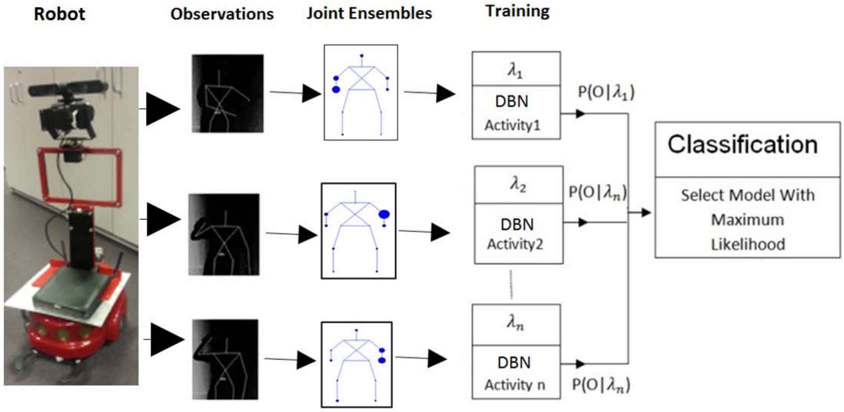


Fig. 1 Block diagram of the recognition process

Fig. 1 shows the overall process which is utilized in the proposed human activity detection method. First we identify joint ensembles and their associated weights for each and every activity in the data-set. Then we train separate Dynamic Bayesian Networks (DBN) by incorporating joints weights for each activity in the data-set. Once a new sequence of skeleton features has been captured, the previously trained models produce likelihood estimation, from which the maximum is selected.

3.1 Learning Action Ensembles

We represent each activity as weighted joint ensembles to better characterize intra class (same activity done with different body poses) variations of human activities. This allows us to identify common joint movements associated with each intra-class activity. The approach can be justified by the fact that all 14 skeleton joints do not contribute equally to a particular human activity. For example, for the activity "drinking water" most descriptive skeleton features would be 3D joint skeleton data of hands and the head. Therefore more weight may be assigned to joint positions of hand and head for the activity "drinking water". Following section describes the learning mechanism that has been utilized to identify joint ensembles and their associated weights for each activity in the data set.

3.1.1 Calculating Joint Confidence Weights

In the proposed algorithm weighted joint ensembles are denoted as W_a^j , where $j \varepsilon J = \{j_1, j_2, ..., j_n\}$ and $a \varepsilon A = \{a_1, a_2, a_3..a_m\}$. Here J is the set of skeleton joints and A is the set of all activities in the dataset and m is the number of activities. In addition the weights of each joint are constrained as in (1) to give probability value for each joint.

$$\sum_{j=1}^{n} W_a^j = 1 \tag{1}$$

We assumed each joint is independent of each other when calculating joint weights for a given activity. For the person p_n , joint j_n and for the activity a_n, we can denote the set of k observation sequence as $O_{J_n}^{p_n} = \{O_1, O_2, O_3, ..., O_k\}$. For each subset of observation sequences $S \subset O_{J_n}^{p_n}$, what we are interested in knowing is the similarity or the likelihood between the observation sequences. When calculating the likelihood of each observation sequence tempo-spatial movement of the joint need to be considered. In order to calculate the likelihood between observation sequences, we should be able to build models that efficiently represent observation sequences. Hidden Markov Models (HMM) have shown a great deal of success to model sequential data [10] and therefore, intra activity likelihood is calculated based on a HMM by training each joint and subsequent testing.

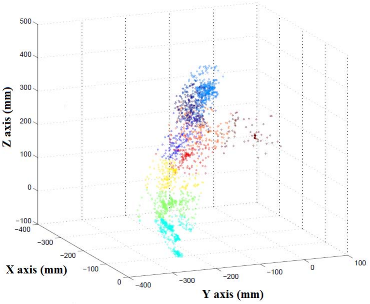


Fig. 2 (x,y,z) positions of the right hand with respect to the torso, when the action "drinking water" is performed

Fig. 2 shows position information (with reference to torso) of the right-hand's wrist joint when "drinking water" activity is performed. It shows few distinguishable clusters. In addition, within each of these clusters, few sub clusters can also be observed. This is due to the variation caused when the subject performs the same activity in different poses. Although unimodal Gaussians are used in HMM to model continuous data, it is not capable of capturing multimodal nature of the joint movements and hence in this research we implemented HMM based on Gaussian Mixture Models (GMM) in order to calculate joint likelihoods.

In GMM based HMM, observation probability given states s can be modelled with weighted sum of M component Gaussian densities as,

$$b_s(O) = \sum_{i=1}^{M} w_i g(x|\mu_i, \Sigma_i) \tag{2}$$

where x is a 3-dimensional continuous-valued joint position vector , $w_i, i = 1, ..., M$, are the mixture weights, and $g(x|\mu_i, \Sigma_i), i = 1, ..., M$ are the component Gaussian densities. Each component density is a 3-variate Gaussian function with mean of μ_i and covariance matrix of Σ_i.

GMM based HMM was trained for each joint with every observation sequence for a given activity. Given such two HMMs, λ_1 and λ_2, our interest is to find similarities from which the weights can be estimated: for higher similarities higher weights are assigned where as for less similarities lower weights are assigned. This concept of model dissimilarity can be generalized by defining the distance measure $D(\lambda_1, \lambda_2)$, between two HMMs as ,

$$D(\lambda_1, \lambda_2) = \frac{1}{T}[logP(O^{\lambda_2}|\lambda_1) - logP(O^{\lambda_2}|\lambda_2)] \tag{3}$$

where $O^{\lambda_2} = O_1, O_2, O_3..O_T$ is a sequence of observations generated by model λ_2 [10]. Equation (3) is a measure of how well λ_1 matches observations generated by model λ_2, relative to how well model λ_2 matches observations generated by itself. The dissimilarity measure discussed above is none-symmetric. Therefore for better representation (4) can be symmetrized by

$$D_s(\lambda_1, \lambda_2) = \frac{D_s(\lambda_1, \lambda_2) + D_s(\lambda_2, \lambda_1)}{2} \tag{4}$$

Finally, to estimate weights W_a^j associated with a given activity following steps have been followed.

```
for activity a=1 to A do
    for Observation o=1 to O do
        Train GMM based HMM $\lambda_j^a(o)$ for each joint $j$
    end
    for joint j=1 to J do
        • For all $S \subset \Lambda = \{\lambda_j^a(1), \lambda_j^a(2).....\lambda_j^a(n)\}$
          s.t N(S)=2, calculate dissimilarity measure
          $D_j^a(n)$ by (4) where $1 \leq n \leq C_3^n$.
        • Calculate total dissimilarity for joint j as $D^a_{j_{total}} = \sum_{n=1}^{C_3^n} D_j^a(n)$
        • Assign weight for the joint as $W_a^j = \frac{1}{D^a_{j_{total}}}$
    end
    Normalize all joint weiights s.t $\sum_{i=1}^{n} W_a^j = 1$ to assign probability value for weights.
end
```

Algorithm 1. Learning action ensemble joint weights

3.2 DBN for Action Recognition

Once joint weights are known, we can effectively model each activity by a Dynamic Baysian Network (DBN) as shown in Fig. 3. A DBN is a directed acyclic graph, which represents the conditional independencies and the conditional probability distributions of each node [7]. Shaded nodes represent the observed continuous 3-dimensional joint positions (J_j^t *where* $1 \leq j \leq 14,\ 1 \leq t \leq T$) and transparent squares represent the discrete hidden nodes. We have incorporated joint weights to the observation probability by an exponents as shown in (7). We assumed each human activity is a collection of different poses that evolves over time. Therefore, in the proposed model, top hidden node represents pose class and the middle hidden nodes represent mixture weight components. Pose classes are not directly observed as opposed to the joint positions, which can be directly measured from RGB-D camera's skeleton information.

The proposed DBN can be parameterized by three probabilities A, B and π as follows. First we define individual pose states as $S = \{S_1, S_2, ..., S_N\}$, the state at time t as q_t and K as the number of states. In the proposed model $a_{i,j}$ is the state transition probability from state i to state j and $b_t(i)$ represents the probability of the observation O_t given the i^{th} state of the pose nodes. Then initial state distribution, $\pi = \{\pi_i\}$ can be defined as

$$\pi_i = P(q_i = S_i),\ 1 \leq i \leq N \tag{5}$$

The observation probability distribution can be defined as, $B = \{b_t(i)\}$ where

$$b_t(i) = P(O_t | q_t = S_i),\ 1 \leq i \leq K, 1 \leq t \leq T \tag{6}$$

O_t is the joint observation at time t.

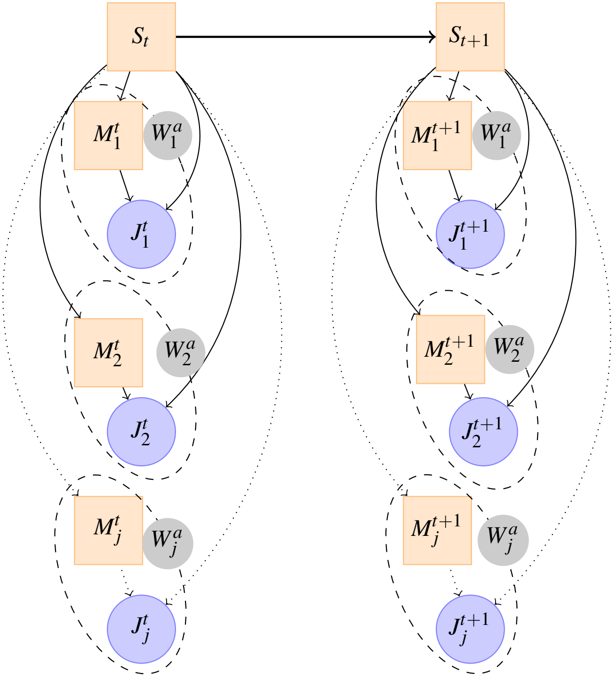


Fig. 3 Graphical representation of the proposed DBN. Square nodes represent discrete hidden nodes and round nodes represent observed continuous 3-dimensional joint positions. Dotted ellipses that encircle observations represent weights associated with each joint.

The observation probability with joint weight W_a^j that represents contribution of that joint to the activity, can be modeled as

$$b_t(i) = \prod_{j=1}^{J} \lceil \sum_{m=1}^{M_i^n} w_{i,m}^j N(O_t^j, \mu_{i,m}^n, \Sigma_{i,m}^j) \rceil^{W_a^j} \tag{7}$$

where J represents the total number of joints, O_t^j the observation vector of the j^{th} node at time t, M_i^j is the number of mixture components in the joint j and state i, and $\mu_{i,m}^j$, $\Sigma_{i,m}^j$, $w_{i,m}^j$ are the mean, covariance matrix, and mixture weight for the j^{th} joint, i^{th} state, and m^{th} Gaussian mixture component, respectively.

Finally the state transition probability distribution can be defined as $A = \{a_{i,j}\}$

$$a_{i,j} = P(q_{(t+1)} = S_j | q_t = S_i), 1 \leq i, j \leq K \tag{8}$$

4 Implementation

The proposed activity recognition model has been implemented using the Bayes Net Toolbox (BNT) for Matlab [8] which is public domain toolkit for modelling Dynamic Bayesian Networks.

4.1 Training Dynamic Bayesian Network

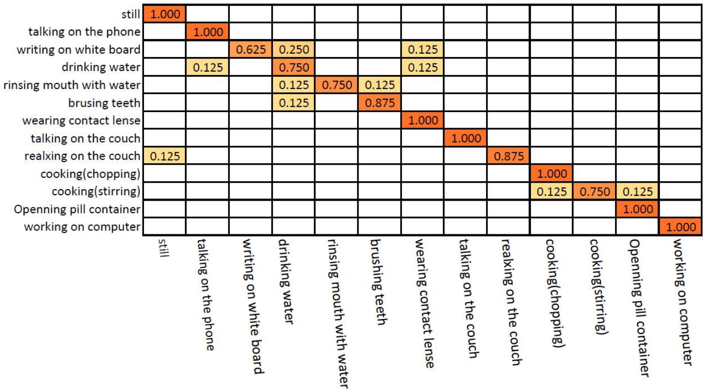
It is a standard practice to use expectation maximization (EM) algorithm to train parameters when a DBN contains any hidden nodes [3]. However it is well known that EM algorithm only converges to a local optimum solution. Therefore initial parameters of the model needed to be carefully chosen in order to get good classification results. In our proposed DBN for activity recognition we used an efficient method to initialize the parameters as explained in our previous research [9].

4.2 Activity Recognition

Once a HMM is trained for each action class, we need to select the most likely activity given an observation sequence. Given the observation sequence $O = O_1, O_2, ..., O_t$, and model $\lambda = (A, B, \pi)$ we calculated $P(O/\lambda)$, the probability of the observation sequence once the model is given(likelihood). Then the activity with the maximum likelihood is selected as the most probable activity. The log-likelihood calculation is done using the forward algorithm [10] for HMM that enabled us to recognize activities in real-time.

5 Experiments

First we tested our activity recognition model on the publicly available Cornell Activity Dataset 60 [11] to validate the model. Then we carried out our own experiments on activities with high intra class variations to test the performance of the model to intra activity variations. The empirical results show that proposed framework is capable of recognizing even highly similar activities with reasonable accuracy.



	still	talking on the phone	writing on white board	drinking water	rinsing mouth with water	brushing teeth	wearing contact lense	talking on the couch	realxing on the couch	cooking(chopping)	cooking(stirring)	Openning pill container	working on computer
still	1.000												
talking on the phone		1.000											
writing on white board			0.625	0.250			0.125						
drinking water		0.125		0.750			0.125						
rinsing mouth with water				0.125	0.750	0.125							
brusing teeth				0.125		0.875							
wearing contact lense							1.000						
talking on the couch								1.000					
realxing on the couch	0.125								0.875				
cooking(chopping)										1.000			
cooking(stirring)										0.125	0.750	0.125	
Openning pill container												1.000	
working on computer													1.000

Fig. 4 Confusion matrix for Cornell activity60 dataset

5.1 Model Validation through Cornell Activity Dataset

The Cornel activity [11] is consists of 14 activities carried out by four different individuals performing an activity once. Therefore, it is to be noted that the intra-activity complexity is limited to the variation among subjects.They have used the Microsoft Kinect RGBD sensor to record both depth and skeleton data of human daily activities done in a indoor environment. Data has been collected with four different people: two male and two females, recorded for about 45 seconds with each person, without compromising to any occlusion of arms and body. Therefore full skeleton was always observed throughout the activity. With this dataset, 2-fold cross validation testing has been carried out i.e we trained our model on two people and tested on others. Our experiments recorded precision and recall accuracies of 90% and 89% respectively. The confusion matrix is shown in Fig. 4. These results are in general better than the results obtained by [11] as can be seen from the Table 1.

5.2 UTS Experiments

There are few publicly available datasets that include skeleton data, which can be used in activity detection. However, they offer very limited intra-activity variations. The concept of weighted joint ensembles can be better explained and tested with a data set which has higher intra-activity variations.

Therefore, we have collected a dataset (UTS-Skeleton3D) consisting of 3D skeleton data. Fig. 5 shows the hardware set-up of the robot that we developed to aid our experiments. It consists of a RGB-D sensor mounted on a Ambigobot^{TM} mobile robotic platform. RGB-D sensor is mounted on a Pan-Tilt module. The robot, Pan-Tilt Module and the RGB-D sensor are interfaced by Robotic Operating System (ROS) and its drivers.

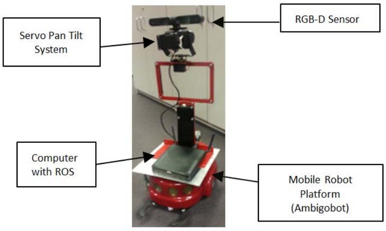


Fig. 5 Hardware set-up of the Robot

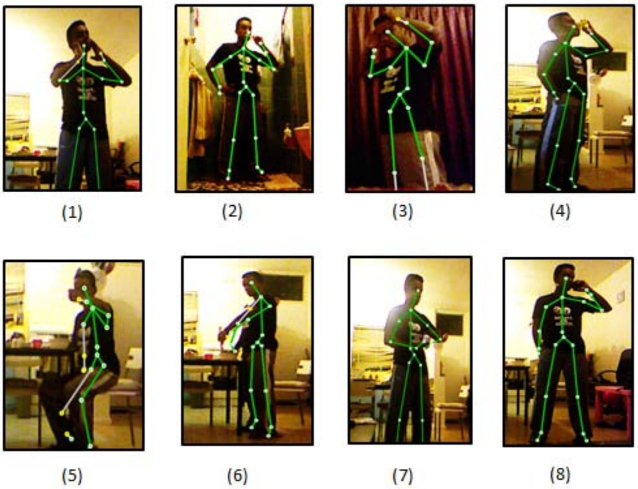


Fig. 6 Samples from our data set. (1) Applying Cream, (2) Brushing Teeth, (3) Combing, (4) Drinking, (5) Eating, (6) Stirring, (7) Opening a pill container, (8) Phone Call

The experimental dataset consists of 8 highly similar activities: applying cream on the face, brushing teeth, combing, drinking water, eating cereals, phone call, stirring and opening a pill container in a domestic environment. Four subjects were used to collect the data in which each activity is performed in three different body poses like, “sitting”, “standing” and “walking”. All together there are 96 samples of activities in the experiments. Each subject performed the activity about 45-60 seconds and data is recorded from different camera angles with a Microsoft Kinect sensor. Initially we recorded each joint’s 3-D position and orientation with respect to the sensor. Later we transformed the data w.r.t the torso coordinates to alleviate the effects of the sensor location

First we have calculated joint weights associated with each activity in the dataset by using the algorithm described in the Sect. 7. Fig. 7 shows the joint weights assigned for each activity in the dataset. The radius of the dark circle at each joint is proportional to the probabilistic weight assigned by the algorithm. We trained separate models for right-handed people and left-handed people. Therefore each activity is consisted of two models and likelihood calculations were done for each model, once observation sequence is received. From the Fig. 7 it is clear that proposed algorithm is capable of identifying importance of joints for a given activity. For example, the activity " applying cream on the face" has higher probability weights assigned to hand, forearms, and head while relatively low probability values has been assigned to other joints.

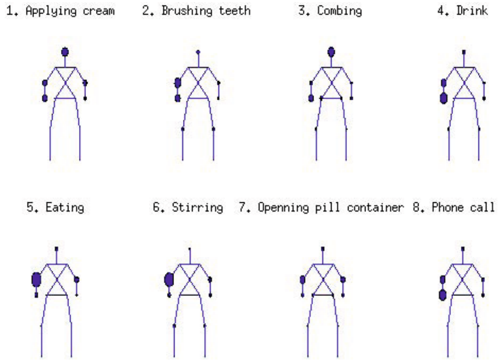
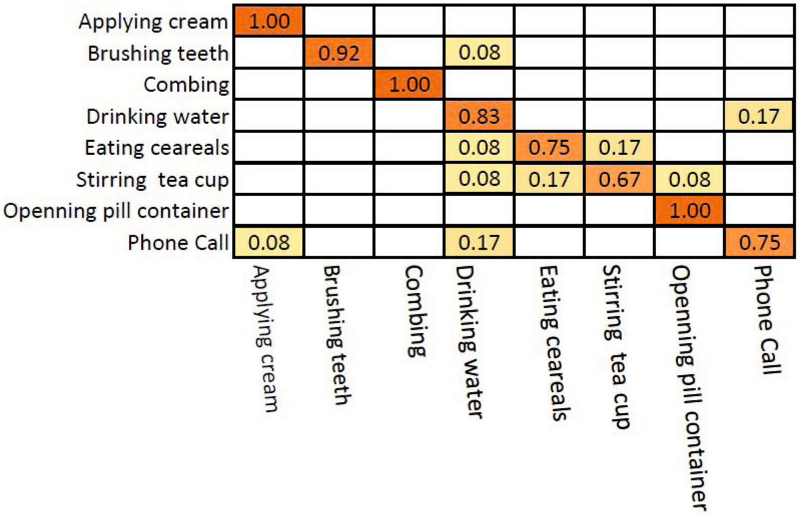


Fig. 7 Learnt weighted joint ensembles for right handed person. The radius of the circle at each joint is proportional to the joint weight.

Once joints weighs have been calculated, the DBN was trained for each activity with their associated joints weights ensembles. K-fold cross validation was used for testing, i.e we left out one sample activity and trained model and weights on others. Left out sample was then used as the activity to be detected. Same procedure was followed for all activity samples in the dataset. Confusion matrix of the test is shown in the Fig. 8. As can be seen, it has a very high rate of activity detection accuracies. It seems the "phone call" activity was slightly confused with "drinking water" activity. This is due to high similarity of the hand movements when these activities are performed and skeleton tracker often fails to track the hand when it is moving very close to the human body. "Stirring" is slightly confused with "Eating cereal" since "Eating cereal" often includes the "Stirring" as a sub activity of it. The proposed method was able to achieve recall and precision accuracies of 85% and 86% respectively. This is a high detection rate given the high intra-class complexity of the dataset. As it can be seen from Table 1 there is a significant improvement of detection rate when joint weighted ensembles are introduced to the DBN, in UTS experiments. This is because UTS experiments contain highly similar activities with very high intra-activity variation.



	Applying cream	Brushing teeth	Combing	Drinking water	Eating ceareals	Stirring tea cup	Openning pill container	Phone Call
Applying cream	1.00							
Brushing teeth		0.92		0.08				
Combing			1.00					
Drinking water				0.83				0.17
Eating ceareals				0.08	0.75	0.17		
Stirring tea cup				0.08	0.17	0.67	0.08	
Openning pill container							1.00	
Phone Call	0.08			0.17				0.75

Fig. 8 Confusion matrix for UTS experiments

Table 1 Recognition Accuracy Comparison for Different Datasets

Dataset	**DBN only**		**Proposed Method**	
UTS Experiments	Recall 66%	Precision 69%	Recall 85%	Precision 86%
Dataset	**Maximum Entrophy Markov Model [11]**		**Proposed Method**	
Cornel activity 60	Recall 57%	Precision 69%	Recall 90%	Precision 89%

6 Conclusions

In this paper, we presented weighted joint ensembles based human activity recognition system using skeleton features generated from an inexpensive RGB-D sensor. In the proposed technique, joint weights model the importance of that particular joint to the activity. Then we trained a DBN for each activity in the datasets and maximum log-likelihood estimation is calculated in-order to select the most probable model for a given sequence of observations. The proposed algorithm was tested with a challenging publicly available dataset and through UTS experiments with very promising accuracies. More importantly, it is shown that the proposed model is robust to intra-activity variations when people perform the same activity in different body poses.

In a real situation, the humans perform activities in a continuous way. Therefore future work involves detecting end of the activity to improve the model to a long term activity recognition system. In addition currently we are using supervised learning techniques to recognize activities that were previously seen. The reliability of the system can be further improved if the system can detect the difference between a new activity and a previously trained activity.

References

1. Brand, M., Oliver, N., Pentland, A.: Coupled hidden Markov models for complex action recognition. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 994–999 (1997), doi:10.1109/CVPR.1997.609450
2. Demiris, Y., Meltzoff, A.: The robot in the crib: A developmental analysis of imitation skills in infants and robots. Infant Child Dev. 17(1), 43–53 (2008)
3. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society, Series B 39(1), 1–38 (1977)
4. Stork, J.A., Spinello, L., Silva, J., Arras, K.O.: Audio-based human activity recognition using non-markovian ensemble voting. In: Proc. of IEEE International Symposium on Robot and Human Interactive Communication, RoMan (2012)
5. Lopes, M., Melo, F.S., Montesano, L.: Affordance-based imitation learning in robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, USA, pp. 1015–1021 (2007)
6. Martinez-Contreras, F., Orrite-Urunuela, C., Herrero-Jaraba, E., Ragheb, H., Velastin, S.A.: Recognizing Human Actions Using Silhouette-based HMM. In: 2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance, pp. 43–48 (2009)
7. Murphy, K.: Dynamic bayesian networks: Representation, inference and learning. Ph.D. thesis, UC Berkeley, Computer Science Division (2002)
8. Murphy, K.P.: The bayes net toolbox for matlab. Computing Science and Statistics 33 (2001)
9. Piyathilaka, L., Kodagoda, S.: Gaussian mixture based hmm for human activity recognition uisng 3d skeleton features. In: 8th IEEE Conference on Industrial Electronics and Applications (2013)
10. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE 77(2), 257–286 (1989), doi:10.1109/5.18626
11. Sung, J., Ponce, C., Selman, B., Saxena, A.: Human activity detection from rgbd images. In: Plan, Activity, and Intent Recognition, vol. WS-11-16. AAAI (2011)
12. Theodoridis, T., Agapitos, A., Hu, H., Lucas, S.M.: Ubiquitous robotics in physical human action recognition: A comparison between dynamic anns and gp. In: ICRA, pp. 3064–3069. IEEE (2008)
13. Wu, Y., Yuan, J., Liu, Z., Wang, J.: Mining actionlet ensemble for action recognition with depth cameras. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1290–1297 (2012), doi: http://doi.ieeecomputersociety.org/10.1109/CVPR.2012.6247813

Building Environmental Maps of Human Activity for a Mobile Service Robot at the ”Miraikan” Museum

Ippei Samejima, Yuma Nihei, Naotaka Hatao, Satoshi Kagami,
Hiroshi Mizoguchi, Hiroshi Takemura, and Akihiro Osaki

Abstract. This paper describes environment maps that are comprised of the following three types of information, 1) 3D environmental changes that represents human activities, 2) human trajectories in 2D that represent how humans move in the environment, and 3) human posture data. These maps are utilized in order to plan safer, quicker and/or non-human-disturbing paths for a mobile service robot at the museum ”Miraikan”. Experiments are conducted within ”Miraikan” and results are shown.

1 Introduction

Autonomous carrier, cleaning, and security guard robots have recently become commercially available. The mobility function of such robots assumes quasi-static, indoor environments with few human inhabitants. Research in simultaneous localization and mapping (SLAM), mapping, localization, and path planning fields actively targets quasi-static environments. However, robots in human rich environments need further information about how humans use the environment to achieve safe and efficient behavior. Konolige et al. proposed lifelong environmental maps for

Ippei Samejima
Digital Human Research Center,
National Institute of Advanced Industrial Science and Technology,
2-3-26, Aomi, Koto-ku, Tokyo, 135-0064, Japan
e-mail: i.samejima@aist.go.jp

Yuma Nihei · Hiroshi Mizoguchi · Hiroshi Takemura
Tokyo University of Science, 1-3 Kagurazaka, Shinjuku, Tokyo 162-0825, Japan
e-mail: yuma-nihei@aist.go.jp, hm@rs.noda.tus.ac.jp

Naotaka Hatao · Satoshi Kagami
National Inst. of AIST, 2-3-26 Aomi, Koto-ku, Tokyo, Japan
e-mail: n.hatao@aist.go.jp, s.kagami@aist.go.jp

Akihiro Osaki
The National Museum of Emerging Science and Innovation (Miraikan),
Tokyo, Koto, Aomi, 2-3-6
e-mail: a-osaki@miraikan.jst.go.jp

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_28, © Springer International Publishing Switzerland 2015

daily living environments [1] to tackle this kind of problem that focused on temporal changes in map occupancy. Hamada et al. also proposed a human trajectory map that categorized human use by using moving speed and direction [2]. This information was used to bias results from robot path planning so that they could efficiently move through space cohabited by human beings.

We focused on a map used by a mobile service robot at a museum. Several such studies have been reported such as the RHINO robot at the Deutsches Museum in Bonn [3], and Minerva at the Smithsonian's National Museum of American History [4]. The Minerva study demonstrated that robots in museum situations sometimes encounter crowds of people, and not just single individuals, and the typical interaction time is quite short.

We have also done studies at the ”Miraikan” Museum and identified three difficulties with mobility such as a) crowds of people moving in corridors and stopping in front of displays, b) frequently changing environments (such as fences, stools for small children, and guideboards), and c) changes in displays or exhibition areas. It is important for robots operating in museums, whatever their tasks are (such as guiding, giving tours, or cleaning), to better understand how humans use the environment. Therefore, we propose three types of maps to help robots understand the way humans use the environment: 1) a 3D environmental change map that represents human activities, 2) a map of human trajectories in 2D that represents how humans move in the environment, and 3) a human posture map to detect human interaction with museum displays.

2 Concept of Human Environment Maps

Maps are searched in the path planning phase for minimum cost paths by minimizing the accumulated costs that are assigned to each map cell. Although we can obtain minimum length paths, such a strategy might not be appropriate for museum environments. What we would like to obtain is somewhat safe, efficient, and nonhuman- disturbing paths. Three maps were constructed to obtain such paths: 1) a 3D environmental change map, 2) a map of human trajectories in 2D, and 3) a human posture map. The robot could understand more about the environment by combining these three maps and generate better paths to reach its goal. This information was also useful for museum management purposes, in addition to such direct usage by the robot, such as how many people looked at displays, how long they spent looking at them, the human density of corridors, and the existence of bottlenecks.

3 3D Environmental Change Map

Static maps that include walls and columns of buildings are relatively easy to construct. However, there are objects in the environment whose locations/states can change, such as doors, chairs, fences, stools for small children, and guide boards. Also, museum displays occasionally change, and some displays are actually movable. Such objects can be avoided after they become visible to robot sensors by re-planning paths. However, when a robot becomes aware of typical changes in the environment,

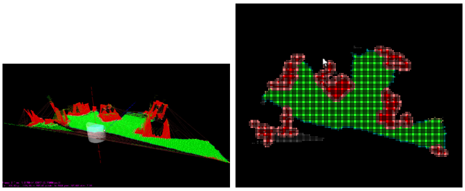


Fig. 1 DEM generation

it can utilize that information to plan paths that traverse through areas more likely to be open space.

3.1 Finding Regions with 3D Obstacles

There are obstacles with complex shapes in a museum environment that a robot needs to distinguish from the background to avoid possible collisions. A swinging laser range finder (3D LRF) was utilized to achieve this. The 3D LRF returned a point cloud that represented the surrounding environment. As the points were not distributed uniformly over 2D space, we utilized digital elevation maps (DEMs) and Delaunay triangulation to detect 3D obstacle regions [5]. A DEM is a kind of 2D grid map with each grid cell containing height information. DEMs enable the number of data to be reduced in dense sampling areas. If there are several 3D LRF points in a DEM cell, the height of the cell becomes the height of the highest 3D LRF point. Figure 1 shows the results for DEM cell generation. Local path planning uses DEMs by converting "ground" into traversable regions and "obstacles" and "near obstacles" into non-traversable regions.

3.2 Generation of 3D Obstacle Changing Map

When the robot moves in the environment, 3D LRF data are always converted into local DEM cell classifications to avoid obstacles around the robot. The global 3D obstacle changing map is statistically updated with Eq. 1 by using this local DEM map.

$$cost = \frac{1}{t}\sum_{i=0}^{t} ||M_i - M_{i-1}||, \qquad (1)$$

where M indicates whether the cell is traversable (0), or not (1), and t is the number of updates. The path planner can utilize this 3D environmental change map to generate the path most likely to avoid potential collisions due to moving obstacles.

4 Pedestrian Trajectory Map

Useful information can be extracted by looking at human pedestrian behavior. For example, as regions that humans quickly traverse can be considered to be corridors, the robot can also utilize these regions. People walking in a museum tend to avoid going between displays and other people that are looking at the displays. Such behavior is observed and aggregated into maps, so that robots can plan similar paths.

4.1 Tracking and Identifying Pedestrian Trajectories

We often see groups of people moving together at museums. As they occlude each other in such places, it is difficult to track and identify human trajectories. We adopted cluster based sample-based joint probabilistic data association filters (SJPDAFs) [6] that use 2D LRF input to accomplish this.

SJPDAFs are robust against false positives and negatives, and they make it possible to flexibly design individual trackers using particle filters. The proposed method divides tracking targets and corresponding LRF segments into clusters, and classifies each cluster as a group of pedestrians. The number of pedestrians in each cluster is estimated independently, and each pedestrian in a cluster is tracked individually. Individual tracking enables the number of merging and splitting clusters to be estimated. Furthermore, the proposed method can remove the unwanted effects of false segmentation of LRF scans.

A SVM is adopted to classify and estimate the number of pedestrians. The method adopts time-series estimates as the shapes of LRF scan segments are not stable. The class definition of moving objects is c_0: false positive, c_1: one pedestrian, c_2: two pedestrians, and so on, where c_k indicates the label of each class. We define the feature vector of LRF scans in a cluster at time t as $z_f(t)$, and a set of feature vectors from time 0 to t as $Z_f^t = z_f(0) \cdots z_f(t)$. The value we want to estimate is $P(c_n|Z_f^t)$, and we obtain:

$$P(c_k(t)|Z_f^t) = \alpha \cdot P(z_f(t)|c_k(t)) \cdot P(c_k(t)|Z_f^{t-1}) \tag{2}$$

We define the feature vector of LRF scans in a cluster at time t as $z_f(t)$, and a set of feature vectors from time 0 to t as $Z_f^t = \{z_f(0) \cdots z_f(t)\}$. The value we want to estimate is $P(c_n|Z_f^t)$, and we obtain:

$$P(c_k(t)|Z_f^{t-1}) = \sum_n [P(c_k(t)|c_k(t-1) = n) \cdot P(c_k(t-1) = n|Z_f^{t-1})] \tag{3}$$

Also, from Bayes' theorem, we obtain:

$$P(z_f(t)|c_k) = \alpha \frac{P(c_k|z_f(t))}{P(c_k)} \tag{4}$$

$P(c_k|z_f(t))$ can be estimated using SVM, and $P(c_k)$ can be estimated using training sets of SVM.

Table 1 Numbers and frames in SVM training data set

Class	c_0	c_1	c_2	c_3	c_4	c_5
desc.	False positives	1 pedestrian	2 pedestrians	3 pedestrians	4 pedestrians	5 or more pedestrians
Numbers	212	318	54	9	5	3
Frames	868	20362	3631	352	267	58

The features for SVM are defined as follows:

z_{f0} : Number of LRF segments
z_{f1} : Sum of lengths of LRF segments
z_{f2} : Average speed
z_{f3} : Difference between angle of directed bounding box and angle of average velocity vector
z_{f4} : Length of long side of directed bounding box
z_{f5} : Length of short side of directed bounding box

Table.1 lists the total numbers of each class and the total numbers of frames detected in each class.

4.2 Aggregated Pedestrian Existence Map

An aggregated pedestrian traversal map is generated by accumulating detected pedestrian trajectories into each cell and normalizing them by time.

This map is useful for finding where humans can be reached in the entire area. If a cell and a surrounding area have no aggregated data, that may mean that humans are not allowed in the region or it is not it possible for them to traverse it, so even this is an open space for a robot, it may not be a good idea to use that region.

4.3 Aggregated Pedestrian Stopping Position Map

Slow or zero speed pedestrian trajectories are chosen to accumulate at each cell to generate an aggregated pedestrian stopping position map. If the average speed in the last N frame measurements is less than the threshold, v[m/s], this segment is treated as a stopping motion. The number obtained at each cell is normalized as previously described for the existence of aggregated pedestrians.

If the cell and surrounding area has a high stopping position score, humans tend to stay around that area. This often happens in front of displays and resting areas in museums. A robot may not find it nice to traverse those areas since humans are concentrating on displays or are relaxing. In either case, they may not want to be distracted by the robot, or the presence of humans may cause the robot to potentially collide with them. However, as slowly moving humans in this area may need help with guidance, the robot may have to reach humans to interact with them using an additional margin for safety.

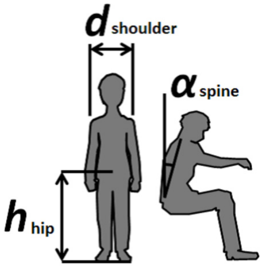


Fig. 2 Conditions to assess posture

4.4 Variance in Pedestrian Velocity Map

Variations in the velocity of pedestrians at each cell were calculated from the walking speed of those who traversed each cell to generate this map.

When a group of people is moving in the corridor area, a few people first occasionally stop to wait for other people, where there are no displays. This information can be used to find corridor regions where people tend to stop and gather. This means there is enough space in the region, so that the robot can utilize such places for moving purposes, since there should be few obstacles around.

5 Human Posture Map

There are displays that can physically interact with human beings. The robot can find humans who are interacting with such displays by previously knowing where such displays are, and where interfaces are.

We adopted the Microsoft Kinect sensor as a marker-less motion capture system. This sensor was installed on a pan unit, and combined with the previously described pedestrian tracking system to control who (where) to look at. Controlling where to look at is required as the Kinect sensor has a limited working range, view angle, and maximum number of subjects that can be simultaneously tracked. We usually controlled Kinect to look at the closest human from the robot, but it could also track a specified human. Joint angles and limb lengths were obtained by utilizing the Kinect software development kit (SDK), and these data were mapped onto a 2D map.

Joint angles and limb lengths could also be used to identify where humans sat or rested.

The sitting posture was distinguished by joint angle and position conditions. Fig.2 shows three condition parameters. These values were thresholded by ratio p of

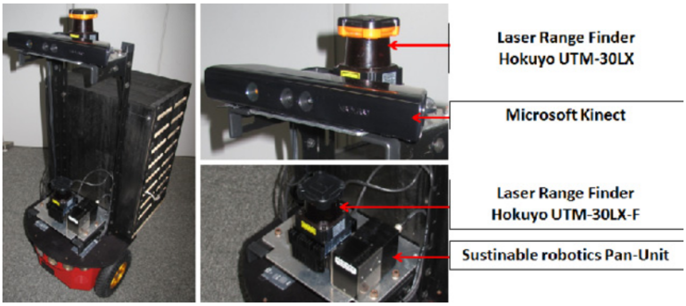


Fig. 3 Robot System

observation time where all three parameter conditions were satisfied. The three parameters were:

- Hip height, h_{hip},
- Angle between the spine and the vertical direction, α_{spine}, and
- Shoulder width $d_{shoulder}$.

Hip height h_{hip} was used to distinguish standing and sitting postures. α_{spine} had a maximum and minimum threshold to exclude forward and backward bent postures. Since Kinect SDK often incorrectly recognized the human skeleton when looking from the side of the subject, especially when the far side arm was hidden by the torso, $d_{shoulder}$ had a maximum and minimum threshold. Those three conditions were used to determine sitting/standing postures.

6 System and Environment Setup

Fig.3 has photographs of the robot that we used in the experiments. The mobile base was the Pioneer-III of Adept Mobilerobots. The onbody sensor for the 3D LRF was the Hokuyo UTM-30LX-F with a Pan-Unit from Sustainable Robotics, and that for the 2D LRF was the Hokuyo UTM-30LX with the Microsoft Kinect sensor. 2D LRF was not only utilized to track pedestrians racking, but also to generate 2D maps and 2D localization.

The experiments were conducted on the 3rd floor of the National Museum of Emerging Science and Innovation (Miraikan). Fig.4 shows a CAD drawing and photographs of the environment. The total area was about 120×35 [m]. There are displays with complex shapes in the environment as can be seen in Fig.4 (bottom two rows). There are also many stools for small children that can be moved.

A 2D map was generated before the experiment from 2D LRF data obtained by manually controlling the robot. Fig.4 shows a 2D map with a 5-cm grid.

7 Experiments

Three type of maps were generated and evaluated: 1) a 3D obstacle changing map, 2) a pedestrian trajectory map, and 3) a human posture map. The experiments were conducted at Miraikan as explained in the previous section.

7.1 Experiment to Generate 3D Obstacle Changing Map

The robot autonomously navigated itself by generating a path while the 3D LRF obtained data associated with localization results from a previously obtained 2D environmental map; a human operator manually provided input.

This experiment was conducted after the museum had closed and the exhibition areas were free of visitors (however, there were maintenance people present). The total length of the robots path was 1014.9 [m]. The DEM resolution was a 5-cm grid (the same as that for the 2D map) and the threshold for the Delaunay edge angle was set to 10 [deg]. DEM voting was limited to 5 [m] around the robot from the 3D LRF results since increasing the distance made sampling very sparse.

The size of the error ellipsoid area and variance in the ellipsoid direction calculated from the covariance matrix of the particle filter are summarized in Table.2 to evaluate 2D localization.

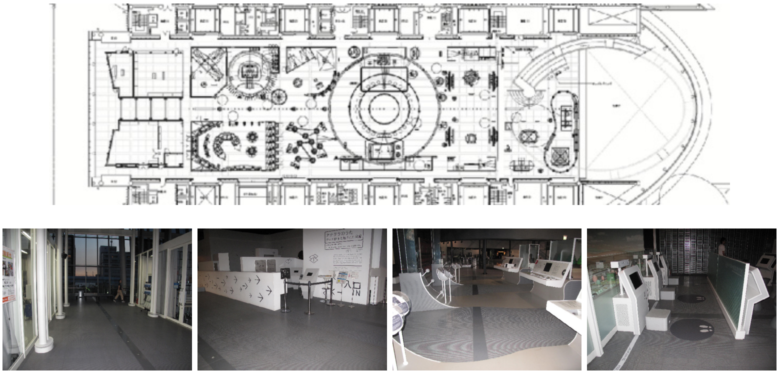


Fig. 4 CAD drawing of 3rd floor of "Miraikan" (top) and photographs of 3D obstacles (bottom)

Table 2 Accuracy of localization

	Max.	Min.	Average	SD
Size of error ellipsoid area [m^2]	0.24	0.0	0.029	0.019
Variance in error Ellipsoid direction [rad]	0.14	0.01	0.040	0.013

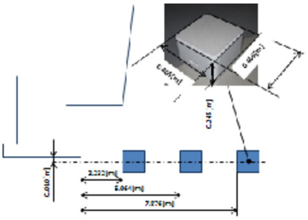


Fig. 5 Stools and their placement

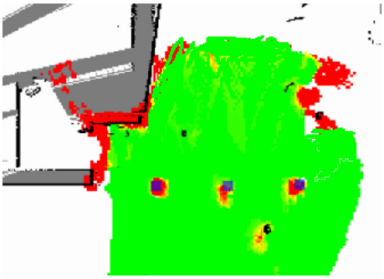


Fig. 6 Detected stools

Table 3 Accuracy of detecting 3D obstacles

	Average error	SD
Distance between centers of gravity [m]	0.19	0.06
Area size of obstacles [m^2]	0.11	0.10

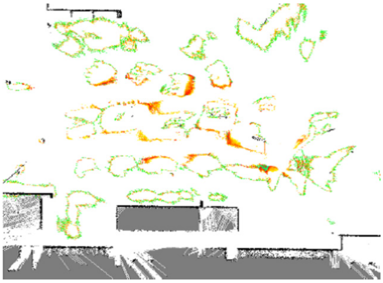


Fig. 7 Environmental change map

7.1.1 Measurements of Accuracy of Detection

We manually moved the stools for small children into a flat area to evaluate the accuracy of detection, and manually measured their positions. Accuracy was evaluated by the size of the detected area and distance between the centers of gravity of the objects. Fig.5 outlines the experimental setup.

The same conditions as in the previous experiment were adopted, and robot behavior was also the same. The resulting map is given in Fig.6.

Table.3 summarizes the size of the detected area and the distance between stools.

Since the 2D map and DEM resolution were on 5[cm] grids, we considered the results we obtained to have sufficient accuracy.

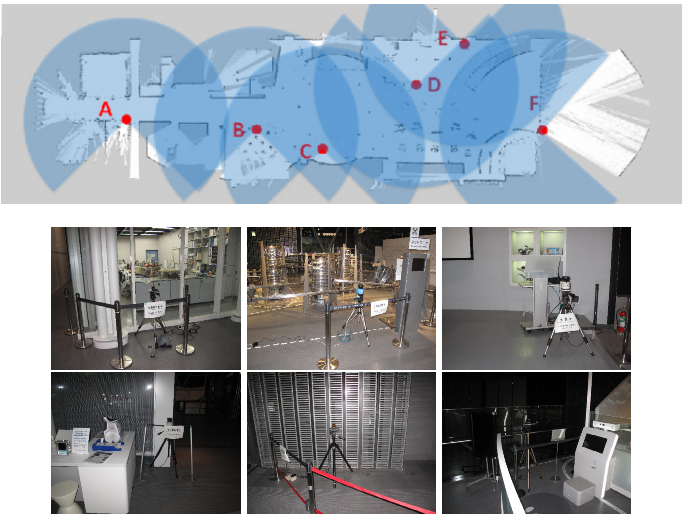


Fig. 8 Positions and photographs of sensors

7.1.2 Detection of 3D Changing Environment

We manually moved the stools for small children into the display area to generate a 3D environmental change map, and the robot measured the previous stools.

Fig.7 shows the 3D environment change map. The red regions in the figure indicate changes that have happened. Because the stools were moved, there are yellow or red regions around the displays.

There are also yellow or red regions around the obstacles because of errors with localization, odometry, or 3D LRF.

7.2 Experiment to Generate Pedestrian Trajectories

7.2.1 Obtaining Pedestrian Trajectories

Measurements were conducted on one weekday from 11:00 am until 5:00 pm. Fig.8 shows the 2D LRF arrangement. Six 2D LRFs were placed on top of a tripod. Three SICK LMS200s were placed at points B, C, and D, and three Hokuyo UTM-30LXs were placed at points A, E, and F for 2D LRF. The height of the sensors was set so that it was 0.9 [m] above the ground and this was the same as the 2D LRF on the robot. The working range to detect pedestrians was limited to 20 [m]. The coverage of the sensors overlapped as can be seen from the figure.

Fig.9 shows the trajectories we obtained. Trajectories that were shorter than 5 [s] were omitted to prevent false positives from being detected caused by occlusion or noise. The total number of trajectories that remained was 126,839. The color indicates the average speed of each trajectory, where blue means slow and red

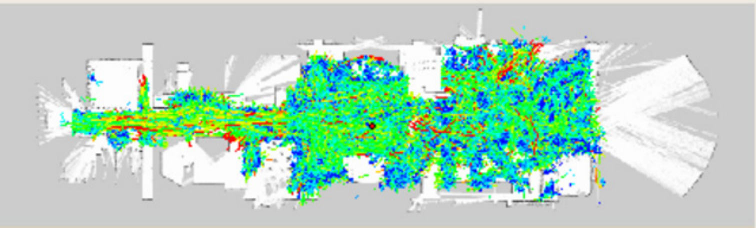


Fig. 9 Obtained pedestrian trajectories

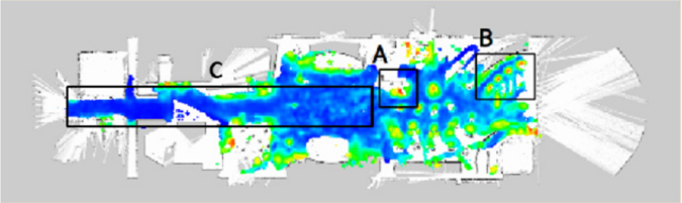


Fig. 10 Map of aggregated pedestrian stopping positions

means fast. As can be seen from the figure, visitors covered all possible movable regions.

7.2.2 Generation of Pedestrian Trajectory Map

Three maps were generated from the previous experiment. Theses were 1) an aggregated pedestrian stopping map (Fig.10), 2) an aggregated pedestrian velocity map (Fig.11), and 3) a variance of pedestrian velocity map (Fig.12).

The red region in Fig.10 indicates a higher probability of pedestrians. Regions A and B in the figure are mostly red, and as they are close to displays they tend to stop there. However, region C in the figure is basically used as a corridor, and indicates a low probability of pedestrians. Consequently, these tendencies indicate that this map is of a real use environment.

Fig.11 indicates pedestrian speed and the red regions indicate faster motion. Region D in the figure, which is farther away from the wall, indicates faster motion. As this region has few obstacles to move around and there are no displays close by, humans there tend to move quickly.

Fig.12 indicates variance in the walking speed. The red region has higher variance. Region E is used as a corridor where there are no displays around, but sometimes people stop there to wait for other people in their group, so that variance increases. Region F is around a display and variance in speed tends to be small in such regions. However, as this region is at the entrance of a corridor, variance increases.

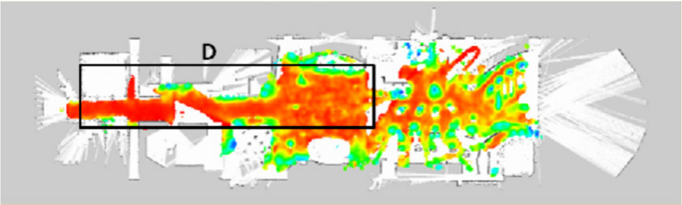


Fig. 11 Map of aggregated pedestrian velocity

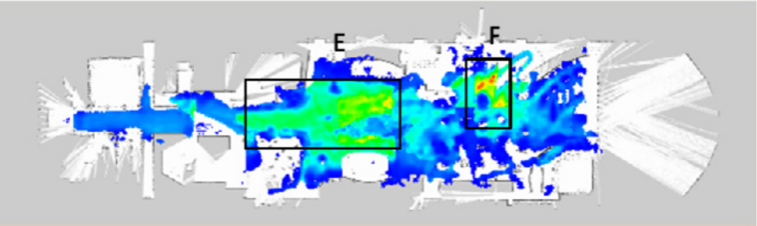


Fig. 12 Map of aggregated variance of pedestrian velocity

7.3 Experiment on Generation of Posture Map

An experiment to detect human postures and map them onto a posture map was conducted at points A, B, and C in Fig.13. There are displays with physical interaction at points A and B, where point C is a technical information counter where visitors sit and ask the science staff of Miraikan questions.

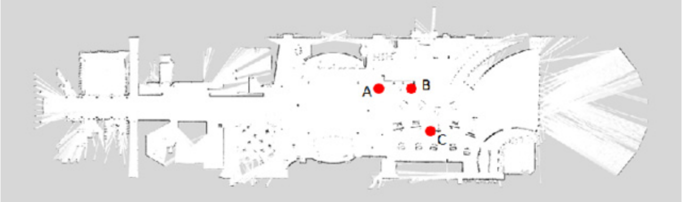


Fig. 13 Target position in experiment to generate posture map

The results are shown in Fig.14,Fig.15, and Fig.16. The yellow circle indicates the robots position, the blue human posture indicates that the subjects were detected as standing, and the red human posture indicates that the subjects were detected as sitting.

All posture and detection results for standing or sitting postures were aggregated onto a map. Such a map can be utilized to identify where displays are with physical interactions, and where people are resting or standing.

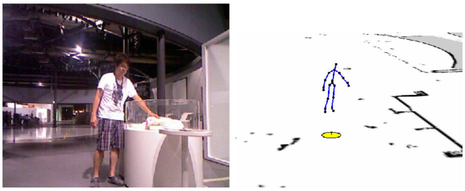


Fig. 14 Posture map experiment at point *A*

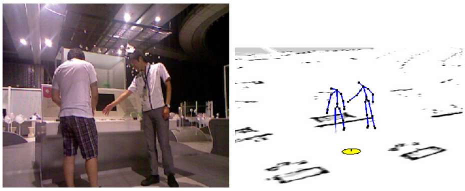


Fig. 15 Posture map experiment at point *B*

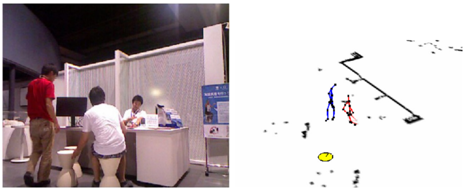


Fig. 16 Posture map experiment at point *C*

Fig. 17 A developing robot and children

8 Conclusion

We described three types of methods of generating maps that were aimed at utilizing an autonomous robot in a museum environment, i.e., 1) a 3D environmental change map to identify human activities, 2) a pedestrian trajectory map to identify how humans use locations, and 3) a human posture map to identify how people interact with displays or the environment. Since simply a static map of environmental shapes was not sufficient in such a crowded environment, this new information should be useful in mobility based robotic services such as guidance, tours, and cleaning.

We developed several key components to obtain each map. DEM with a Delauney triangulation representation was used to segment 3D obstacle regions from a 3D point cloud with the method of detecting 3D obstacles from 3D LRF explained in Section III. The method of SJPDAFs with SVM based classification and estimation was used to simultaneously find multiple pedestrians in the pedestrian tracking from 2D LRF discussed in Section IV. This pedestrian tracking function was also used to orient the KINECT sensor presented in Section V.

This was a joint project undertaken with the National Museum of Emerging Science and Innovation (Miraikan), and the experiments were conducted on the third floor.

We are now jointly developing a mobile service robot that has an omni-directional telescopic microphone array for human interaction and is covered with a flexible shell so that it does not injure visitors.

References

1. Konolige, K., Bowman, J.: Towards lifelong visual maps. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009), pp. 1156–1163 (2009)
2. Hamada, K., Kagami, S.: Generation of Environmental Information Cost Map for Mobile Robots based on Measurement of Human Trajectory. In: The 28th Annual Conference of the Robotics Society of Japan (RSJ 2010), p. 3Q2-1 (2010)
3. Burgard, W., Cremers, A.B., Fox, D., Hähnel, D., Lakemeyer, G., Schulz, D., Steiner, W., Thrun, S.: The Interactive Museum Tour-Guide Robot. In: Proc. of the Fifteenth National Conference on Artificial Intelligence, AAAI 1998 (1998)
4. Thrun, S., Bennewitz, M., Burgard, W., Cremers, A.B., Dellaert, F., Fox, D., Hähnel, D., Rosenberg, C., Roy, N., Schulte, J., Schulz, D.: MINERVA: A second generation mobile tour-guide robot. In: Proc. of the IEEE International Conference on Robotics and Automation, ICRA 1999 (1999)
5. Hatao, N., Kagami, S., Hanai, R., Yamazaki, K., Inaba, M.: Construction of Semantic Maps for Personal Mobility Robots in Dynamic Outdoor Environments. In: Yoshida, K., Tadokoro, S. (eds.) Field and Service Robotics. STAR, vol. 92, pp. 281–296. Springer, Heidelberg (2014)
6. Bar-Shalom, Y.: Extension of the Probabilistic Data Association Filter to Multi-Target Tracking. In: Proc. of the 5th Symposium on Nonlinear Estimation, pp. 16–21 (1974)

Part IX
Agriculture Robots

Accuracy and Performance Experiences of Four Wheel Steered Autonomous Agricultural Tractor in Sowing Operation

Timo Oksanen

Abstract. In agriculture, a typical task is to do a coverage operation for a field. Coverage path planning algorithms can be used to create the path for a vehicle. In case of an autonomous agricultural vehicle, the path is provided to the guidance or navigation system that steers the vehicle. In this paper, a four wheel steered tractor is used in autonomous sowing operation. The full size tractor is equipped with 2.5 m hitch mounted seed drill and the developed guidance system is used to sow about six hectares spring wheat. In this paper is presented the results of the guidance accuracy in the field tests, in four field plots. The guidance accuracy in terms of lateral and angular error to the path is typically less than 10 cm and one degree, respectively. The paper also presents real life problems happened in the field tests, including losing GPS positioning signal and tractor safety related wireless communication problems.

1 Introduction

The trend in agricultural engineering is to improve the productivity; to make one farmer to produce more food with the help of tools and technology. The improvement has been remarkable since the 19th century. Agricultural mechanization was ranked 7th in all achievements in the 20th century, leaving e.g. computers and telephone behind [1]. During the 20th century the development was also important for the industrialization as agriculture did not require so much workforce anymore. The trend to improve the productivity continues, but the mechanization and scaling up are not providing remarkable improvement any more. Electronics,

Timo Oksanen
Aalto University, School of Electrical Engineering,
Department of Automation and Systems Technology. Otaniementie 17, Espoo, Finland
email: timo.oksanen@aalto.fi

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_29, © Springer International Publishing Switzerland 2015

automation and ICT are used together with the mechanical agricultural tools to improve systems. Some call this mechatronization, some call it automation and some call it robotization – despite the name, the main purpose is to improve productivity; to produce the food with less workforce and less effort.

Agricultural field robots or autonomous agricultural machines have been proposed for a long time [2, 3]. A remarkable step towards autonomous operations was the development and availability of GPS technology [4]. So far, very little commercial success is reached compared with the visions of the future, but on the other hand the same applies to autonomous road vehicles. The field robots help to improve productivity in a way; the farmer does not need to guide the vehicles onboard, but he could use his/her time to do other agricultural tasks meanwhile. Still, the story of field robots is not only improving the productivity, but also improve the precision and accuracy of agricultural production. The precision refers to naturally to spatial accuracy, but also approach towards operations-on-demand – as the price to do field operations decreases and allows more unmanned operations to care the crops and harvest site-specifically.

A roadmap towards the commercial success of agricultural field robots is needed to improve robustness. Several attempts to build autonomous field robots have proven to be accurate enough, e.g. in guidance, but not in a way that long working hours and durability in a season is paid much attention [2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]. Some of the studies are related to specific application, like weed scouting or control [3, 13, 16] or planting [12], but typically the guidance studies are not related too much on the agricultural operation, but being generic [2, 5, 7, 9, 10, 11].

Looking back in the history of agricultural engineering, many early tractor prototypes with an internal combustion engine were build in 1910’s and the versatility in engineering was blooming. In 1920’s *tractor trials* were organized, e.g. in England, to make the tractor more common tool in farms [17]. In the tractor trials, the purpose was not only marketing in public demonstration, but the events also pushed the manufacturers to improve robustness, performance and usability as the machines were benchmarked and evaluated and the results were published. Perhaps ”agricultural field robot trials” are necessary one day to promote the technology and push the engineers to improve quality.

This paper presents a trial, carried out in Finland, to evaluate the developed autonomous tractor in sowing operation, in real fields, in real operation. The purpose of the trial was to evaluate performance factors in long run; the trial took one and half days, sowing 6.6 ha. This cannot be considered a season long operation, but it would reveal design flaws, e.g. in durability, robustness and temperature issues.

2 Materials

The tractor, the positioning system and navigation algorithms are reported and discussed earlier in [18], [19], [20] and [21]. In this chapter, the main functions and features of the system are presented.

2.1 The Autonomous Tractor "APU-Module"

The tractor known as "APU-Module" is shown in Fig. 1. The tractor was originally built by a Finnish company Modulaire Oy in the years 1990-1995. The four-wheel steered tractor is equipped with 123 kW diesel engine and hydrostatic transmission. The wheelbase is 2.7 m and the weight is 5900 kg. Each wheel steers maximum 22° and the control system keeps the steering angles synchronous, to realize the Ackermann steering principle. The control system (electrics, electronics, communication, software) was completely refurbished in the years 2009-2012. [18]

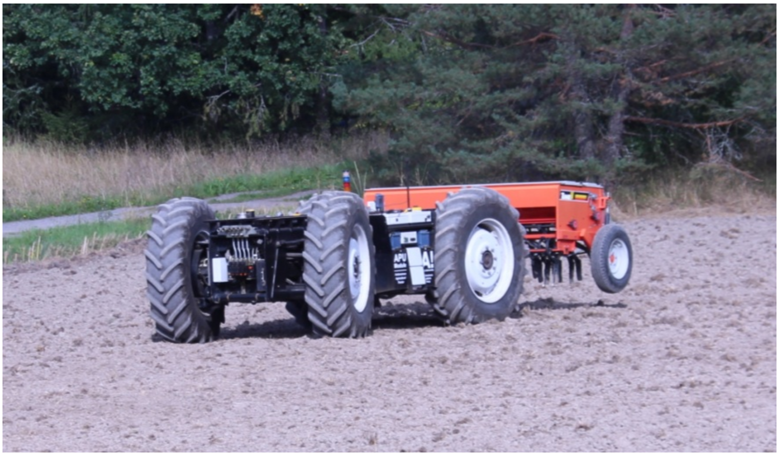


Fig. 1 APU-Module, the autonomous tractor

The steering is based on a dedicated hydraulic pump, a hydraulic valve block with four directional valves and four steering cylinders, one for each wheel. The control system for the synchronous steering of four wheels was developed during the refurbishment process. The maximum steering rate is limited by the hydraulic pump that needs to produce flow for four cylinders, compared with traditional tractor design where only one hydraulic cylinder or actuator is used for steering. In the closed loop control system, the identified control delay was 400 ms plus second order dynamics (time constant ~600 ms); the maximum steering rate is 8-12 °/s depending on steering directions if all the wheels are steered synchronously with 1500 RPM engine speed. Based on the field experiments, the hydraulic pump should be larger (volume per revolution) in order to do navigation in full speed, 3.0 m/s. [18, 20]

As the steering rate is limited and there is remarkable dead time delay, the guidance accuracy decreases when high driving speed is used. In first field tests, it was found that in practice with driving more than 2.0 m/s the path following accuracy decreases under tolerance. Therefore, the operating speed in navigation requiring accuracy is limited to about 2.0 m/s. In the trial presented in this paper, operating speed 1.8 m/s was used as a compromise between operational efficiency and navigation accuracy. [21]

In the sowing operation, the tractor is equipped with a mounted seed drill: Tume KL-2500; see Fig. 2. The seed drill is a combined seed and fertilizer drill, both seeds and local fertilizing are applied at the same time. Local fertilizing is applied in the middle of every other seed row by using another set of coulters (a device making a furrow for seeds/fertilizer). For seed rows the inter row width is 12.5 cm. The working width of the seed drill is 2.5 m, thus 20 seed coulters and 10 fertilizer coulters are installed. The seed drill is a 25-year-old machine without any electronic control, still in use for farming and in excellent condition. [20]

With both hoppers full, the weight is about 1450 kg, which does not cause any trouble for the tractor to lift or no counter weights are needed. The power of the tractor would be enough for much wider seed drill, but this was the only option available for the trials.

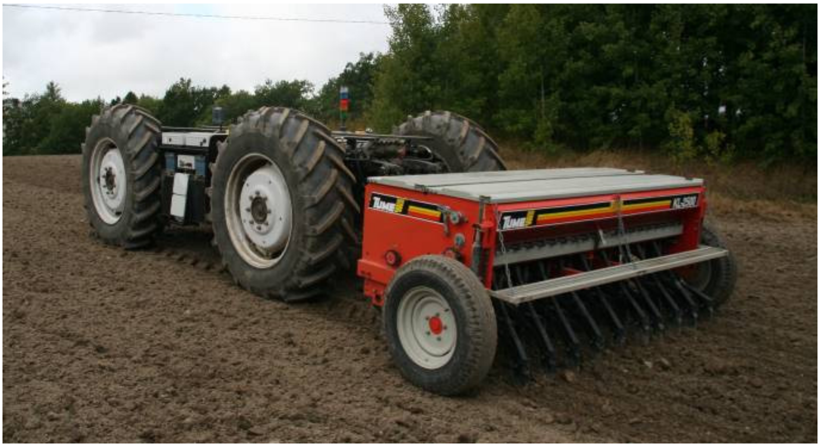


Fig. 2 APU-Module with the seed drill, Tume KL-2500 in the field trial

2.2 Positioning System

The positioning system is based on GNSS technology. The positioning system consists of a) a RTK-GPS receiver, b) a fiber-optic gyroscope in heading and c) an inclinometer for tilt compensation. Trimble 5700 RTK-GPS receiver was used for global positioning; with the virtual base station signal provided by Trimble. The positioning accuracy is claimed to be typically better than 2 cm, but if the view to the GPS satellites is poor, the positioning accuracy is worse. There are several quality indicating values the receiver transmits besides the coordinates: Fix level, RTK correction in action, HDOP (and other DOP values), pseudorange noise statistics and the number of satellites. These values can be used to measure the quality of positioning, to trust on the coordinates the receiver transmits. [20]

For the heading estimation, a fiber-optic gyroscope (KVH DSP-3000) is used together with odometry and GPS heading information. Inertial-Link 3DM-GX2 was used for tilt compensation. All the signals were fused together in an embedded controller; the GPS positioning is corrected to ground level, heading is estimated based on GPS heading, fiber-optic gyro and odometry information and all the relevant information is transmitted to the navigation system by using CAN-bus. [20]

2.3 Guidance System

The desired route shall be given as waypoints that form a polyline. Here it is assumed that the route planner above in the system gives these waypoints 5 seconds in advance before they are passed – in order to utilize prediction. In curves, the waypoints are given frequently, typically a new waypoint is added if the angle deviates more than five degrees, so that the polyline is smooth enough to follow. It is also assumed that the route planner gives feasible waypoints, so that the vehicle kinematic constraints are not violated – for instance the minimum turning radius is considered in the route planner. The used geographic coordinate system is KKJ3/YKJ, a Finnish projection commonly used in cartography.

As the vehicle knows its position and attitude in the global coordinate system by using the sensors, and the waypoints are given in the same coordinate system, the path tracking algorithm needs to consider two error variables: lateral error and angular error.

The path tracking algorithm knows the waypoints it has passed successfully, and the path error computation is computed for the line segment that starts from the last passed waypoint to the next one. However, computing the error variables only for the current position and attitude does not provide enough information for steering controllers, as in curves this kind of vehicle would overshoot remarkably.

The guidance system computes the error variables not only for the current position all the time, but also for all the predicted positions of the vehicle based on the current position, heading, speed and steering angles. The prediction relies on the kinematic model of the vehicle and it takes the dynamics of speed and steering actuators into account in integration. The prediction horizon is a tuneable parameter; the value used in the tests was five seconds. The predicated error variables are converted to single error variables by using weighed averaging; the weighting function was an exponent function; the current state is weighted more than the predicted error five seconds ahead. The weighting factor affects on overshooting vs. cutting corners in the curves.

The structure of the path tracking algorithm is presented in Fig. 3. In the first block the path error is computed including the prediction, described above. The Approach Filter tweaks the error signals in case the lateral error is very large (over one meter): it modifies the angular error signal in order to guide the vehicle quicker to the route. This is helpful in case the automatic guidance is started after manually manoeuvring the vehicle to the field and/or after refilling the hoppers. If the lateral error is less than one meter, the error variables are passed through as is, which is the normal case during the autonomous operation.

Lateral Controller and Angular Controller are controllers in the feedback loop; the structure of both is PID. The feed forward part helps to stabilize heading control. The last block is Inverse Kinematics, which translates the desired lateral speed and

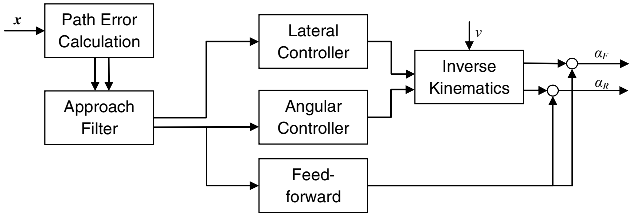


Fig. 3 Structure of path tracking algorithm; the input is the state of robot, the outputs are the steering angle setpoints for the steering servo controller

angular speed commands to the steering angles in front and rear. The so called feed-forward part is used to transfer the angular error to the setpoints of steering angles without dynamic filtering. The outputs are the steering angles in front and rear [20].

2.4 Path Planning

The approach in coverage path planning is semiautomatic. Path planning was done in Matlab, by using two stage strategy: 1) 7 times around the field counter-clockwise to lay headlands (the turning area) and 2) swathing the mid area by following one of the edge trails of the field and using U-turns. The user may give the field boundaries as a polygon and the coverage path planning generates at first the headlands that are laid around the field. In the headland, the driving direction is always either CW or CCW, in order to sow the corners of the field by reversing. Seven times around the field is necessary for this machine in order to generate large enough headland for forward turnings. In the second step the semiautomatic path planning algorithm asks which edge of the field to follow. This was done in order to allow a user to select non-optimal path, in order to compromize with practical issues, like refilling station or stationary position of operator in the field. The path planning algorithm generates waypoints with the information whether tool is up or down (working position) and required ramps for acceleration/deceleration.

3 Results and Discussion

3.1 Fields and Path Planning

The system was tested in four field plots in southern Finland, May 7-8, 2013. The total area of cultivated area was 6.6 ha. The autonomous sowing trial was carried out during the normal sowing season, the harrowing was made appropriately before the sowing for the seed drill. Spring wheat was sown with local fertilizing using the Tume combined fertilizer and seed drill. The hoppers of the seed drill were refilled manually, in the field, by manually guiding the vehicle next to the refilling wagon. With one refill, it was possible to sow about 0.85 ha. The operating speed 1.8 m/s was used in sowing and 1.5 m/s in turning manoeuvres.

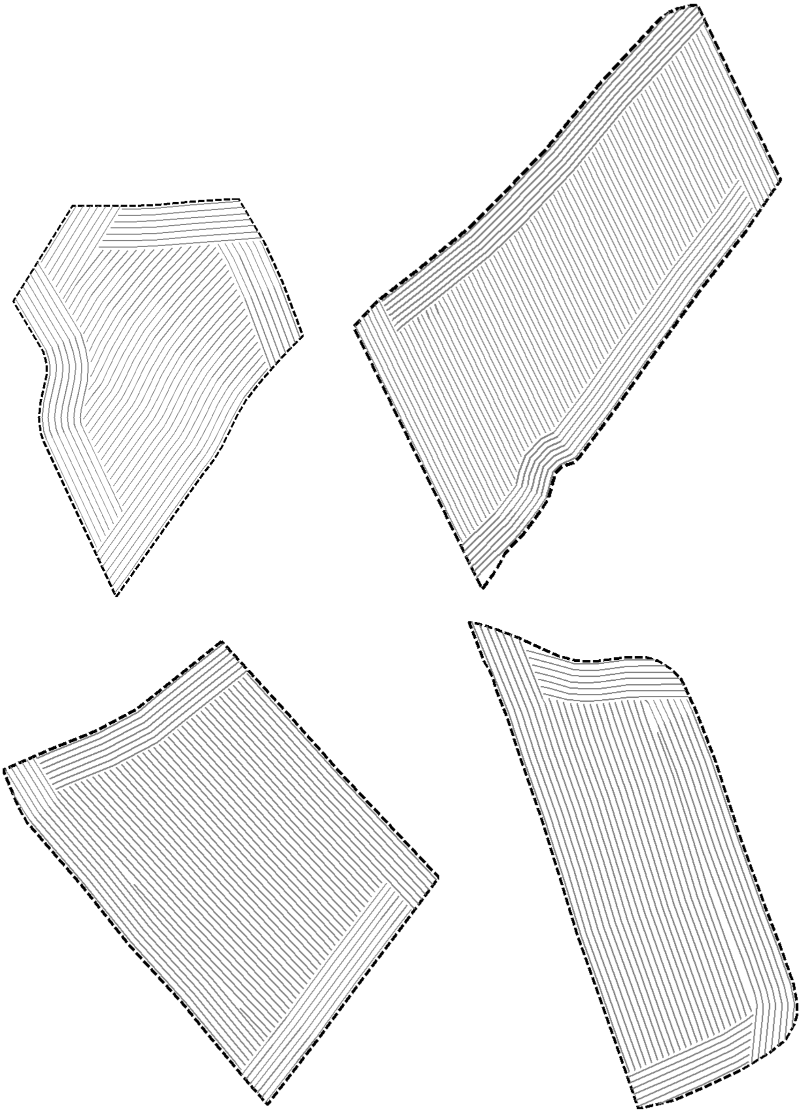


Fig. 4 The field plots used in the field trials. Top: Fields #1 and #2, Bottom: Fields #3 and #4. The planned path is presented in gray lines.

The trial started from Field #1 to Field #4 in order. The field areas are: 1) 1.07 ha, 2) 2.40 ha, 3) 1.76 ha, and 4) 1.36 ha. The path planning is semiautomatic, the user may select from which corner of the field the operation starts. A typical rule of thumb is to follow the longest edge; this was used in this trial in ¾ of the cases. For Field #2 it was more convenient to select short swaths in order to avoid long runs in the large field; this was selected for a practical reason – the human operator needs to be all the time within 50 m distance from the tractor, otherwise the tractor safety system would kill the engine and this strategy allowed the operator to stay in place instead of continuously running behind the machine back and forth. Similarly, the first stage is driven counter-clockwise because the remote controller radio receiver/transmitter is located on the left side of the tractor, so the range is slightly better when the operator stands in the middle of the field during the operation. The fields and the planned paths are presented in Fig. 4.

3.2 Autonomous Sowing in Practice

In theory, the developed system is autonomous. However, there are many practical issues why it was necessary to have at least one human in the field all the time. First of all the hoppers of the vehicle needs to be refilled after every 0.85 ha; this is required more or less every hour with the system (Fig. 5). The other practical reason is the seed drill, which is not developed for autonomous use and in some moist places of the field the coulters were blocked by moist clay soil, which is not uncommon for this type of machines – a human operator was able to see blockages during autonomous operation (as there was nothing else to do) and by interrupting the autonomous mode, lifting the hitch and cleaning the coulters helped; and became a common practise during the trial.

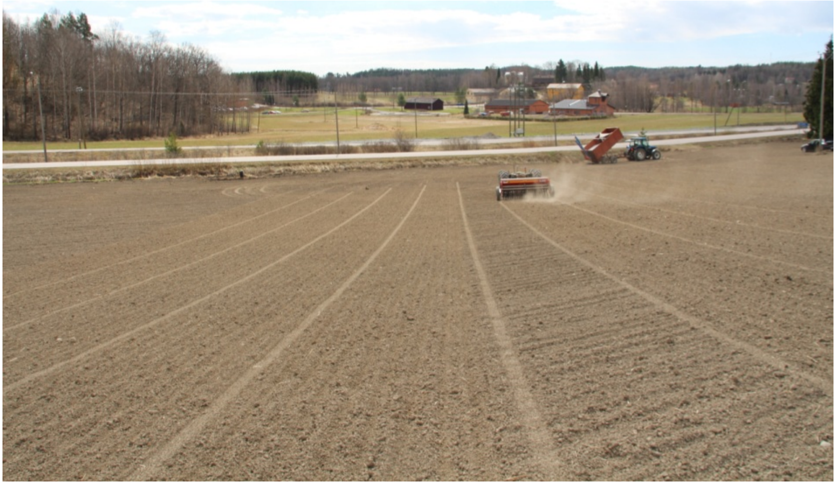


Fig. 5 Autonomous operation going on in Field #2. The refilling wagon is seen on right.

The wireless radio communication between the tractor and operator is another issue. The wireless remote control system, manufactured by Technion Oy, has versatile control knobs and an integrated emergency stop feature. The manufacturer has used 2.4 GHz band in communication; the manufacturer says it is done according to IEEE 802.15.4. The manufacturer sells the remote control system for cranes and other machines in the market and in the design safety has been an important requirement. Therefore, in the wireless system continuous two way communication is alive all the time, and in case of the communication timeout, the system goes to the safe state, which can be programmed in the tractor end. In the tractor, in case of timeout, the receiver module shuts off the electricity from all safety critical ECU's; including the one controlling the diesel engine – the engine shuts down, the drives go idle and the parking brake engages. However, 2.4 GHz band is crowded in urban areas, not necessarily in rural areas like in the fields, but still any unexpected noise is easily created; e.g. by mobile phone (Bluetooth) or computers (WiFi) of the operator; or a WiFi access point nearby. Furthermore, the transmission power is limited, which limits the range to less than 50 m in practice. A closed system operating in 2.4 GHz band, limited range, the nature of wireless communication (scattering, reflection, multipath, diffraction etc.) and the safety feature together caused the tractor to shut down 36 times altogether during two trial days. The restart requires not only restarting the engine, but also resetting the radio communication which typically requires walking closer to the machine; the mean time to recover the system was 52 seconds (range 27-92 seconds).

For instance, Field #3 is surrounded by three houses (see Fig. 6), which is not unusual as the field plots in Finland are rather small. The fields are not in safe from 2.4 GHz noise transmitted by household apparatuses, like WiFi and wireless surveillance cameras.

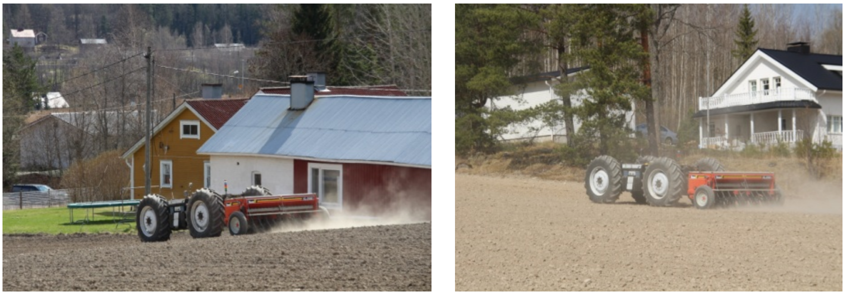


Fig. 6 Houses surround Field #3. The household equipment using 2.4 GHz wireless band may cause noise to radio communication in the field.

Finally, the autonomous vehicles contain a lot of basic technology, not only the standard equipment like engine control, fuel system, hydraulic system but also more electronics and electrical systems compared with conventional vehicles. There is always a risk that some failure appears in the basic system. This risk realized in the second day, in the last field (Field #4), as only 0.5 ha to go, the hydraulic system breakdown (oil leak) forced the trial to end. Therefore, the covered area during the trials was only 6.1 ha.

3.3 Temporal Performance

As explained above, Field #4 was not completed due to the machine breakdown. Otherwise the system was performing in a similar way in all four fields. The Table 1 reveals the statistics when it comes to the interrupts of autonomous operation, the reasons for that. Altogether, the system shut off altogether 36 times during these fields. On average, it takes 52 seconds to recover the system, so about a half an hour was spent on that during the trial days. Manual interventions were done 35 times, usually to clean the coulters or check the level of hoppers (seeds & fertilizer), or to adjust the seed drill settings.

RTK-GPS signal quality dropped under the required level (the number of satellites, HDOP, RTK fix status, STD major axis of pseudorange noise statistics) altogether 30 times, which is worse behaviour than unintended shutdowns as the average recovery time is longer. The total time required to recover from GPS signal quality level drops was 1h27min in four fields together – it would have been even larger if not helping the vehicle couple of times manually away from forest corners where trees shadowed GPS receiver. Generally, the reason inside the GPS receiver was not losing the correction signal for RTK, but the loss of GPS satellites tracking. The median number of satellites was seven and with less than five satellites in view, the positioning accuracy drops below an acceptable level.

Generally, the number of interruptions to the autonomous operation was pretty high, varying from 11-20 times per hectare, as seen in Table 1. However, the variation in the length of continuous autonomous pieces in time is high, the longest period was 1223 seconds (20 minutes) and the shortest was 2.8 seconds. The typical length of a continuous work period is 4-5 minutes.

Table 1 Temporal statistics of autonomous operation in the fields

	Field #1	Field #2	Field #3	Field #4
Area (ha)	1.07	2.40	1.76	1.36
Operated area (ha)	1.07	2.40	1.76	0.85
# Unintended shutdown	6	10	13	7
# Manual intervention	6	10	11	8
# GPS signal quality lost	3	7	11	9
- of which correction lost	1	1	1	1
- too few satellites / HDOP	2	6	10	8
Total # interrupts in autonomy	15	27	35	24
Area / # Total interrupts	14.0	11.2	19.9	17.6
GPS recov. time, average (s)	72	143	175	155
GPS recov. time, max (s)	108	674	841	346
Uninterrupted work, aver. (s)	233	274	147	111
Uninterrupted work, max (s)	1223	818	880	611

As seen in the results, it is crucial to take the GPS signal quality, properties and breaks into a consideration in the development process of a guidance system and algorithms. Simulation of GPS error is crucial in order to take all the cases into consideration as it is hard to repeat the conditions in the field. [22] presents a simulator model capable creating similar noise and quality signals than happened in real life.

3.4 Spatial Accuracy

In coverage operation, a general requirement is to keep the parallel swaths 12.5 cm apart from each other in order to create seamless crop rows. The hard requirement for navigation accuracy is to avoid parallel rows to overlap, which would mean 12.5 cm deviation from the desired path. From this hard requirement it is possible to lead soft requirements, e.g. the navigation accuracy should be better than ±10 cm, ±7.5 cm or ±5 cm. At the beginning of this study the objective ±10 cm was set, in order to avoid overlapping, as the same was used in [23].

The Fig. 7 shows the angular and lateral errors as a histogram, all four fields together. It can be seen that the lateral error is most of the time within the requires ±10 cm tolerance and heading error is relatively small also, under ± one degree. The mean angular error is -0.06°(standard deviation 0.75°) and the mean lateral error is -0.15 cm (standard deviation 5.91 cm). In lateral error, 90% of the samples are in range ±6.2 cm, and 95% of the samples in range ±9.6 cm.

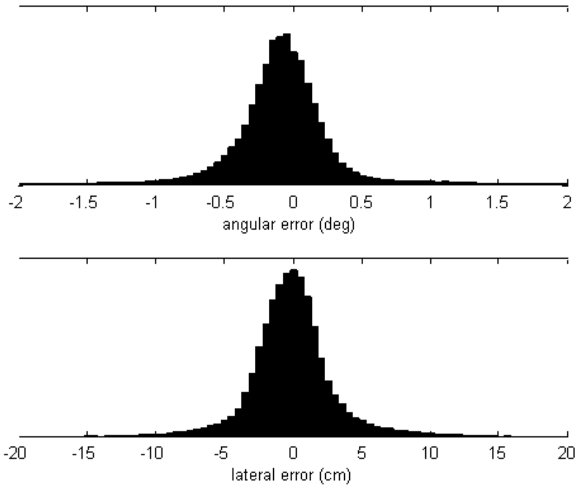


Fig. 7 Histogram of angular and lateral error, respectively

4 Conclusions

This paper presented the results of the autonomous sowing trial in Finland in the spring of 2013. The guidance system is able to navigate in accuracy that is suitable for cereal sowing; the lateral error is typically less than ±10 cm.

It was discussed also the practical challenges that limit the full autonomy. The positioning system based on RTK-GPS receiver is suffering from shadows and other reasons that cause interrupts for reliable positioning signal. Based on the experiences and presented results, RTK-GPS is not suitable alone for an autonomous tractor that should operate without interrupts. Multi technology GNSS is seen as an option to improve the situation, e.g. by using a receiver that utilizes both GLONASS and GPS satellites. Using multiple technologies increases the number of possible satellites in view, but the RTK system would have to provide correction for all signals as well.

Furthermore, the safety system may also limit operational efficiency, if an operator needs to be all the time within 50 meters; if the operator forgets this, the system will shut down or halt – and valuable time during sowing season is spent on recovering the system back online.

Generally, the after some hectares, monitoring an autonomous field robot becomes very boring, as the only thing to do is to stand and watch with a hand on the emergency button and try to be concentrated all the time to see the risks of hazardous movement. Standing a full day in dust and under the burning hot sun is not necessarily more convenient than sitting in a tractor cabin with air conditioning on. Nevertheless, before the full autonomy is achieved and safety issues solved, it is necessary to supervise the system all the time.

In spite of the problems and harms presented, the autonomous sowing trial was successful and it proved that the developed guidance and control system works in real work.

Acknowledgments. This research was funded by Academy of Finland. The research was supported by MTT Agrifood Research Finland, by giving permission to use their facilities including equipment and agricultural fields in production. Especially the author would like to express his gratitude to Mr. Raimo Linkolehto for all his engineering and maintenance efforts during the development process and assisting in the field; and to Mr. Juha Backman and Mr. Jere Kaivosoja for their support and help to make the positioning system working.

References

1. Cornstable, G., Somerville, B.: A Century of Innovation: Twenty Engineering Achievements that Transformed Our Lives. Joseph Henry Press (2003)
2. Rintanen, K., Mäkelä, H., Koskinen, K., Puputti, J., Sampo, M., Ojala, M.: Development of an autonomous navigation system for an outdoor vehicle. Control Engineering Practice 4(4), 499–505 (1996)
3. Bak, T., Jakobsen, H.: Agricultural robotic platform with four wheel steering for weed detection. Biosystems Engineering 87(2), 125–136 (2004)

4. Heraud, J.A., Lange, A.F.: Agricultural Automatic Vehicle Guidance from Horses to GPS: How We Got Here, and Where We Are Going. ASABE Distinguished Lecture Series 33, 1–67 (2009)
5. Takigawa, T., Sutiarso, L., Koike, M., Kurosaki, H., Hasegawa, H.: Trajectory control and its application to approach a target: Part I. Development of Trajectory control algorithms for an autonomous vehicle. Transactions on ASAE 45(4), 1191–1197 (2002)
6. Blackmore, B.S., Griepentrog, H.W., Nielsen, H., Nørremark, M., Resting-Jeppersen, J.: Development of a deterministic autonomous tractor. In: CIGR International Conference, Beijing, China (2004)
7. Zhou, Q., Want, F., Li, L.: Robust sliding mode control of 4WS vehicles for automatic path tracking. In: Proceedings of Intelligent Vehicles Symposium 2005, pp. 819–826 (2005)
8. Jørgensen, R.N., et al.: HortiBot: A System Design of a Robotic Tool Carrier for High-tech Plant Nursing. Agricultural Engineering International: the CIGR Ejournal IX (July 2007); Manuscript ATOE 07 006
9. Vougioukas, S.G.: Reactive trajectory tracking for mobile robots based on nonlinear model predictive control. In: IEEE International Conference on Robotics and Automation ICRA, Rome, Italy, pp. 3074–3079 (2007)
10. Cariou, C., Lenain, R., Thuilot, B., Martinet, P.: Adaptive control of four-wheel-steering off-road mobile robots: Application to path tracking and heading control in presence of sliding. In: Proceedings of IEEE International Conference on Intelligent Robots and Systems, Nice, France, September 22-26, pp. 1759–1764 (2008)
11. Cariou, C., Lenain, R., Thuilot, B., Berducat, M.: Automatic guidance of a four-wheel-steering mobile robot for accurate field operations. Journal of Field Robotics 26, 504–518 (2009)
12. Nagasaka, Y.: An Autonomous Rice Transplanter Guided by Global Positioning System and Inertial Measurement Unit. Journal of Field Robotics 26(6-7), 537–548 (2009)
13. Ruckelshausen, A.: BoniRob: an autonomous field robot platform for individual plant phenotyping. In: Precision Agriculture 2009 (Proceedings of the European Conference on Precision Agriculture), Wageningen, The Netherlands, pp. 841–847 (2009)
14. Snider, J.M.: Automatic Steering Methods for Autonomous Automobile Path Tracking. Technical Report CMU-RI-TR-09-08, Robotics Institute, Carnegie Mellon University, USA (2009)
15. Bakker, T., Van, K.A., Bontsema, J., Muller, J., Straten, G.: Systematic design of an autonomous platform for robotic weeding. Journal of Terramechanics 47(2), 63–73 (2010)
16. Godoy, E., Tangerino, G., Tabile, R., Inamasu, R., Porto, A.: Networked Control System for the Guidance of a Four-Wheel Steering Agricultural Robotic Platform. Journal of Control Science and Engineering 2012 (2012)
17. Goddard, N.: Royal Shows and Agricultural Progess, 1839-1989. History Today 39(7) (1989)
18. Oksanen, T.: Embedded control system for large scale unmanned tractor. In: 5th Automation Technology for Off-road Equipment Conference (ATOE) / CIGR-AgEng 2012, Valencia, Spain (2012)
19. Oksanen, T.: Path following algorithm for four wheel independent steered tractor. In: 5th Automation Technology for Off-road Equipment Conference (ATOE) / CIGR-AgEng 2012, Valencia, Spain (2012)
20. Oksanen, T., Backman, J.: Guidance system for agricultural tractor with four wheel steering. In: IFAC Bio-Robotics Conference, Sakai, Japan, March 27-29 (2013)

21. Oksanen, T., Linkolehto, R.: Control of Four Wheel Steering Using Independent Actuators. In: Fourth IFAC International Conference Agricontrol 2013, Espoo, Finland, August 28-30 (2013)
22. Backman, J., Kaivosoja, J., Oksanen, T., Visala, A.: Simulation environment for testing guidance algorithms with realistic GPS noise model. In: Proceedings of Agricontrol 2010, vol. 3(1), pp. 139–144 (2010)
23. Backman, J., Oksanen, T., Visala, A.: Navigation system for agricultural machines: Nonlinear Model Predictive path tracking. Computers and Electronics in Agriculture 82, 32–43 (2012)

Robotics for Sustainable Broad-Acre Agriculture*

David Ball, Patrick Ross, Andrew English, Tim Patten, Ben Upcroft, Robert Fitch, Salah Sukkarieh, Gordon Wyeth, and Peter Corke

Abstract. This paper describes the development of small low-cost cooperative robots for sustainable broad-acre agriculture to increase broad-acre crop production and reduce environmental impact. The current focus of the project is to use robotics to deal with resistant weeds, a critical problem for Australian farmers. To keep the overall system affordable our robot uses low-cost cameras and positioning sensors to perform a large scale coverage task while also avoiding obstacles. A multi-robot coordinator assigns parts of a given field to individual robots. The paper describes the modification of an electric vehicle for autonomy and experimental results from one real robot and twelve simulated robots working in coordination for approximately two hours on a 55 hectare field in Emerald Australia. Over this time the real robot 'sprayed' 6 hectares missing 2.6% and overlapping 9.7% within its assigned field partition, and successfully avoided three obstacles.

1 Introduction

The current trend in agriculture is to increase the farmer's productivity by using larger machinery combined with controlled traffic farming, which is where the

David Ball · Patrick Ross · Andrew English · Ben Upcroft · Gordon Wyeth ·
Peter Corke
School of Electrical Engineering and Computer Science
Queensland University of Technology (QUT)
e-mail: david.ball@qut.edu.au

Tim Patten · Robert Fitch · Salah Sukkarieh
Australian Centre for Field Robotics The University of Sydney

* This work was supported in part by the Australian Research Council Linkage Project LP110200375 "Robotics for Zero-Tillage Agriculture" awarded to QUT, SwarmFarm Robotics and The University of Sydney.

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_30, © Springer International Publishing Switzerland 2015

vehicles traverse exactly the same paths using precision guidance. However, the growth in complexity, size and weight of agricultural equipment, combined with repeatedly traversing the same path, has led to concentrated soil compaction damage as well as longer disruptions due to single machine failures. Soil compaction and single points of failure ultimately decrease yield and productivity. The goal of this project is to create a new class of machines for sustainable agriculture that will increase broad-acre crop production and reduce environmental impact; small, light, inexpensive robots that coordinate as a team to manage the fields and work 24 hours a day. This represents a movement back towards a time when large numbers of human workers would tend the fields and provide individualised plant treatment.

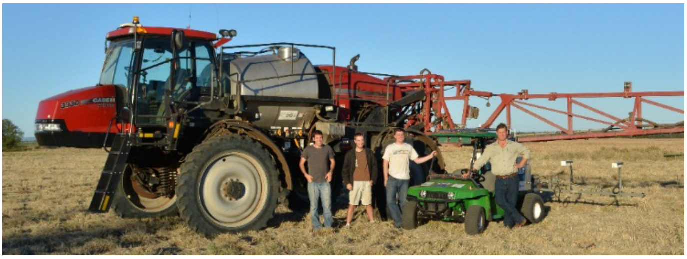


Fig. 1 The small robot platform (green) shown next to a typical farm spray machine (red) with humans for scale. While the robot shown here is a research platform it is approximately the size envisaged by the work in this paper.

This project is focussed on an immediate problem facing farms in Australia – resistant weeds. *Zero-tillage* agriculture, where soil disturbance is kept to a minimum, is considered best practice farming in Australia to reduce topsoil erosion. However, to compensate for removing a mechanical means of weed destruction, farmers typically use more herbicide to manage weeds which has led to the emergence of resistant weeds. The magnitude of the issue in Australia is that the agricultural cost of weeds alone is in the vicinity of $4 billion per annum [1]. Our solution to combat increasing weed resistance is to introduce multiple lighter machines that are able to be deployed into the field rapidly after a rain event as they are less prone to being bogged, cause less soil damage, and operate as a system that is more robust to individual machine failures. See Fig. 1 for a comparison in size between existing farm machinery and the robot platform used in this study. While the robot shown in the figure is a test platform this is approximately the size envisaged for the approach described in this paper.

Since our approach is based on multiple robots, to keep the overall system affordable, the goal is to use relatively low cost sensors for obstacle detection and pose estimation. Cameras are preferred over lasers as they provide rich and informative snapshots of the surrounding environment at high rates. This paper describes a study into using this system to ‘spray’ a large field using one real robot and twelve simulated robots using low cost sensors to track pose and detect obstacles typical to a farm.

Coverage planning is a topic that has a rich history in the robotics community [17]. Recent work has studied coverage in the agriculture context in simulation and addressed the problem of choosing an optimal track orientation [18–20]. In this paper, the existing planting patterns determine the track orientation in advance and therefore basic methods of coverage may be applied. Our work involves simulation to illustrate the behaviour of many robots, but our focus instead is on the whole-system aspects, including navigation and perception, that allow these results to be applied in practice.

The next section describes the prior art in robotics related to farm automation. Section 3 describes the design of the robot system. Section 4 describes the experimental setup for the experiments in this paper and then section 5 has the results of the experiments. Section 6 has a discussion of the results and section 7 has concluding remarks.

2 Literature

The chief application of intelligent technology in agriculture has been to increase the accuracy with which a vehicle is guided through a field which allows the principle of controlled traffic farming [2]. Precision guidance has been improved using GPS [3] and vision techniques such as for row following [4, 5], however, these by themselves are not enough for driverless farming as they lack a full navigation system which includes obstacle avoidance.

Early work in developing autonomous farm machinery is described by Ollis and Stentz [6] who use vision for following the boundary between cut and uncut crop and demonstrated harvesting over 48 hectares of crop [7]. To detect obstacles they use probability density functions to threshold novel regions in the images demonstrating preliminary results. Stentz et al. [8] describes a semi-autonomous tractor, which waits for human advice when detecting an obstacle in the path of the robot. A human trains the system by driving the relevant routes and the robot uses pure pursuit [9] to follow the path. To detect obstacles they combine a neural network processing single images with a stereo camera system. Torii et al. [10] reviews a number of approaches including using neural networks, genetic algorithms, and fuzzy logic for robots capable of tillage, planting and plant care.

Since then there have been numerous approaches to the robotics farm with a significant focus on adding a variety of sensors to existing tractors. Johnson et al. [11] describe a complete multi-robot system demonstrated working over a long period of time. To detect obstacles the tractor has colour cameras, infrared cameras and a nodding laser which generates tagged 3D data. Recent long term work in an orchard, with a similar set of sensors added to an existing tractor, demonstrates substantial productivity improvements of 30% over human operated machinery [12].

Some groups have focussed on the development of custom platforms. Bakker et al. [13] describe a systematic approach to the design of an autonomous platform for robot weeding considering a large range of issues such as: methods to detect the weeds and destroy them, guidance technique, energy sources and vehicle type. The result of their design process is a four wheel driven and steered mechanical weeder guided by vision and GPS. Another example is the BoniRob [14] which can vary its height and track width to suit different fields and was designed for sensing the state of the crop. Hortibot [15] and Armadillo [16] are designed as custom generic flexible tool carriers.

3 System Design

The overall system, shown in Fig. 2, consists of a centralised multi-robot planner that takes regions for spray coverage and assigns each robot a section to cover. The multi-robot planner sends each robot the perimeter and a list of waypoints that define each pass. The robots regularly send back their global locations. The system is designed to allow each robot to operate autonomously within its assigned section of the field, independent of the multi-robot planning system, for extended periods of time. The system uses a 3G mobile data connection to communicate over the internet between the multi-robot planner and the robots. Mobile data may dropout, in particular in rural areas, however the system is designed to gracefully handle long and regular periods of communication drop out. The following sections describe the robot system including its sensors, and then describe each of the main software components.

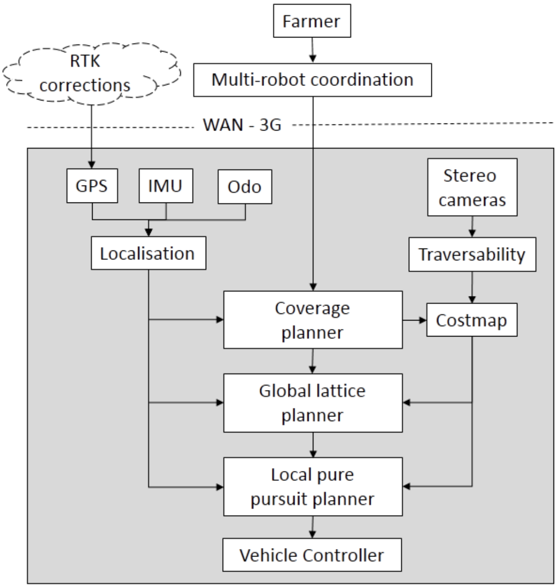


Fig. 2 System architecture. The farmer interacts with the multi-robot coordination module which then communicates with the rest of the system over a 3G mobile data connection via the internet.

3.1 Robot Platform

The agricultural robot platform is based on a John Deere TE Gator modified for autonomous operation. The Gator is an Ackermann steered 48V 4.6kW electric vehicle, approximately 2.6 meters long and 1.5 meters wide, with a ground clearance of 0.18 meters. The vehicle has a nominal 8 hour battery life and 12 hour recharge time. The Gator is fitted with an automation conversion kit supplied by RoPro Design. The computer controls the vehicle over a Controller Area Network (CAN) bus by intercepting the control lines that feed into the Gator's standard motor controller and adding a smart motor to control the steering wheel and another smart motor to control the brake. Via a single switch the vehicle can be changed between autonomous and manual human driven modes. For emergencies, a pneumatic emergency brake has also been fitted to the robot and can be released via local estop buttons or wirelessly. The robot has been fitted with a 200L spray tank, 5 meter wide boom with spray nozzles and a commercial plant detection system, WeedIT. The robot receives Real Time Kinematic (RTK) precision correction data from the SmartNet Australian Continually Operating Reference Station (CORS) network. The robot has several sensors.

- Two forward facing iDS UI-5240CP Power over Ethernet GigE cameras with wide field of view lens (~US$1,400 each).
- Two quadrature encoders S63B-37ADQ-OCCP4-AF mounted on the rear wheels providing a resolution of 6mm (~US$250 each).
- A CH Robotics UM6 Inertia Measurement Unit (IMU) (~US$200).
- A Skytraq S1315F-RAW GNSS with Tallysman TW3100 (~US$300).

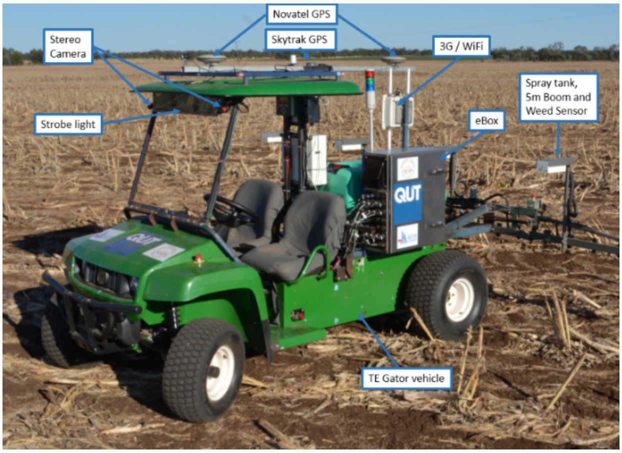


Fig. 3 The agricultural research platform, used to test the sensors and algorithms, shown on a sorghum stubble field. The base vehicle is an electric John Deere TE Gator. The strobe light was used for night time studies not described in this paper.

3.2 Software and Hardware Details

The robot runs the Robot Operating System (ROS) [21] framework which uses a topic-based publish and subscribe model. Local and remote nodes communication over topics using pre-defined messages. The robot has two standard off the shelf computers running Ubuntu 12.04 and ROS Fuerte, one that runs the traversability node and the other for localisation, path planning and vehicle control. The traversability node sends information about obstacles using a ROS *PointCloud* message. The path planner communicates the desired motion using ROS *ackermann_msgs*. The path planners use the ROS *move_base* and *costmap* framework. Most of the nodes on the robot operate at 10 Hz.

A separate laptop is used to run the multi-robot planner and communicates with the rest of the system over 3G. Potential problems with 3G include narrow bandwidth, delays and network failures; however, this isn't a problem as the communication between the multi-robot planner and the robots is limited to sparse commands and status updates. To handle communication failures and provide namespace separation, there are two ROS masters, one is on a laptop computer, and another is on the robot. These communicate using custom messages over a ROS actionlib interface and relevant topics are connected using ROS foreign_relays.

3.3 Multi-robot Planner

The task of weed management through controlled herbicide delivery is algorithmically an instance of the *coverage* problem [22]. This section describes the planning subsystem for multi-robot coverage of large fields.

The goal of coverage is to plan a path such that the robot(s) eventually visit (cover) all points within a defined area. Finding an optimal coverage path is related to the well-known Travelling Salesman Problem (TSP) and is NP-hard [23]. However, this application is a restricted case of coverage where robot motion is constrained to travel parallel to pre-existing rows within a field. In this case, cell-decomposition algorithms work well in practice.

Following [22], we apply the *boustrophedon decomposition*, where the coverage area is exactly partitioned according to a back-and-forth (lawnmower) pattern. For a single field with crop rows at a known row orientation, the boustrophedon decomposition is computed using a sweep-line moving perpendicular to the rows. This method partitions the paddock into cells of approximately-equal area measured by the sum of row lengths. One robot is allocated to each cell. The robot's path within its cell is determined by the existing row pattern which can be obtained from satellite data or from the farmer. The number of cells within a field is determined by the number of available robots. The initial decomposition and allocation of cells to robots is computed offline given the field boundaries. For this, the system calculates a list of waypoints located at the start and end of each pass. Then, each robot performs row-following and obstacle avoidance within its allocated cell using the algorithms described in the following sections. Cells may also be allocated to simulated robots that operate simultaneously with real robots, although the simulated robots drive directly to waypoints without obstacle avoidance.

3.4 Coverage Planner

This node, which runs on each robot, is responsible for ensuring the robot follows the coverage plan waypoints provided by the multi-robot planner. It sequentially provides the next goal waypoint located at the end of each row. The node also sends a *funnel* field to the costmap to ensure that even while avoiding obstacles the robot will stay close to the row. The funnel field strength is proportional to the distance from the desired path and so forms an inverted triangle. So that overall progress can be tracked, this node regularly sends the multi-robot planner the global location of the robot.

3.5 Pose Estimation

For localization, a particle filter combines sensor information from several low-cost sensors, specifically a single-channel GPS, IMU and velocity from the robot's wheel encoders. The particle filter allows the robot to determine its pose based on a series of noisy readings and allows the robot to continue to operate even if GPS drops out for a period of time. GPS position calculations are performed on the robot's PC using the free software library RTKLIB [24] which applies RTK corrections to the raw GPS satellite observations to dramatically improve the robot's global accuracy. The robot receives corrections from the CORS network via a 3G internet connection.

3.6 Traversability

This node is responsible for determining the traversability of the area in front of the robot using only vision. Currently, the node detects obstacles, which is a subset of traversability which would include determining the terrain type.

The node firstly determines novel regions in the left camera image and then processes these novel regions using stereo vision. To determine novel regions the node maintains a model of the typical appearance of the field, under the assumption that obstacles typically deviate significantly from this appearance model. Candidate obstacles are detected in image space by looking for novel image regions with respect to this model. Novelty detection uses a weighted variant of Parzen windows [25] where samples lose weighting over time. Fast inference on this model is performed using the Fast Library for Approximate Nearest Neighbours (FLANN). Candidate obstacle image regions are then passed through stereo matching in order to generate a metric point cloud of the candidate obstacles. Stereo matching was performed using LIBELAS [26]. The point cloud is filtered based on their height and distance from the camera, and the remaining points are obstacles.

This two-step process has advantages over purely stereo matching-based obstacle detection. The first advantage is that stereo matching typically performs poorly in agricultural fields due to its highly repetitive nature, hence stereo matching alone generates significant false positives. Empirical results indicated that stereo matching is more robust on obstacles over stubble (based on the assumption that their appearance deviates from the typical), since obstacles provide strong edges which are useful for stereo matching. The novelty pre-processing then implicitly encodes an understanding of the image regions in which stereo matching performs well, and significantly reduces false positives while having negligible impact in terms of false negatives. The second advantage is that since stereo matching is only performed on a subset of the image regions, it typically has a reduced computational load for equivalent performance.

3.7 Costmap

The costmap maintains a 2D representation of the environment surrounding the robot and the global lattice planner and local pure pursuit planner use it to generate paths. As is typical, the map encodes the cost of the robot occupying particular cells. To ensure that obstacles are avoided and the robot remains on the desired row, the obstacles are given high values and the funnel from the coverage planner is given low to medium values. Specifically, each cell is 0.2 meters squared and the costmap is 100 meters squared centered on the robot. The costmap is aligned with the known direction of the crop rows to allow the system to plan smooth straight paths.

3.8 Global Lattice Planner

This node plans a long-term path to the coverage planner's goal pose through the costmap. This node uses the Search Based Lattice Planner (SBPL) [27] to generate collision free paths around obstacles. SBPL incrementally searches for the best path considering the cost of motion primitives and the cost of traversing the costmap cell. Obstacles have high cells costs. The funnel provided by the robot's coverage node, provides increasing costs perpendicular to the desired row, and therefore ensures that SBPL will generate paths that, after avoiding an obstacle, will guide the robot back onto the correct row. SBPL constructs the path using motion primitives specific to the Gator vehicle. These motion primitives represent the Gator vehicle's kinematics, in particular the minimum radius of curvature, and include travelling straight and turning left and right. These plans extend in the direction of the goal and are clipped to the extent of the costmap.

For typical row following without obstacles this approach is more complex than required however provides flexibility for the future functionality such as moving between fields. The global lattice planner recalculates a new path every 10 seconds, or when the goal changes or when the local planner rejects the plan.

3.9 Local Pure Pursuit Controller

This node is responsible for ensuring that the robot tracks the long-term lattice path using a pure pursuit controller [9]. The node has two proportional integral controllers to minimize the error in the distance between the robot and the lattice path and between the robot's heading and the lattice path orientation. The pure pursuit controller plans the path of the robot forward several seconds checking for collisions between the robot footprint and obstacles in the costmap. If a collision is detected then the local controller rejects the global path and the robot slows down while the global lattice planner generates a new path. In practice, as the global lattice planner quickly generates paths, the robot does not slow down notably.

3.10 Vehicle Controller

The vehicle controller node manages the low-level state of vehicle including the forward velocity of the robot, the steering wheel angle and the state of the brake. The controller interfaces with the vehicle over a CAN bus to smart motors and relays. The node has a hand-tuned proportional-integral-feedforward velocity controller based on feedback of the robot's velocity from the wheel encoders.

4 Experimental Setup

The user defines the boundary of the field by manually selecting appropriate latitude and longitude coordinates as shown in Fig. 4. The area represented by the coordinates in this experiment are slightly smaller than the actual field to avoid the headland which is where the large farm machinery turns around at the end of each row. The area defined for this experiment is 55 hectares of broad-acre sorghum stubble field located in Emerald, Australia. In the section of the field assigned by the multi-robot planner to the real robot there are three obstacles typically found on a farm and include as shown in Fig. 5: a grey electricity pole which is a permanent fixture, a small utility vehicle, and a human. The boundary, locations of these obstacles and a contour bank are shown marked on the field in Fig. 6.

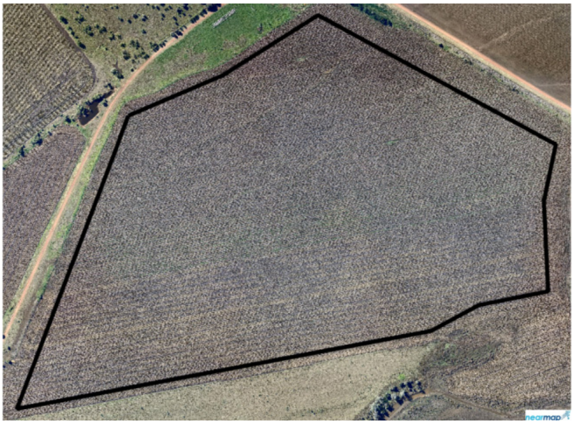


Fig. 4 A 55 hectare sorghum stubble field with the boundary for coverage by the multi-robot system marked in black. Imagery © 2013 Nearmap.

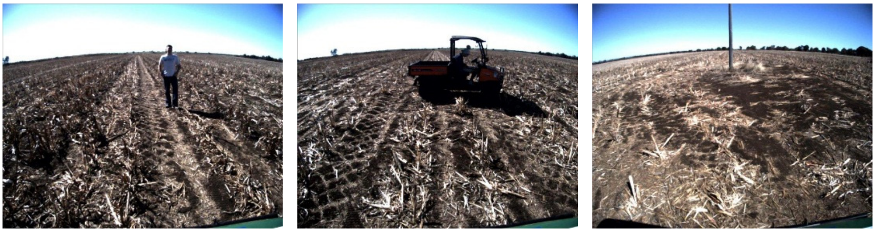


Fig. 5 Three obstacles typically found on a farm: an adult human (left), a utility vehicle (middle) and an electricity pole (right)

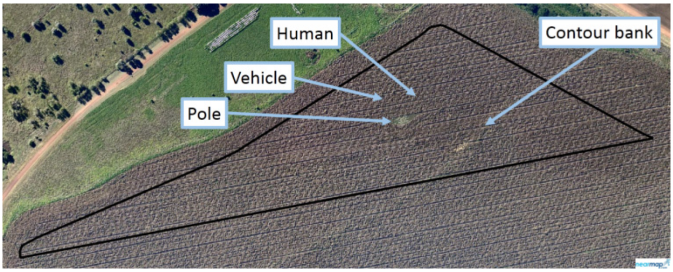


Fig. 6 The boundary assigned to the real robot by the multi-robot planer showing the location of the obstacles and the ridgeline. Imagery © 2013 Nearmap.

The robot's forward velocity was limited to 5 km/hr for this experiment which is the speed specified by the farmer as optimum for spraying weeds. This speed is lower than the large manned spray vehicle, however they drive faster than they would like so as to increase their own productivity at the cost of suboptimal spraying. With a 5 meter wide spray boom travelling at 5 km/hr it would take one robot approximately 26 hours to traverse this 55 hectare field. For this experiment the multi-robot planner was configured to use 1 real and 12 simulated robots.

The goal is to cover the entire field only once while minimizing the missing and overlapped areas. Therefore, the pass-to-pass error is calculated based on the perpendicular distance to the previous vehicle pass, ignoring the turning regions at the ends of the rows and avoiding obstacles. The user configured the multi-robot planner with a pass-to-pass distance of 5 meters to generate the waypoints for the robots. The coverage results are based on a 5.5 meter wide boom due to the side spray from the nozzles on the ends of the boom. A high-performance Novatel multi-constellation GNSS with tactical grade IMU and dual antennas provides ground truth data, and its positions are used for calculating coverage results and for plotting the paths.

5 Results

Fig. 7 illustrates the performance of the path planning nodes in avoiding one of the obstacles detected using only vision. The costmap shows the obstacles added at the correct location. Some detection of obstacles is made at 10 meters away while robust detection occurs at 5 metres. The combination of the funnel and the obstacle means that the global lattice planner determines a path that avoids the obstacle and guides the robot back onto the original row path. The obstacle detection and path planning results are similar for the other two obstacles.

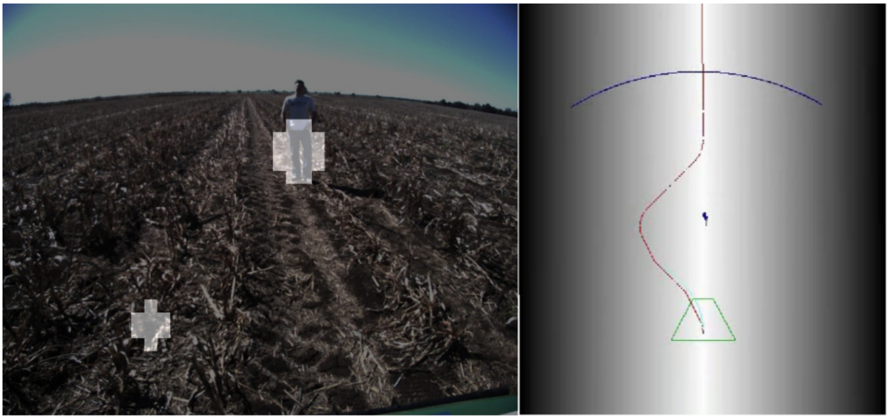


Fig. 7 Demonstration of the robot avoiding a human during the coverage task. The camera image showing the position of the human and highlighted regions of the image that the traversability node has detected as novel (left). The costmap and path planner outputs are shown on the (right). The dark blue shows the output from the traversability node showing that the human is added as an obstacle and that the stubble detected as novel is not added as an obstacle. The funnel gradient, in greyscale, can be clearly seen biasing the robot's path back onto the row. The green outline represents the robot's footprint. The red line is the global lattice path and the small cyan line is the pure pursuit plan.

The paths taken by the real and simulated robots to cover the assigned area are shown as different colours in Fig. 8. All of the robots completed their assigned section in 1.8 hours. Unsurprisingly, the simulated robots precisely traverse between waypoints and cover their sections exactly. The coverage for the real robot is shown in Fig. 9 showing the area that is covered, missed and overlapped. The extra overlap where the robot transitions onto the next pass is due to a slightly asymmetric bulb turn.

The three large deviations shown in the path and the coverage are the successful avoidance of the true positive obstacles. The robot also slightly deviated from the path to avoid a false positive obstacle shown as a small red region in Fig. 9. This false obstacle was detected just as the robot was driving down the far side of a contour bank. The traversability node detected a small region of stubble as novel and because the stubble was higher in the world it was briefly determined to be an obstacle. Table 1 shows the final coverage results. The real robot covered 6 hectares in 1.8 hours and missed 2.7% of the area and overlapped 9.7% of the area. The RMS pass-to-pass error was a low 0.18 m.

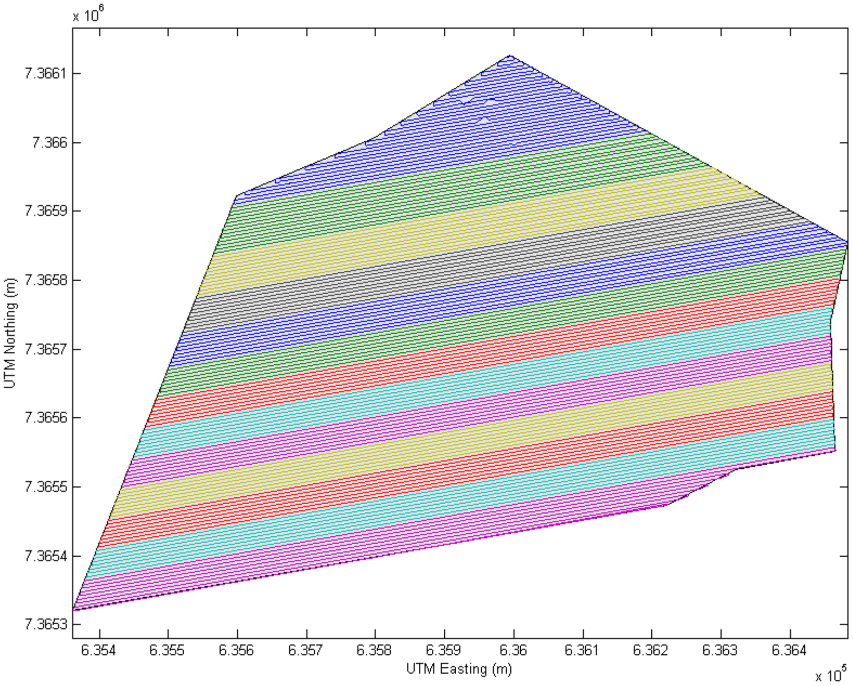


Fig. 8 Coverage of the entire 55 hectare area using 1 real and 12 simulated robots shown in different colours. The result shows that each robot was assigned approximately the same distance to travel. The multi-robot planner assigned the real robot the top section (blue). The real robot's three large deviations are due to true obstacles and the small deviation is due to a single false positive obstacle.

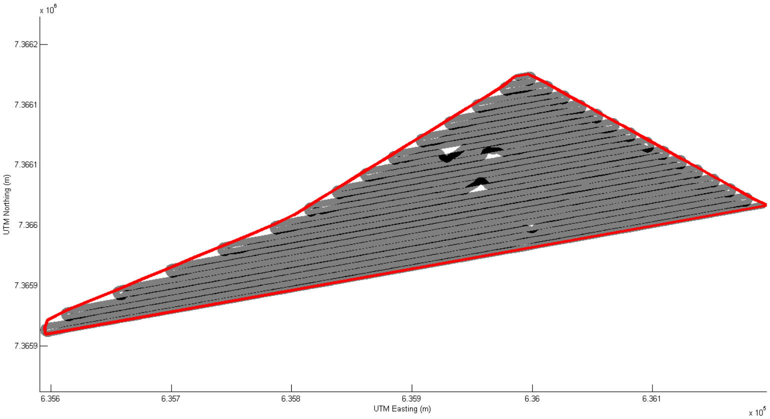


Fig. 9 This plot shows the coverage for the real robot (grey) within its assigned boundary (red) including the parts of the field where the spray would be overlapped (black) and missed part of the field (white). The top three large missed and overlapped areas are true positive obstacles that were successfully avoided. The small area towards the bottom in the middle was where the robot slightly deviated to avoid a false positive obstacle.

Table 1 These results represent the overall performance of the real robot for the coverage task

Result	Metric
Missed percentage	2.6%
Overlap percentage	9.7%
True positive obstacles avoided	3
False positive obstacles avoided	1
Real robot run time	1.8 hours
Area covered	6 hectares
RMS pass-to-pass error	0.18 meters

6 Discussion

The robot autonomously ‘sprayed’ 97.4% of its section of the field while avoiding obstacles typical to the farm environment. The real robot was shown to work alongside many simulated robots to perform complete coverage of a large area. The robot was able to globally localise to perform the coverage task with a precision only marginally worse than provided by commercial agricultural solutions. This was achieved using a particle filter to fuse inexpensive odometry, IMU and GPS sensors combined with the open source RTKLIB and a correction signal. The missed and overlapped percentages can be traded depending on the cost of herbicide and the loss due to missed areas by changing the desired pass-to-pass value relative to the spray width.

The results demonstrate that the obstacles were successfully added to the costmap. In the sorghum field, by themselves the stereo vision and novelty detection methods generated many false positive obstacles. However, using the novelty detector to identify regions for stereo matching overcame problems with ambiguity in the appearance and ground plane. The contour banks were higher than the system was designed to handle resulting in a false positive obstacle. While the robot briefly deviated from the row to avoid a false obstacle, the system was tuned to ensure detection of true positive obstacles at the expense of some false obstacle detection.

The navigation system generated suitable paths to avoid obstacles and guide the robot back onto the correct row. In particular, the combination of the lattice planner generating kinematically suitable paths around obstacles and through the funnel proved successful. A simpler navigation system could have been used for the exact experiment described in this paper. However, the benefit is that this system allows for a wide range of flexibility for adding future functionality, changing sensors or handling increasingly complex navigation challenges.

The robot platform, while sufficient for this experiment, will be unsuitable for commercial deployment in broad-acre fields. Due to the existing large machinery, the terrain is rugged and the gator bounced around during turns and while avoiding obstacles. To address the rugged terrain for this experiment the tire pressure was lowered which increased the cost of transport, and so is not a suitable long term solution.

There are several areas for future work. The multi-robot planner will be updated to adaptively handle robots that are unable to complete their assigned sections due to robot failure or very large obstacles. A vision based docking system will be added to so that the robots can autonomously recharge power and herbicide. Lastly, there will be continued development of more robust techniques for detecting traversability and estimating the robot's pose.

7 Conclusion

The paper has described a new approach to increasing broad-acre agricultural productivity with small affordable autonomous robots. This will lead directly to improved productivity through reduced soil compaction and specifically for weed management, reduced herbicide usage through smarter local application, providing direct environmental benefits. The technology will lower production costs through more timely interventions and the increased robustness and incremental scalability inherent in multiple small machines.

References

1. Sinden, J., Jones, R., Hester, S., et al.: The economic impact of weeds in Australia. CRC for Australian Weed Management (2004)
2. Taylor, J.: Benefits of permanent traffic lanes in a controlled traffic crop production system. Soil and Tillage Research 3, 385–395 (1983)
3. Bell, T.: Automatic tractor guidance using carrier-phase differential GPS. Computers and Electronics in Agriculture 25, 53–66 (2000)
4. Reid, J.F., Searcy, S.W.: Vision-based guidance of an agricultural tractor. IEEE Control Systems Magazine, 39–43 (1987)
5. Billingsley, J., Schoenfisch, M.: The successful development of a vision guidance system for agriculture. Computers and Electronics in Agriculture 16, 147–163 (1997)
6. Ollis, M., Stentz, A.: Vision-based perception for an automated harvester. In: Proceedings of the IEEE International Conference on Intelligent Robot and Systems, pp. 1838–1844 (1997)
7. Pilarski, T., Happold, M., Pangels, H., et al.: The demeter system for automated harvesting. Autonomous Robots 13, 9–20 (2002)
8. Stentz, A., Dima, C., Wellington, C., et al.: A System for Semi-Autonomous Tractor Operations. Autonomous Robots 13, 87–104 (2002)
9. Amidi, O.: Integrated mobile robot control. Carnegie Mellon University (1990)
10. Torii, T.: Research in autonomous agriculture vehicles in Japan. Computers and Electronics in Agriculture 25, 133–153 (2000)
11. Johnson, D.A., Naffin, D.J., Puhalla, J.S., et al.: Development and implementation of a team of robotic tractors for autonomous peat moss harvesting. Journal of Field Robotics 26, 549–571 (2009)

12. Moorehead, S.J., Wellington, C.K., Gilmore, B.J., Vallespi, C.: Automating orchards: A system of autonomous tractors for orchard maintenance. In: Proceedings of the IEEE International Conference on Intelligent Robot and Systems (2012)
13. Bakker, T., Asselt van, K., Bontsema, J., et al.: Systematic design of an autonomous platform for robotic weeding. Journal of Terramechanics 47, 63–73 (2010)
14. Ruckelshausen, A., Biber, P., Dorna, M., et al.: BoniRob: an autonomous field robot platform for individual plant phenotyping. Precision Agriculture 9, 841 (2009)
15. Jorgensen, R.N., Sorensen, C.G., Maagaard, J., et al.: HortiBot: A System Design of a Robotic Tool Carrier for High-tech Plant Nursing. Agricultural Engineering International 9 (2007)
16. Jensen, K., Nielsen, S.H., Joergensen, R.N., et al.: A low cost, modular robotics tool carrier for precision agriculture research. In: Proceedings of the 11th International Conference on Precision Agriculture (2012)
17. Choset, H.: Coverage for robotics–A survey of recent results. Annals of Mathematics and Artificial Intelligence 31, 113–126 (2001)
18. Oksanen, T., Visala, A.: Coverage path planning algorithms for agricultural field machines. Journal of Field Robotics 26, 651–668 (2009)
19. Jin, J., Tang, L.: Coverage path planning on three-dimensional terrain for arable farming. Journal of Field Robotics 28, 424–440 (2011)
20. Hameed, I.A.: Intelligent coverage path planning for agricultural robots and autonomous machines on three-dimensional terrain. Journal of Intelligent & Robotic Systems, 1–19 (2013)
21. Quigley, M., Gerkey, B., Conley, K., et al.: ROS: an open-source Robot Operating System. In: International Conference on Robotics and Automation (2009)
22. Choset, H., Lynch, K.M., Hutchinson, S., et al.: Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press, Cambridge (2005)
23. Oksanen, T., Visala, A.: Coverage path planning algorithms for agricultural field machines. Journal of Field Robotics 26, 651–668 (2009)
24. Takasu, T., Yasuda, A.: Development of the low-cost RTK-GPS receiver with an open source program package RTKLIB. In: International Symposium on GPS/GNSS (2009)
25. Parzen, E.: On estimation of a probability density function and mode. The Annals of Mathematical Statistics 33, 1065–1076 (1962)
26. Geiger, A., Roser, M., Urtasun, R.: Efficient Large-Scale Stereo Matching. In: Kimmel, R., Klette, R., Sugimoto, A. (eds.) ACCV 2010, Part I. LNCS, vol. 6492, pp. 25–38. Springer, Heidelberg (2011)
27. Cohen, B.J., Chitta, S., Likhachev, M.: Search-based planning for manipulation with motion primitives. In: 2010 IEEE International Conference on Robotics and Automation, pp. 2902–2908. IEEE (2010)

A Pipeline for Trunk Localisation Using LiDAR in Trellis Structured Orchards

Suchet Bargoti, James P. Underwood, Juan I. Nieto, and Salah Sukkarieh

Abstract. Autonomous operation and information processing in an orchard environment requires an accurate inventory of the trees. Individual trees must be identified and catalogued in order to represent their distinct measures such as yield count, crop health and canopy volume. Hand labelling individual trees is a labour-intensive and time-consuming process. This paper presents a trunk localisation pipeline for identification of individual trees in an apple orchard using ground based LiDAR data. The trunk candidates are detected using a Hough Transform, and the orchard inventory is refined using a Hidden Semi-Markov Model. Such a model leverages from contextual information provided by the structured/repetitive nature of an orchard. Operating at an apple orchard near Melbourne, Australia, which hosts a modern Güttingen V trellis structure, we were able to perform tree segmentation with 89% accuracy.

1 Introduction

Information gathering and processing are becoming increasingly important in horticulture as we try to keep up with demand increases. Having accurate information such as crop health, yield estimates and tree counts aid in optimising control processes and therefore allows for better farm management. Efficient chemigation, fertigation and fruit thinning processes then ultimately lead to maximising yield count.

Information gathering is already commonly used by farmers, but it is a time-consuming and labour-intensive task. Additionally, the data is extrapolated through significant sub-sampling, where a farmer uses their judgement to pick a few trees that best represent the average over the entire farm. The future of farm robotics and information systems will involve making key measurements for *all* trees in a timely and accurate manner.

Suchet Bargoti · James P. Underwood · Juan I. Nieto · Salah Sukkarieh
Australian Centre for Field Robotics (ACFR), The Rose Street Building J04,
The University of Sydney NSW 2006, Australia
e-mail: {s.bargoti,j.underwood,j.nieto,
s.sukkarieh}@acfr.usyd.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics,*
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_31, © Springer International Publishing Switzerland 2015

A natural way to store and process this information is to quantise and associate to the standard farm unit, which is the individual tree. This enables all aspects of information processing to be done in a tree by tree topological fashion rather than relying on three dimensional Cartesian mapping. For example, consider change detection for growth rates. We could attempt to visualise this in 3*D* over the entire farm, requiring accurate 3D sampling, vehicle localisation, and scan registration and alignment. Instead, a simpler approach is to estimate local properties in the data such as tree height, canopy volume and yield count, and link them to individual trees in the farm topology.

This paper provides this unit representation by automatically building a tree inventory over a farm. In particular, we present an autonomous pipeline for localising apple tree trunks using a LiDAR mounted on an Unmanned Ground Vehicle. The trunks, being primary components of the trees, can therefore be used to map tree locations. The procedure is designed for an apple orchard located near Melbourne, Australia (Fig. 1), which hosts a plantation structure known as the Güttingen V trellis. Trees are planted on V-shaped trellises rather than the traditional squat formation, which helps with better weight support of the apple limbs, allows for more sunlight for the fruits and easier fruit picking [3]. The orchard also has a top cover netting to prevent damage to trees caused by hail.

The contribution of this paper is the end-to-end pipeline to automatically process orchard-wide LiDAR data to produce a tree inventory. This includes the necessary adaptations from the segmentation framework proposed by [14] to enable operation in orchards with a continuous Güttingen V trellis structure.

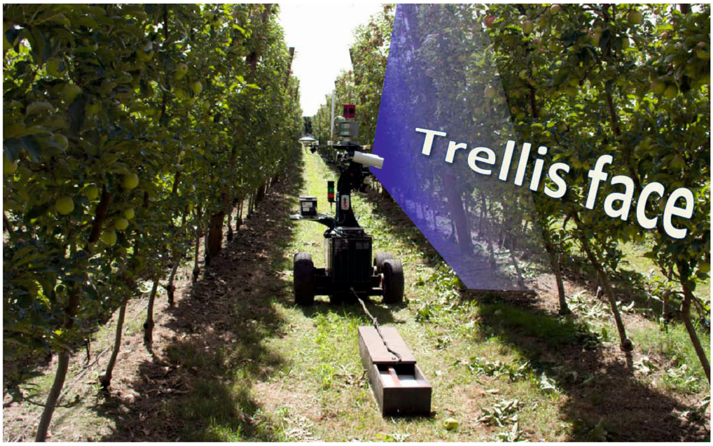


Fig. 1 Apple Orchard, Warburon 3799, Victoria. The research ground vehicle *Shrimp* traversing between two trellis rows, scanning one side using a LiDAR.

The remainder of the paper is organised as follows. Section 2 presents related work on crop/tree localisation. Section 3 gives an in-depth description of the pipeline built for apple trunk localisation. Section 4 evaluates the success of the pipeline on the orchard, discussing its capabilities and shortcomings. We conclude in Section 5, discussing the future directions of this work.

2 Related Work

Automation and advanced sensing in orchards helps farmers make improved decisions regarding farm management. Research in this field has ranged from orchard mapping, autonomous driving for farm vehicles and segmentation and classification of the farm.

Work done by [7] and [10] looks at autonomous vehicle guidance in orchards. Using a combination of visual and laser sensors to detect trees and drivable terrain, along with GPS for localisation, they build a map of the environment. Their developments led to autonomous spraying operations across the farm, however, they do not target individual trees.

Ultrasonic, image and laser sensors have been used by [13, 11] for predicting canopy volume and height for individual trees. From this data, tree health and yield are inferred and associated with trees that have been manually labelled. We wish to automate the construction of a tree inventory to support a database of health, yield and other tree specific information.

Geometry and feature based model fitting have been popular choices in outdoor scenes for locating individual trees. Using data from lasers on a ground vehicle, [8] fits Gaussian Mixture Models (GMMs) onto a row of trees. A single Gaussian cluster represents a tree (assuming they are well separated in the point cloud data) and the ideal number of clusters (representing the count and position of individual trees) is evaluated through an information criterion algorithm. With regards to feature based approaches, tree detection can be performed by classifying points using shape descriptors and shape functions [6, 9]. Airborne laser scanning has also been used for tree detection. In [15] trees are segmented based on height variations captured by a downward facing laser sensor.

The configuration of the apple orchard prevents the use of these techniques for the following reasons: the protective net ceiling would not allow for reliable airborne sensing and the heavily intertwined and overlapping trees blur the definition of an individual tree, preventing the use of canopy geometry or feature based models for segmentation. Additionally, these methods do not leverage from the regular tree spacing, which has been shown by [14] to improve the accuracy of tree segmentation.

In [14], a tree segmentation method is presented that uses laser data on a citrus orchard. By splitting the point cloud data into thin slices along an orchard row, slice by slice classification is performed to separate trees, tree boundaries and gaps between trees. A Hidden Semi-Markov Model (HSMM) helps predict the most optimal state sequence giving rise to a quantifiable observation. Adjacent trees are differentiated from each other through changes in the point cloud heights along the slices. This method is able to encapsulate orchard structural information by setting state duration probabilities, which explicitly encode the repetitive tree spacing as a constraint in the solution. Through segmentation, this method provides a means of localising each tree, building a tree inventory over the orchard.

However, as stated before, in a trellis structured orchard with heavily intertwined trees, the concept of tree segmentation is ambiguous. Additionally, an observational model based on tree height changes is not representative of individual tree locations. Instead, as the trees in orchards are generally pruned at the bottom, the trunks are clearly visible and act as distinct markers for the individual trees. This paper presents a novel observational model that helps identify each tree by performing trunk localisation. In particular, a perception pipeline which is geared towards extracting this model is detailed. An HSMM framework, like the one used in [14], can then be applied for building a tree inventory for meshed tree canopy, trellis orchards.

3 Trunk Localisation Pipeline

This section describes the main stages of the trunk localisation pipeline used for tree identification in trellis environments, as illustrated in Fig. 2.

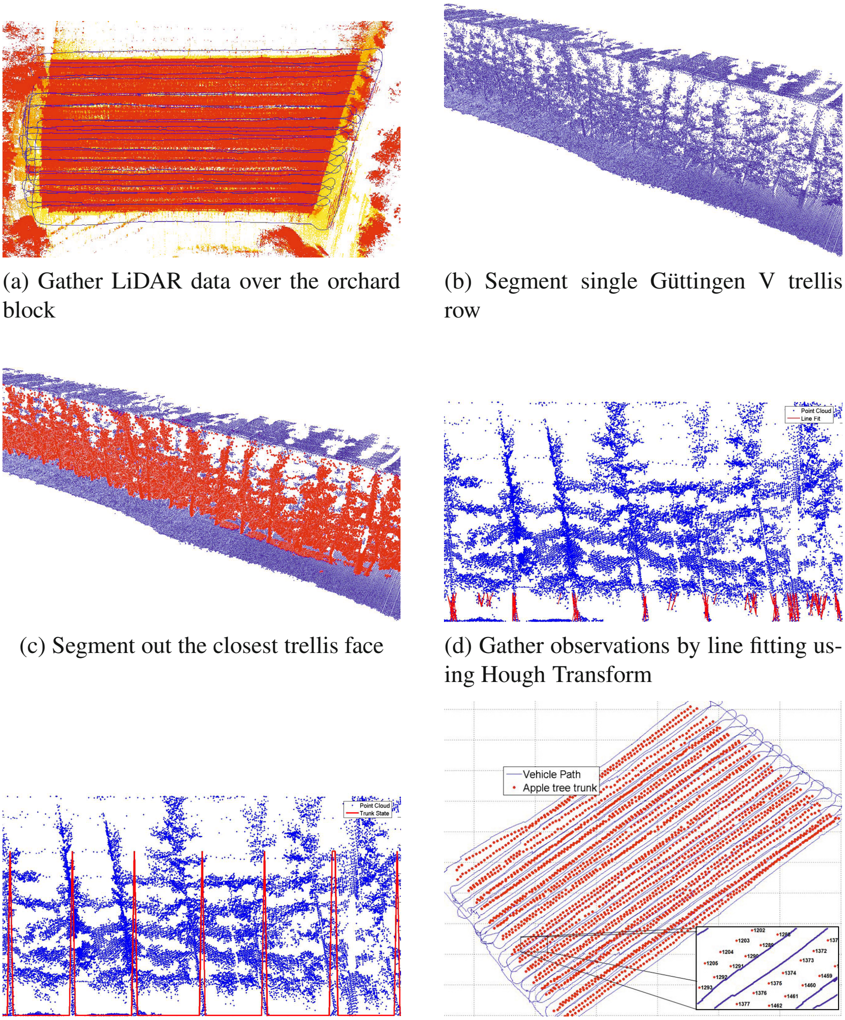
We operate on a half hectare block at the apple farm, which consists of 15 adjacent rows of trees. The primary input into the pipeline is a point cloud representing this block, which is a set of raw measurements that represent the geometry of the orchard (top view in Fig. 2a). This is obtained by geo-referencing a vertically scanning LiDAR mounted on a ground vehicle. Within the sensor frame, a distance threshold is applied to ensure that data from adjacent rows is discarded. Individual rows are segmented using GPS boundaries derived from the structure of the whole block.

A point cloud representation of a row (Güttingen V trellis) is shown in Fig. 2b. It consists of the two halves (trellis faces) of the V structure plantation. The rear trellis face along a row has lower point cloud density due to occlusions. Therefore, trees are segmented on one trellis face at a time. Support poles are fixed along these faces to hold up the trees. By running a parallel localisation process, the poles and trees are detected independently, using two instances of the same algorithm.

We capture observations that relate to states representing trunks and the gaps between the trunks. Splitting this into vertical slices along the row, the HSMM provides a probabilistic framework for estimating the most optimum state sequence resulting in the set observations. By running inference on this model we can estimate individual trunk locations along the row and subsequently build a tree inventory across the farm.

3.1 Trellis Segmentation

Fig. 3 shows the view of a Güttingen V structure from between the two trellis faces. The trees are planted periodically down the row and the trunks of the two halves are separated by roughly 0.5 *m* at the ground. We aim to segment out the two halves of the point cloud shown in Fig. 3b. This is made easier by first removing points belonging to the ground. Ground removal also helps the trunks appear as more distinct linear structures in the point cloud.



(a) Gather LiDAR data over the orchard block

(b) Segment single Güttingen V trellis row

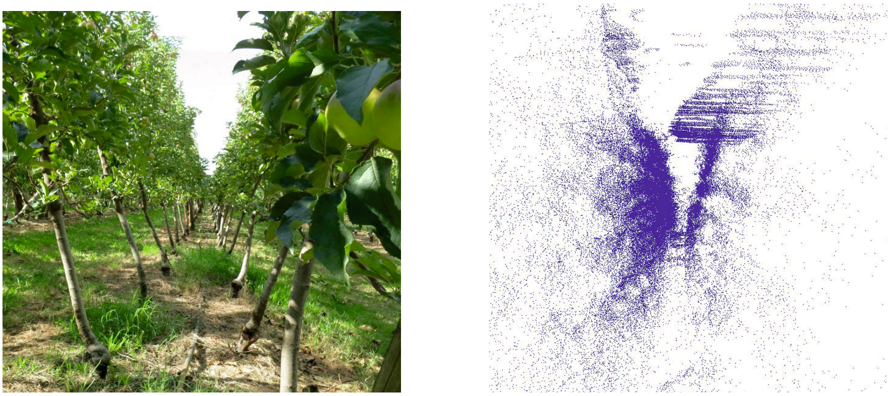
(c) Segment out the closest trellis face

(d) Gather observations by line fitting using Hough Transform

(e) Run inference on HSMM for trunk detection

(f) Tree inventory over the orchard block

Fig. 2 Segmentation pipeline a) → f).
a) Point cloud representation of the orchard block. Points are coloured by elevation and the blue path represents the vehicle trajectory.
b) A single segmented Güttingen V row.
c) Red points represent the segmented trellis face.
d) Hough transform applied to the segmented face to detect lines as shown in red, which equate to possible trunk locations.
e) Trunk and gap estimates obtained by running inference on the HSMM, indicated by the red line with 1:trunk and 0:gap.
f) Tree inventory over the orchard block. Vehicle path in blue, trunks in red. Inset shows individually labelled trunks.



(a) Photo

(b) Laser data

Fig. 3 The Güttingen V trellis structure at the apple orchard. In (a), we are looking through the centre of one Güttingen V trellis. In (b) there is a side view of the equivalent point cloud. The scan was acquired from the left hand side. The density on the right is lower due to occlusion.

3.1.1 Ground Removal

Ground removal in laser data has been a subject of investigation in many mapping and segmentation problems [4]. However, given the constraints on the geometry of the orchard, and the vehicle and sensor configuration, a simple local height threshold was found to be sufficient. To simultaneously account for non-uniform terrain and GPS altitude errors due to poor satellite visibility, we artificially set the altitude coordinate of our localisation estimate to zero everywhere, prior to geo-referencing the laser data. In the resulting data, ground points are at a constant height, which can be cropped with a threshold. A similar approach is used to remove the top cover.

3.1.2 Face Segmentation

To segmented the two faces shown in Fig. 3, we form a piecewise linear boundary along the row length by observing changes in point density into the trellis. A piecewise linear approach is robust to point cloud errors caused by geo-referencing with erroneous localisation due to an occluded GPS signal.

Fig. 4 shows a top view of the point cloud representing a row. To find the separation boundary, the point distribution is analysed along the Y-axis for a sliding window along the X-axis (one instance of the window is the black rectangle in the figure). The sliding window is configured with 2m width and no overlap. A too narrow window would have suffered from noise in the point cloud data and a window too large would fail to capture any sharp transverse changes along the row. Fig. 5 shows the point count histogram for the illustrated window. We can see two peaks representing the front and rear face (smaller second peak due to occlusion).

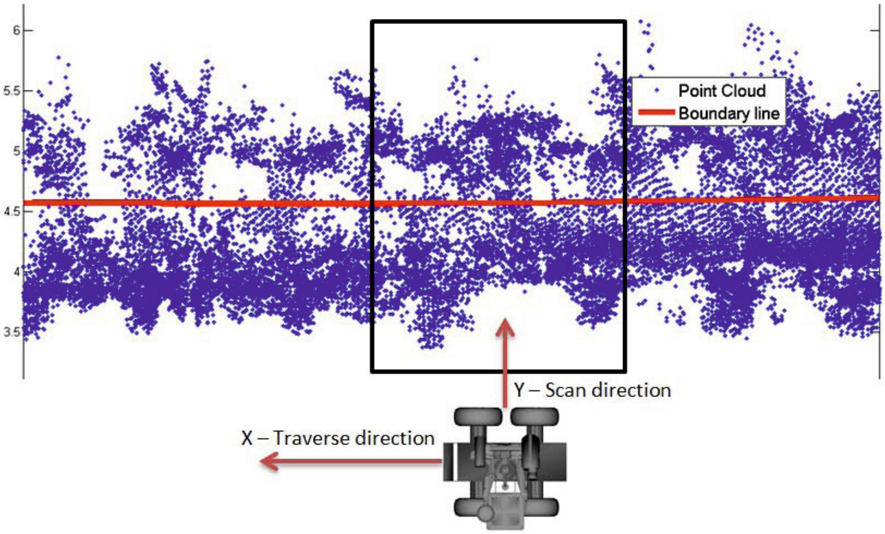


Fig. 4 Top view of the point cloud. The data was obtained from the perspective shown in the image resulting in a higher point density on the closer trellis face. Discrete boundary points are found at sliding windows along the x-direction (black rectangle). The resultant separation boundary is illustrated by the red line.

A Gaussian Mixture Model (GMM) with two modes is a fit to this, and the means represent the central position of the two faces. The boundary point between the faces is taken as the midpoint between the means (shown by the red dot). Additionally, the GMM parameters from the preceding window are used as priors for the next window, which allows to filter out peaks from excessive foliage or inaccurate ground removal. Joining together the boundary points along the row results in the red boundary line illustrated in Fig. 4. The points to one side represent the trellis face closest to the vehicle, which are shown in red in Fig. 2c.

3.2 Hidden Semi-Markov Model

The second half of this pipeline focuses on individual tree trunk detection. For this we make use of a Hidden Semi-Markov Model (HSMM) [2]. The same process is also used in parallel for the detection of support poles, which will be described in Section 3.2.2. HSMMs are used to estimate the most likely sequence of states given a sequence of sensory observations. They have been used diversely, for example, in speech recognition, building rainfall models and modelling traffic on web servers. For trunk detection the states represent trunks and gaps between the trunks, and the observations are derived from raw LiDAR data.

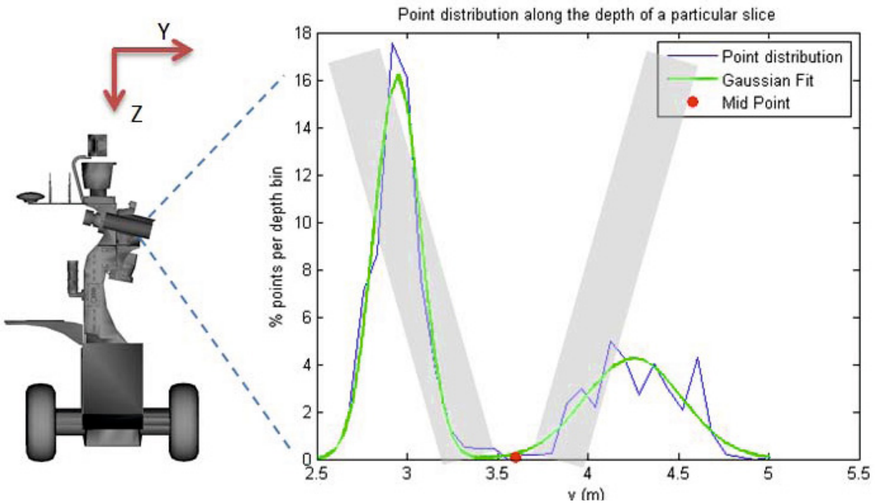


Fig. 5 Distribution of points in a sliced segment of the row. The grey patches represent the two faces of the Güttingen V trellis. A GMM of two modes is fitted onto this data (green lines) and the mid point between their means (red point) is the boundary point between the faces for the given slice.

3.2.1 Observations

The point cloud is not directly representative of trunks and gaps, hence we must define a feature space that can emphasise their unique linear, cylindrical structure.

There are several approaches in the literature that are used to estimate the linearity or cylindrical nature of regions of point clouds. Derived from the work by [12], Principal Component Analysis (PCA) was used to define the linearity, planarity or scatter of the points. In our initial experiments as in theirs, GMMs were trained using hand labelled classes such as ground, trunks and foliage. However, due to a sparse point cloud, the point density on the trunks was not high enough to obtain clear linear structures. This caused the trunks to be often labelled as foliage. Additionally, our aim was to have an unsupervised segmentation pipeline and we wished to avoid the manual labelling.

Another approach that relies on 3*D* point statistics worked through ellipsoidal region growth by using minimum spanning trees [9]. This works by connecting neighbouring points according to an edge weight equivalent to the similarity in the PCA feature space. However, once again due to a sparse representation of the trunks, foliage was often mis-labelled as trunks.

We instead chose a direct line fitting approach using the Hough Transform [5] on a region of interest at the lowest 0.75 *m* of data [1].

[1] RANSAC line fitting was tested, but found to be too computationally expensive due to large ratio of outliers in our experimental data

Given a parametric equation of a line, a voting procedure is used in the parameter space to find lines of best fit on the input data. For points in $3D$ space (as in the case here), a $4D$ Hough space is needed for line detection, which is computationally expensive [1]. An alternative is to detect edges using intersections between fitted planes in $3D$ space [1]. This works well in indoor environments where objects have flat surfaces and sharp edges but is infeasible in an outdoor scene where planar surfaces are infrequent. Instead, we leverage from the inherent $2D$ structure of the trellis and flatten the $3D$ data onto a $2D$ plane, and apply a $2D$ Hough transformation. As the trunks are approximately vertical structures, we restrict the fitted lines to the domain $-15^\circ \leq \theta \leq 15^\circ$. Fig. 2d shows the fitted line on the flattened point cloud.

For use with an HSMM, we divide the point cloud into 5 *cm* slices (roughly the trunk width) along the length of the row, with the aim to infer the state of each slice. The length of the fitted line gives a probabilistic indication of the presence of a trunk. As seen in Fig. 2d, there are several instances of false line fits due to overhanging foliage or tall grass. However by adding structural constraints in the form of regular spacing in the farm, the HSMM can help significantly reduce mis-classification.

3.2.2 Model Parameters

The states in the HSMM are defined as $S = \{S_{trunk}, S_{gap}, S_{row\text{-}end}\}$. With perfect inference, a trunk slice state indicates the presence of trunk points and a group of gap states would be represent the area between two trunks. For a given row, the size of the area before the first tree and after the last depends on how the rows were initially segmented. A row-end state is therefore introduced to model this arbitrary length. A state at slice n is then denoted as q_n, where $n \in [1:N]$ for a total of N slices.

For a sequence of observations $\mathbf{O} = \{O_1, \ldots, O_N\}$, the HSMM aims to find the state sequence $\mathbf{Q} = \{q_1, \ldots, q_N\}$ that best represents the data:.

$$\arg\max_{Q} P(O|Q, \pi, A, B, C) \tag{1}$$

Here π is the initial state distribution, the probability of a state in the first slice. This can either be the row-end state or a tree trunk.

$$\pi = P(q_1 = S_i) = \begin{bmatrix} 0.5 & 0 & 0.5 \end{bmatrix} \;\; i \in \{trunk, gap, row\text{-}end\} \tag{2}$$

The state transition matrix, A represents the probability of moving between states as we go from slice to slice. Its elements are:

$$a_{ij} = P[q_{n+1} = S_j | q_n = S_i], \;\; i, j \in \{trunk, gap, row\text{-}end\} \tag{3}$$

The probabilities are illustrated in Fig. 6. The most likely transitions are between the trunk and gap states. Naturally, it is only possible to transition to/from the row-end state at the end of the rows, which can be modelled with transition probabilities that are a function of slice position. However, we found that setting a fixed low transition probability into the row-end state sufficed for the segmentation process.

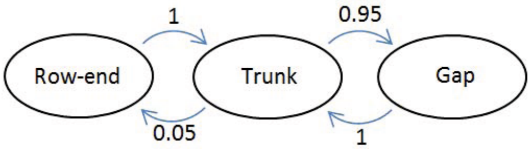
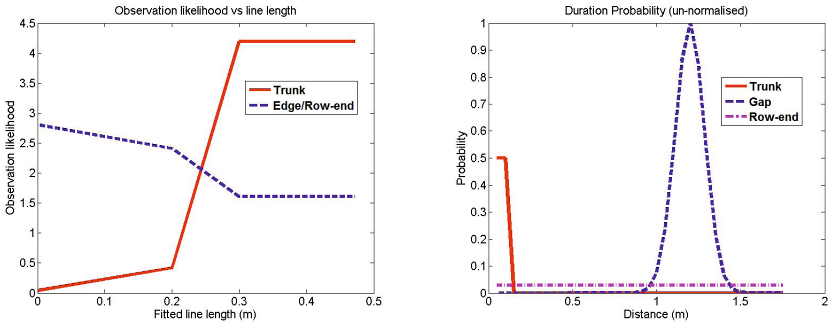


Fig. 6 Transition probabilities between states in the HSMM

The emission probability, B states how likely an observation will occur for a given state

$$B_n = P(O_n|q_n = S_i) \;\; i \in \{trunk, gap, row\text{-}end\} \tag{4}$$

This is illustrated in Fig. 7a. The distribution was manually obtained by analysing the LiDAR data. We state that the longer the fitted line (i.e. O_n), the more likely it is to represent a trunk (red line in Fig. 7a). The inverse was true for the gap/row-end state (blue line), however, we dampened the likelihood as line fits on foliage and tall grass could represent gaps.



(a) State observation likelihood

(b) State duration probability

Fig. 7 Observation and duration probabilities for the HSMM

Finally, the duration probability, C, encapsulates the structural order of the orchard. With slices of 5 cm width, we expect a gap state to last as long as the mean separation between trunks in the farm. Within the HSMM framework[2] we can define a state duration probability:

$$C_i(d) = P(S_i \text{ lasts for d observations}) \tag{5}$$

[2] Self state transitions experience an exponential decay when working with a Hidden Markov Model (HMM). This ability to explicitly set state durations is what differentiates an HSMM from an HMM.

Distributions for the different states are shown in Fig. 7b. The gap duration has been modelled as a Gaussian distribution through manual measurements of trunk separations at the orchard. The trunk states are set to last either one or two slices and the row-end states can last for an arbitrary amount of time.

We run the Viterbi inference algorithm [2] that searches for an optimal solution for Eq 1. The algorithm produces a state estimate at each slice (as shown in Fig. 2e). Tracing back the points present in the trunk slices, a point wise representation of individual trunks can be made (shown by the red points in Fig. 8). Secondly, through this labelling, we can re-evaluate the average trunk separation over a larger data set and therefore redefine the Gaussian duration distribution (Eq 5) shown in Fig. 7b.

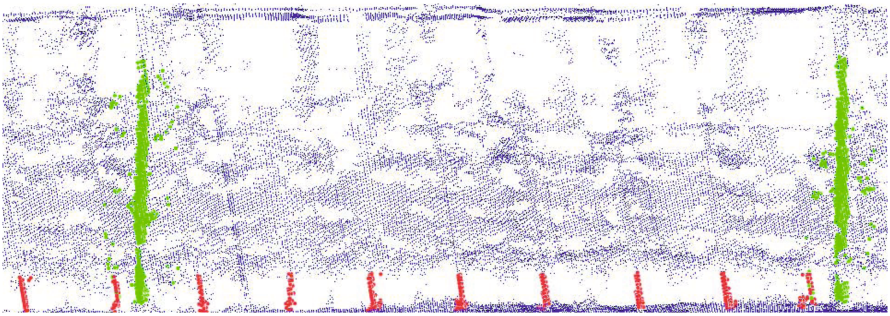


Fig. 8 Resulting Trunk and Pole Localisation

A parallel implementation of the HSMM with different parameters is used to segment support poles (Fig. 8). The states are now $S = \{S_{pole}, S_{gap}, S_{row\text{-}end}\}$ and have a different duration distribution, C. The poles are wider than the trunks and are separated by longer gaps.

4 Trunk Localisation Results

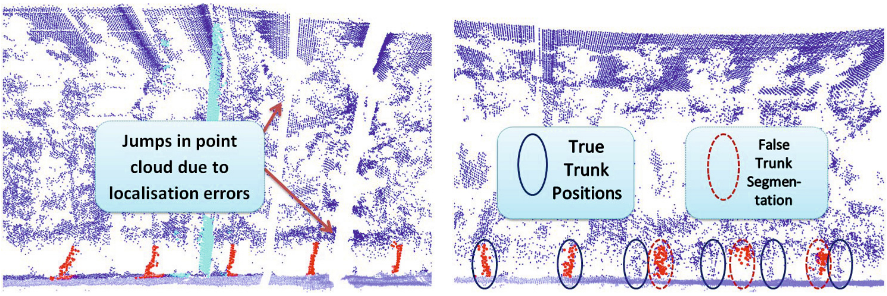
The trunk localisation pipeline was implemented using data obtained from an apple orchard near Melbourne, Australia (Fig. 1). The test vehicle, Shrimp, is a perception research ground vehicle, built at the ACFR, University of Sydney. It is equipped with a vertically oriented $2D$ SICK laser directed perpendicular to the vehicle's direction of travel. The vehicle was teleoperated at $\sim 2\ m/s$ between the rows (vehicle path shown in light blue in Fig. 2a) over a half hectare block on the farm, which is a standard division of the orchard. This consists of 15 rows (30 Güttingen V trellis faces), a total trajectory length of 3157 m and 2629 trees. Using the navigation solution from the on-board Novatel Global Positioning Inertial Navigation System (GPS/INS), a geo-referenced point cloud representation of the entire block was produced. After manually selecting a bounding box around the block, individual rows were automatically segmented by recognising the vehicle's entry/exit points into/out of the box. The manual selection compensates for any GPS errors on a block scale

Table 1 Localisation results over the apple orchard

State	Total count	True Positives	False Positives	False Negatives	Accuracy
Trunk	2629	2350	186	93	89.4%
Pole	294	280	9	5	95.2%

but would need to be repeated when working with a new dataset. We aim to use vehicle heading angle in the future to segment rows, automating this step. Trunk and pole localisation results are summarised in Tab. 1.

True positives represent the correctly identified trunk states, and the localisation accuracy is the ratio of true positives to the total count. A false positive is the classification of foliage and/or tall grass as trunks/poles, and a false negative corresponds to trunks/poles that were missed all together. Instances of these cases can be seen in Fig. 9b. These statistics were evaluated by manual visual interpretation of all individual trunks in the point cloud. To account for any visual bias, we also analysed the respective photos captured by an onboard camera, confirming the presence of a tree or support pole.



(a) Gaps in data due to GPS errors

(b) Trunk mis-classification

Fig. 9 Trunk and Pole segmentation results with a) gaps in the point cloud due to GPS errors. b) large amounts of low hanging foliage.

Localisation errors were due to two primary causes. Firstly, occlusions in the GPS satellite transmission caused jumps in the vehicle position estimates, affecting the geo-referenced point cloud (Fig. 9a). This caused trellis segmentation errors and un-even spacing between trunks. However, as the Viterbi algorithm runs inference on the entire row, small errors are compensated for due to good state estimates on the rest of the row. As a result, we see successful trunk and pole segmentation in Fig. 9a. Larger jumps (not shown on the figure) result in a longer gap state then expected and cause mis-classification, labelling nearby foliage as trunks.

Secondly, errors stemmed from the ambiguity of the line segment observation model. Trunks and support structures are identified through line fits on the point cloud data. However, if the trees have not been trimmed, or if there is tall grass on the ground, distinction between the trunk and gap state becomes harder. Faulty segmentation due to this can be seen in Fig. 9b. The error occurrence was however very minimal and due to the nature of the HSMM model, the error is bounded, with trunk estimates close to the actual trunk.

5 Conclusion

We have described a pipeline that uses a probabilistic approach to localise tree trunks at an apple orchard configured in a Güttingen V trellis structure. The trunk localisation, performed on LiDAR data captured by a UGV includes row extraction, feature selection for representing trees with overlapping canopies and using a Hidden Semi-Markov Model, which leverages from the regular structure of the orchard. Through this, we were able to build a tree inventory over the farm, which can be used for localising yield counts, tree heights/volume and soil quality measurements, providing the farmer with a detailed and accurate model of the orchard.

We found that erroneous GPS data caused by occlusions resulted in erroneous trunk localisation. For future work, we aim to use the regular spacing constraints placed by the HSMM to perform simultaneous localisation and spatial correction, meaning locally smooth odometry could replace GPS entirely. We also intend to expand the tree observation model by combining appearance models from other sensor modalities such as vision and hyperspectral sensing.

Acknowledgements. We would like to thank Gustav Jagbrant for his contribution in developing the HSMM algorithm. We would also like to thank Kevin Sanders for his support during the field trials at his apple orchard. This work was supported by Horticulture Australia as part of the project *AH11009: Autonomous Perception Systems for Horticulture Tree Crops* and the Australian Centre for Field Robotics at the University of Sydney

References

1. Bhattacharya, P., Liu, H., Rosenfeld, A., Thompson, S.: Hough-transform detection of lines in 3-D space. Pattern Recognition Letters 21(9), 843–849 (2000)
2. Bilmes, J.: A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models (1997)
3. Christensen, E.: From orchard to market: Come along on an apple harvest (November 2012), `http://aiweb.techfak.uni-bielefeld.de/content/bworld-robot-control-software` (accessed August 15, 2013)
4. Douillard, B., Underwood, J., Kuntz, N., Vlaskine, V., Quadros, A., Morton, P., Frenkel, A.: On the segmentation of 3D LIDAR point clouds. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 2798–2805. IEEE (2011)
5. Illingworth, J., Kittler, J.: A survey of the hough transform. Computer Vision, Graphics, and Image Processing 44(1), 87–116 (1988)

6. Lalonde, J.-F., Vandapel, N., Huber, D., Hebert, M.: Natural Terrain Classification using Three-Dimensional Ladar Data for Ground Robot Mobility. Journal of Field Robotics 23(10), 839–861 (2006)
7. Moorehead, S.J., Wellington, C.K., Gilmore, B.J., Vallespi, C.: Automating orchards: A system of autonomous tractors for orchard maintenance. In: Proc. IEEE Int. Conf. Intelligent Robots and Systems, Workshop on Agricultural Robotics (2012)
8. Nielsen, M., Slaughter, D.C., Gliever, C., Upadhyaya, S.: Orchard and tree mapping and description using stereo vision and lidar. In: International Conference of Agricultural Engineering (2012)
9. Pauling, F., Bosse, M., Zlot, R.: Automatic segmentation of 3d laser point clouds by ellipsoidal region growing. In: Australasian Conference on Robotics and Automation (2009)
10. Subramanian, V., Burks, T.F., Arroyo, A.A.: Development of machine vision and laser radar based autonomous vehicle guidance systems for citrus grove navigation. Computers and Electronics in Agriculture 53(2), 130–143 (2006)
11. Tumbo, S.D., Salyani, M., Whitney, J.D., Wheaton, T.A., Miller, W.M.: Investigation of laser and ultrasonic ranging sensors for measurements of citrus canopy volume. Applied Engineering in Agriculture 18(3), 367–372 (2002)
12. Vandapel, N., Huber, D.F., Kapuria, A., Hebert, M.: Natural terrain classification using 3-d ladar data. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation, ICRA 2004, pp. 5117–5122. IEEE (2004)
13. Wei, J., Salyarii, M.: Development of a Laser Scanner for Measuring Tree Canopy Characteristics: Phase 2 Foliage Density Measurement. Transactions of the ASABE 48(4), 1595–1601 (2005)
14. Wellington, C., Campoy, J.: Orchard Tree Modeling for Advanced Sprayer Control and Automatic Tree Inventory. In: IEEE/RSJ Int. Conf. on on Intelligent Robots and Systems (IROS) Workshop on Agricultural Robotics (2012)
15. Yu, X., Hyyppä, J., Kaartinen, H., Maltamo, M.: Automatic detection of harvested trees and determination of forest growth using airborne laser scanning. Remote Sensing of Environment 90(4), 451–462 (2004)

LiDAR Based Tree and Platform Localisation in Almond Orchards

Gustav Jagbrant, James Patrick Underwood, Juan Nieto, and Salah Sukkarieh

Abstract. In this paper we present an approach to tree recognition and localisation in orchard environments for tree-crop applications. The method builds on the natural structure of the orchard by first segmenting the data into individual trees using a Hidden Semi-Markov Model. Second, a descriptor for representing the characteristics of the trees is introduced, allowing a Hidden Markov Model based matching method to associate new observations with an existing map of the orchard. The localisation method is evaluated on a dataset collected in an almond orchard, showing good performance and robustness both to segmentation errors and measurement noise.

1 Introduction

Recent years have seen the successful adaptation of autonomous machinery in commercial broad-acre agriculture. In contrast, tree-crops, part of the specialty crops category, are still particularly labour-intensive and suffer from lack of technology for precision sensing and management. The tight and inhomogeneous operating environments present in these fields pose multiple challenges for automation. Nevertheless, robotics and autonomous systems are expected to provoke a revolution in agriculture practices [1].

Information systems, such as the estimation of yield and health, as well as the detection of weeds, pests and diseases, can help to increase agricultural output and to build new practices that ensure long-term sustainability. For most of these operations, the tree is the natural unit of the orchard. A correct recognition of individual trees would thus allow high level objectives to be completed, without introducing any demands for metric accuracy. As such, the detection and recognition of individual trees is a key enabling component for most aspects of orchard automation.

Gustav Jagbrant · James Patrick Underwood · Juan Nieto · Salah Sukkarieh
Australian Centre for Field Robotics (ACFR), University of Sydney, Australia
e-mail: {j.underwood,j.nieto,s.sukkarieh}@acfr.usyd.edu.au

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_32, © Springer International Publishing Switzerland 2015

A key problem for any autonomous robot is localisation; how does the robot know where it is? In agricultural robotics, the most common approach is to use the Global Positioning System (GPS). However, in an orchard environment, the dense and tall vegetation limits the effectiveness and accuracy of GPS, due to occlusion of the satellite signal. In order to enable the use of smaller and more cost efficient robots, it is instead desirable to develop GPS independent systems. The robot will estimate its position using external observations of the environment. However, the repetitive nature of the environment and the lack of distinct salient landmarks can create challenges for general purpose non-GPS based techniques such as Simultaneous Localisation and Mapping (SLAM). Perceptual aliasing is when a region appears visually or geometrically similar to another and is a constant challenge in orchard environments. On the other hand, orchards have a well defined structure, being divided into rows and trees, which can be utilised to aid localisation.

In this paper, we use a 3D data representation of an orchard, created by a 2D LiDAR sensor on a moving ground vehicle, to perform localisation. The approach presented uses the trees as natural orchard landmarks. The trees are first segmented and characterised using distinctive features that permit tree recognition. Using these features for tree recognition, the framework enables topological localisation, or geo-location when the database of trees is geo-referenced.

2 Background

There exist numerous general purpose approaches to segmenting 3D data. For example, the method in [2] can segment multiple arbitrary classes of objects by relying on spatial separation. Empirically, this was found to yield reasonable results in orchard environments, however, frequent contact between adjacent tree canopies caused a percentage of under-segmentation (two trees joined). Using the repetitive orchard structure explicitly as a constraint for segmentation should lead to superior results, hence in this work we focus on such methods.

The authors of [3] present a method utilising Gaussian mixture models with an unknown number of clusters to segment the individual points into trees. In contrast to this per-point segmentation, the authors of [4] propose a method based on splitting the row into vertical perpendicular slices of 0.2 metres width. Based on the height of each slice, the dataset is segmented into trees by employing a Hidden Semi-Markov Model (HSMM).

Our segmentation approach is an extension of the method presented in [4]. In order to increase the segmentation accuracy and the detection performance with regard to irregular small trees, the method is extended to incorporate volume measurements and an additional small-tree state.

To perform localisation based on the segmented trees, it is necessary to be able to compare the similarity of two trees, as observed with sensor data. Determining tree characteristics such as height and volume from 2D LiDAR has been investigated [5] [6], however, to the authors' knowledge there has not been significant research into estimating the similarities of trees measured with 2D LiDAR. Nevertheless, there exists several generic 3D descriptors that could be used for this purpose.

A common trait among many descriptors is to use the normal direction estimated using a sub-sample of points [7] [8] [9]. The significant under-sampling of the tree foliage, due to the sparse nature of the LiDAR data, makes the use of descriptors that require accurate normals inappropriate. Furthermore, as the data is unevenly sampled due to the varying speed of the robot, descriptors aiming to describe the distribution of points [7] [10] [11] are also deemed as inappropriate. In addition, these descriptors are often used to discriminate significantly different classes, whereas we require a descriptor capable of discerning between trees that have been grown to be as similar as possible. Therefore we introduce a new descriptor, specifically designed to differentiate the trees in the orchard.

An integral part of the orchard structure is that the trees are divided into rows. The similarities among trees makes it extremely difficult to perform robust localisation based on one-to-one matching. The ambiguities in data association can be reduced by using sequence matching. In [12], Dynamic Time Warping (DTW) was used to show that sequence-based localisation can be performed despite the information of each element being severely limited. Another option for sequence alignment is to use a Hidden Markov Model (HMM) [13] [14]. HMMs provide greater flexibility in defining the models, making it well suited to explicitly encode the constraints due to the orchard structure and the sensors' noise.

HMMs have been successfully applied in numerous localisation problems. In [13], both sensor measurements and odometry readings are used to perform localisation in a grid map. Another problem was examined in [14], where place based localisation was performed in an indoor environment based only on sensor measurements. This approach is similar to the one employed in this paper, as we want to examine what localisation accuracy can be obtained when based only on tree appearance.

Extending beyond the previous literature on tree segmentation and counting, the primary contribution of this paper is to provide a tree-centric localisation methodology, which not only separates trees as per [4], but introduces *recognition*, allowing us to determine exactly which tree is being observed, by comparison to previously acquired tree databases.

The remainder of the paper is organised as follows: Section 3 describes the outline of the algorithm as well as the individual steps and Section 4 describes the experimental setup and evaluation. Section 5 concludes the paper and discusses potential extensions to the method.

3 Algorithm

The proposed localisation algorithm consists of three dependent steps, as shown in Fig. 1. The input to the algorithm is a 3D point cloud describing an orchard environment, an excerpt of which is presented in Fig. 2. The data are first segmented into individual trees, and then the geometric characteristics of each tree are described.

New tree observations are then matched against a reference map of the orchard, by using a similarity metric. In order to create a reference map, a segmented and characterised point cloud is associated with a secondary reference source. If global localisation is required, the secondary source could be from a vehicle with very accurate GPS-based navigation system, or from geo-referenced satellite or aerial imagery. If topological localisation in a local frame is sufficient, the reference source could simply be the row and tree numbers.

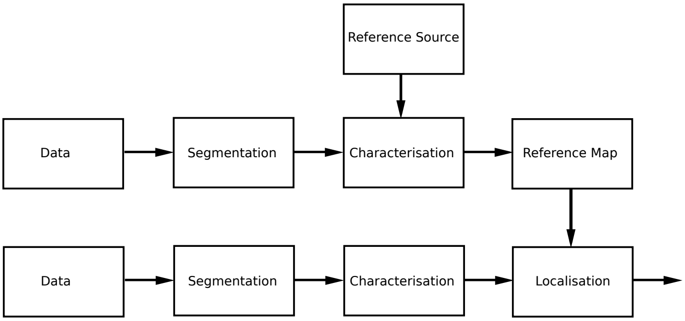


Fig. 1 A block representation of the localisation algorithm

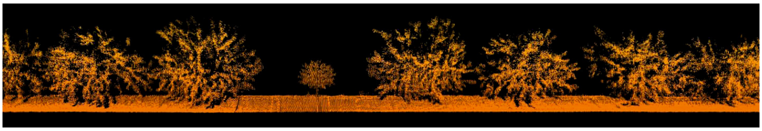


Fig. 2 An excerpt of a point cloud representation of an orchard

3.1 Tree Segmentation

The first step is to segment the LiDAR data for each row in the orchard. This is done using the heading of the robot. Afterwards, each row is quantised into vertical perpendicular slices where the volume and height of each slice are calculated. We used slices of 0.2 m width in our experiments. The volume is calculated by dividing the slice into textbfcubic voxels and counting the number of voxels containing at least one point. The height of a slice is calculated simply as the maximum height of all points in the slice.

Using the height and volume information, a Hidden Semi-Markov Model (HSMM) is used to infer whether the observations of the slice belongs to either: a *tree*, a *gap* (a slice without a tree), the *boundary* of a tree, or a *small tree*. Our model differs from [4] in two respects. First, the volume is used to model the state because the boundaries between trees with meshed canopies were found to be easier to distinguish by this feature. Second, the *small tree* state is introduced in order to improve the performance with regard to irregular small trees (usually new plants).

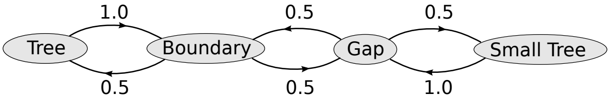


Fig. 3 The state transition diagram of the orchard model. Note that the explicit state duration distributions are not shown.

In order to incorporate prior knowledge of the tree width, the duration of the state *tree* is defined as a Gaussian centred around the expected mean width. The *small tree* state is given a short uniform distribution, making thin trees more probable. However, as this may cause repeated transitions between *gap* and *small tree*, the *gap* state is given a notable minimum duration. Furthermore, different to [4], the *gap* distribution is modelled as uniform, as the width of gaps were found to vary uniformly. Finally, the duration of the *boundary* state is explicitly set to 1 in order to guarantee that it is as precise as possible. Fig. 3 illustrates the state transition model.

The height measurements are used to determine the likelihood of the *gap* and *small tree* states, while the volume measurements determine the likelihood of the *tree* and *boundary* states. For the *boundary* state however, the volume feature is not used directly. Instead the difference between a broad moving average and a small moving average of the volume is calculated, giving an estimate of the size of the local volume compared to the immediate surrounding. An example of this feature is presented in Fig. 4. Similar to the work done in [4], all observation likelihoods were hand-tuned.

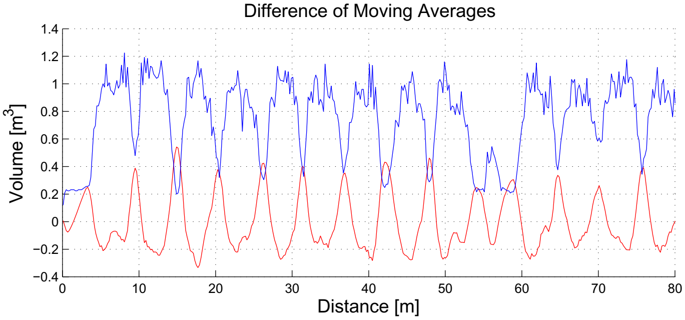


Fig. 4 The measured volume (—) and the difference of moving averages (—)

Using the model presented above, the Viterbi algorithm [15] is used to find the optimal state sequence given the measurements. An example of the result is shown in Fig. 5.

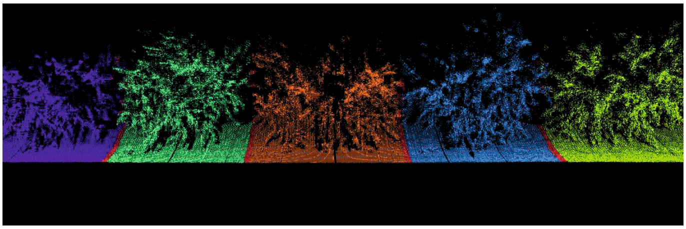


Fig. 5 An typical segmentation result, with unique colours per segment and red boundaries

3.2 *Tree Characterisation*

The data collected in the orchard is both sparse and unevenly sampled compared to the fine geometry of tree canopies, suggesting that the generic 3D descriptors relying on normal estimation or point distributions will yield inaccurate results. Instead, an application specific *height signature descriptor* is introduced, which contains the height measurements of each slice in the segmented tree. The possibility of using an equivalent volume descriptor was also examined, however it was shown to be less consistent. An example height signature descriptor of a single tree is shown in Fig. 6. The tree has been scanned on four separate occasions to illustrate the repeatability of the signature.

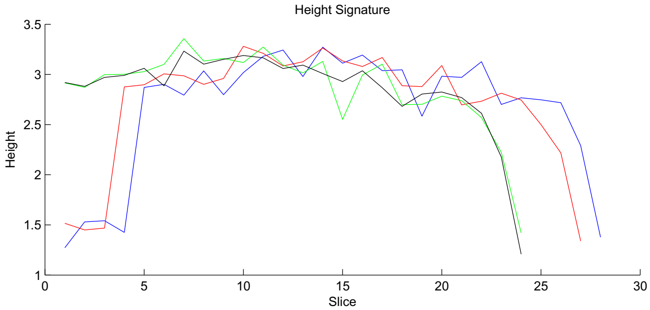


Fig. 6 The height signature of one tree scanned on four separate occasions

The signatures are generated from the slice height measurements of consecutive tree state segments of data, and as such, they are not guaranteed to be of equal length. Therefore the signature difference is calculated by finding the best fit between two signatures. Denoting the longer sequence $\mathbf{s}_{long}$, with length N_{long}, and the shorter $\mathbf{s}_{short}$, with length N_{short}, there are $N = N_{long} - N_{short} + 1$ discrete sequence alignments. For each alignment i, the shorter sequence is padded with zeros:

$$\mathbf{s}^0_{short,i} = [\mathbf{0}(i)\,, \mathbf{s}_{short}\,, \mathbf{0}(N-i-1)], \tag{1}$$

where $\mathbf{0}(i)$ denotes a vector of i zeros. The minimum distance Δ_s is calculated using the L1 norm:

$$\Delta_s(\mathbf{s}_{long}, \mathbf{s}_{short}) = \min_{i \in [0, N-1]} |\mathbf{s}_{long} - \mathbf{s}^0_{short,i}| \qquad (2)$$

3.3 Localisation

Assuming there exists a prior map where each tree is represented by a descriptor, the task of the localisation algorithm is to match newly observed trees with the correct tree in the map. This is useful for online localisation for autonomy and offline (batch processing) to associate new observations with previously mapped trees, allowing the health, growth and yield of individual trees to be monitored and stored over time.

In operation, the robot is constrained to move either forwards or backwards along a row, with no possibility of changing rows before the end is reached. This constraint is integrated into the localisation algorithm. In addition, if the direction of the robot is known, for example from odometry, the algorithm is further constrained. Therefore, two localisation methods are presented, one assumes knowledge of the robot's direction of motion, while the other does not.

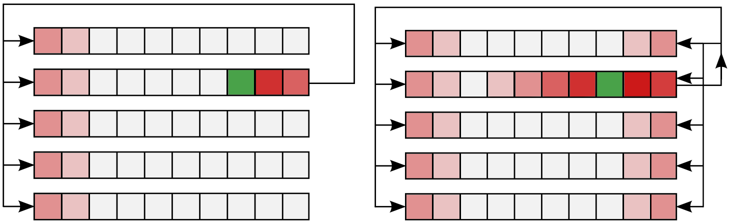
The prior map is represented by a Hidden Markov Model (HMM) [16] where each state represents an individual tree, seen from a specific side. A height signature descriptor is stored for each state and the transition matrix is designed to integrate the motion constraints of the robot. Localisation is performed by computing the descriptors of a newly observed sequence of trees, comparing the descriptors to those in the map, and finding the most probable corresponding states in the HMM. In the online situation, the state sequence is found using the Forward algorithm [16], while the Forward-Backward algorithm [16] is used in offline localisation.

The robot's motion is modelled by the transition matrix A, where entries A_{ij} contain the probability of the robot moving from tree i to tree j. In order to allow for segmentation errors, the intra-row transition elements are defined by Eq. 3, allowing for both self-transitions as well as transitions to non-adjacent trees. If the direction of the robot is not known, the elements are defined according to Eq. 4, which permits transition to trees either in front or behind the robot.

$$A_{ij} = \begin{cases} 0.10 & \text{, if } j = i+0 \\ 0.70 & \text{, if } j = i+1 \\ 0.10 & \text{, if } j = i+2 \\ 0.05 & \text{, if } j = i+3 \\ 0.05 & \text{, if } j = i+4 \\ 0 & \text{, else} \end{cases}, \qquad (3)$$

$$A_{ij} = \begin{cases} 0.025 & \text{, if } j = i-4 \\ 0.025 & \text{, if } j = i-3 \\ 0.05 & \text{, if } j = i-2 \\ 0.35 & \text{, if } j = i-1 \\ 0.10 & \text{, if } j = i+0 \\ 0.35 & \text{, if } j = i+1 \\ 0.05 & \text{, if } j = i+2 \\ 0.025 & \text{, if } j = i+3 \\ 0.025 & \text{, if } j = i+4 \\ 0 & \text{, else} \end{cases}. \qquad (4)$$

Transitions between rows are handled by assuming it is possible to move from one row to any other, including a return into the same row. When Equations (3) or (4) assign probability mass past the end of the current row, the overhanging mass is evenly spread among the corresponding entry points of all rows. This is illustrated by an example where the robot is known to be three trees from the end of a row, with a constraint of motion to the right in Fig. 7(a) and in either direction in Fig. 7(b).



(a) Directed Row Transition (b) Undirected Row Transition

Fig. 7 An example of (a) rightward directed and (b) undirected transitions (🟥) given the current position (🟩). The red shading indicates probability.

To determine the likelihood of a state given an observed tree, the difference between the observed and previously mapped descriptor is compared to the self-similarity distribution, shown for real data in Fig. 11. The match likelihood is defined as the cumulative probability of observing a difference larger than the measured difference given an identical tree, shown in Fig. 8 and described further in the next Section.

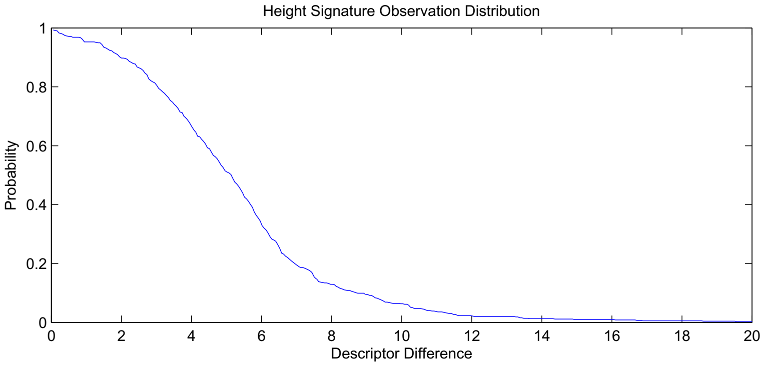
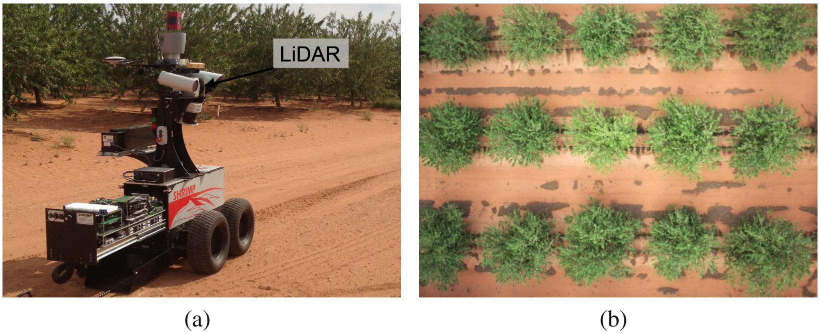


Fig. 8 The likelihood of a matching tree, given a height signature difference observation, as described in Section 4.1

4 Algorithm Evaluation

Data were collected at an almond orchard in Mildura (Victoria, Australia), by the robot shown in Fig. 9(a). The vertically oriented SICK LMS-291 2D LiDAR points to the right of the vehicle, generating range data at 75Hz, which is swept by the forward motion of the vehicle. A global positioning inertial navigation system (GPS/INS) estimates the robot's trajectory, however, due to the vegetation in the orchard, the estimates were inaccurate. Nevertheless, this was used to geo-reference the LiDAR data to create a point cloud, because the INS guarantees local smoothness at the scale of individual trees. Future work will use vehicle odometry to create a smooth local trajectory, with global corrections provided by the tree matching system proposed by this paper. An aerial view of a small part of the orchard is given in Fig. 9(b), showing the orchard row structure. Additionally, an overview of the surveyed orchard block is presented in Fig. 10. Each letter corresponds to scanning one side of a line of trees, as this quantum of data is treated separately in this paper. The pairs (B,C), (D,E), (F,G), and (H,I), represent the same line of trees seen from two different sides. The data are divided into four datasets, with $d1$ containing all rows A-J, and $d2, d3, d4$ being three repeated scans of rows B and C.



(a)

(b)

Fig. 9 Photos from the data collection process showing a) the perception research ground vehicle "Shrimp" and b) an aerial view of the orchard structure

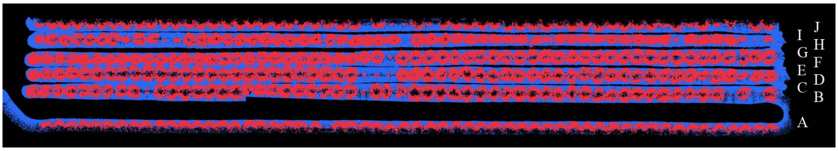


Fig. 10 An overview of the surveyed orchard block where the different rows' sides have been marked with different letters

4.1 Self-similarity Distribution

Using datasets 1 to 4, the height signature difference is calculated for the same trees seen multiple times in rows B and C, and also for different trees in all rows, to estimate the response of the measure for identical and differing trees. The distributions are shown in Fig. 11, which indicates that on average, the descriptor is an effective way to differentiate trees. Note that the cumulative PDF shown previously in Fig. 8 was obtained from these self-similarity distributions.

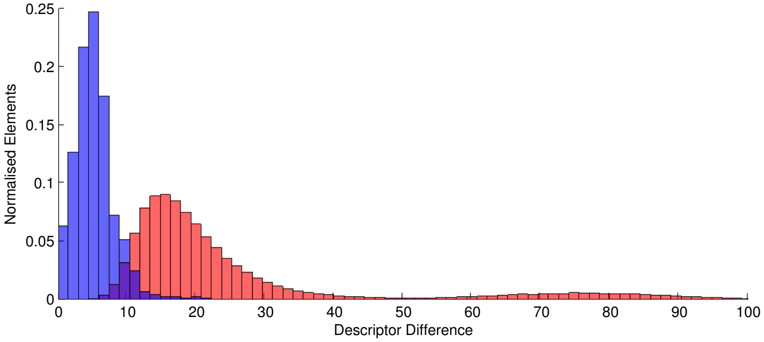


Fig. 11 The self-similarity distribution of the height signature descriptor (🟦) and the similarity between different trees (🟥)

4.2 Segmentation

The performance of the segmentation was evaluated by visually inspecting the segmentation on dataset 1 and counting the resulting:

- True Positives: trees labelled as trees.
- False Positives: gaps labelled as trees
- False Negatives: trees labelled as gaps
- Boundary Errors: a tree/gap boundary that from visual inspection could be placed more precisely.

True negatives were not counted, as a single gap between two trees is represented by arbitrarily many consecutive *gap* states, therefore true negatives are poorly defined. The results of the evaluation are presented in Table 1, showing that the method accurately detects and segments the trees.

Table 1 The performance of the segmentation method applied on dataset 1

True Positives	False Positives	False Negatives	Boundary Errors
579	0	0	2

4.3 Characterisation

To determine the applicability of the height signature descriptor for localisation, a preliminary sequence matching test was done using exhaustive search. Assuming no segmentation errors, this allows to determine the information contained in the height signature features. Sequences of trees from datasets 2 to 4 were localised within a map created from dataset 1. Test sequences of length *N* were compared to all possible sequences of length *N* in the map to find the closest match, by minimising the sum of the individual height descriptor differences. Before the test was performed, the segmentation was visually inspected and it was verified that there were no segmentation errors.

The ratio of correct matches was calculated for the height and volume signature descriptors. For completeness, comparison is made to using only the maximum height, or the total volume of the tree, (analogous to a signature of length 1). The results, presented in Table 2, show that although no descriptor is able to uniquely characterise a single tree, using short sequences provides a considerable performance increase. The power of sequence matching shows that localisation is possible even with single point height or volume descriptor, though signatures enable far shorter sequences. This simple approach presents a good localisation solution, even for the "kidnapped robot problem". The only problem is that the method is sensitive to possible segmentation errors.

Table 2 The correct match ratio (%) for sequences of length *N* from datasets 2-4 vs. dataset 1

Length	Simple Volume	Simple Height	Volume Signature	Height Signature
1	3.46	5.48	65.71	74.64
3	34.03	44.48	92.54	98.21
5	63.16	69.97	97.83	99.69
10	85.32	88.40	100.0	100.0
20	99.57	97.85	100.0	100.0

4.4 Localisation

To evaluate the performance of the complete HMM based localisation method, a map was created from dataset 1. Thereafter a sequence of observations was localised in the map and the ratio of correct matches calculated. Two different types of datasets were localised against the map: *real* datasets 1 to 4, and *synthesised* datasets, created by perturbing the height measurements and boundary positions of dataset 1. Furthermore, segmentation errors were introduced into all sequences. Using this configuration it is possible to determine both the algorithm's performance and its robustness to noise and segmentation errors.

Three different types of segmentation errors were introduced:

- merge errors: two trees are merged into one;
- split errors: a tree is split into two trees;
- detection errors: a tree is incorrectly labelled as a gap.

The correctness of a match with the segmentation errors introduced was defined according to the following definitions:

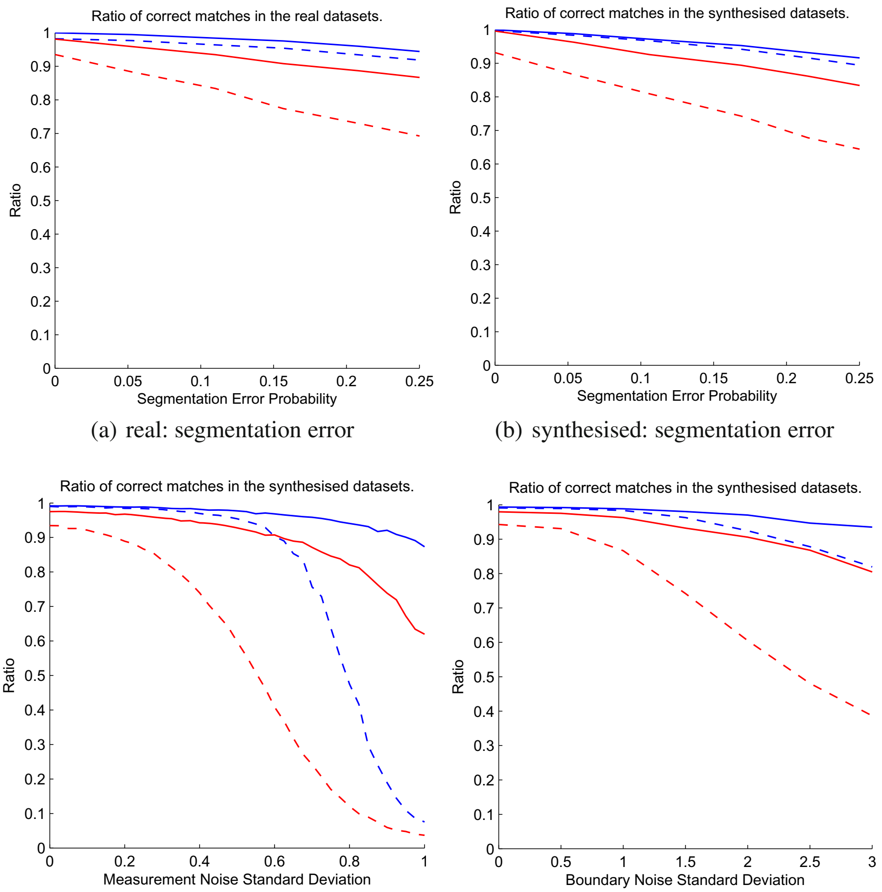
- for a split tree, the correct match for both split components is the original tree;
- for a merged tree, a match is considered to be correct if the merged tree is matched with either of the two original trees;
- trees labelled as gaps are not part of the matching process and yield neither correct nor incorrect matches.

To calculate appropriate measurement noise values, 1000 slices from datasets 1 to 4 were aligned and the slice height variation was found to be Gaussian with mean of approximately 0 and standard deviation $0.25m$. The boundary noise was calculated from the variation in tree width according to $\sigma_{width} = \sqrt{2} \cdot \sigma_{boundary}$, because the labelled tree width is a function of the two boundaries. The boundary noise $\sigma_{boundary}$ was found to be 1.0 slices. For the synthesised tests, one noise or segmentation error parameter was changed while the others remain fixed, as shown in Table 3. Note that the segmentation error probability is the probability of a tree being affected by any of the introduced segmentation errors.

Table 3 Parameters used to evaluate the performance and robustness of the localisation method

Dataset	Boundary StdDev	Measurement StdDev	Segmentation Error Prob.
Synthesised	**0.0-3.0 slices**	0.25 m	0.05
Synthesised	1.0 slices	**0.0-1.0 m**	0.05
Synthesised	1.0 slices	0.25 m	**0.00-0.25**
Real	-	-	**0.00-0.25**

The localisation performance is shown for varying amounts of introduced segmentation error in Figs. 12(a) and 12(b). The results are similar for the real and synthesised datasets, validating the noise modelling process. As expected, the offline localisation (smoothing) performs better than the online version (filtering). Using the direction of the robot is also shown to have a positive impact on the performance, especially when large amounts of segmentation error are introduced. The performance when the measurement and boundary noise is varied are presented in Figures 12(c) and 12(d), showing that all but the online undirected localisation perform well even at noise levels significantly larger than those encountered in the real datasets.



(a) real: segmentation error

(b) synthesised: segmentation error

(c) synthesised: measurement error

(d) synthesised: boundary error

Fig. 12 The ratio of correctly matched trees using offline, directed (—) and undirected (---), and online, directed (—) and undirected (---), localisation. Varying the ratio of segmentation errors introduced to the a) real b) synthesised datasets. Varying the c) measurement noise and d) boundary noise.

5 Conclusion

A three step localisation algorithm that utilises the inherent structure of the orchard has been presented. It has been shown that tree height and volume measurements derived from a 2D LiDAR sensor are sufficiently informative to enable accurate tree segmentation and for sequence-based orchard-wide tree recognition and localisation. Furthermore, we have shown that the proposed algorithm is robust both against segmentation errors and measurement noise.

It remains for future work to evaluate the proposed method when using pure vehicle odometry to arrange the LiDAR slices and calculate local height signatures for each tree. Nevertheless, given the low satellite visibility of the GPS during the experiments and the subsequent reliance on smooth local trajectories for LiDAR compilation, we are confident that similar performance can be obtained using odometry. In addition, it is also necessary to evaluate the effects of seasonal variations in the trees' appearance, and to determine appropriate mechanisms for updating the database to manage the tree signature changes over time.

Acknowledgements. This work is supported by the Australian Centre for Field Robotics at the University of Sydney and Horticulture Australia Limited through project AH11009 Autonomous Perception Systems for Horticulture Tree Crops.

References

1. Bergerman, M., van Henten, E., Billingsley, J., Reid, J., Mingcong, D.: IEEE Robotics and Automation Society Technical Committee on Agricultural Robotics and Automation. IEEE Robotics & Automation Magazine (2013)
2. Douillard, B., Underwood, J., Kuntz, N., Vlaskine, V., Quadros, A., Morton, P., Frenkel, A.: On the segmentation of 3D LIDAR point clouds. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2798–2805 (2011)
3. Nielsen, M., Slaughter, D.C., Gliever, C., Upadhyaya, S.: Orchard and tree mapping and description using stereo vision and lidar. In: SPC-03: IV International Workshop on Computer Image Analysis in Agriculture (2012)
4. Wellington, C., Campoy, J., Khot, L., Ehsani, R.: Orchard tree modeling for advanced sprayer control and automatic tree inventory. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) Workshop on Agricultural Robotics (2012)
5. Wei, J., Salyani, M.: Development of a laser scanner for measuring tree canopy characteristics Phase 2. Foliage density measurement. Transactions of the ASAE 48(4), 1595–1601 (2005)
6. Rosell, P., Ramon, J., Sanz, R., Llorens, J., Arn, J., Ribes-Dasi, M., Masip, J., Camp, F., et al.: A tractor-mounted scanning LIDAR for the non-destructive measurement of vegetative volume and surface area of tree-row plantations: A comparison with conventional destructive measurements. Biosystems Engineering 102(2), 128–134 (2009)
7. Johnson, A.E.: Spin-images: a representation for 3-D surface matching. PhD diss., Microsoft Research (1997)
8. Frome, A., Huber, D., Kolluri, R., Bülow, T., Malik, J.: Recognizing objects in range data using regional point descriptors. In: Pajdla, T., Matas, J. (eds.) ECCV 2004. LNCS, vol. 3023, pp. 224–237. Springer, Heidelberg (2004)
9. Rusu, R.B., Blodow, N., Marton, Z.C., Beetz, M.: Aligning point cloud views using persistent feature histograms. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3384–3391 (2008)
10. Osada, R., Funkhouser, T., Chazelle, B., Dobkin, D.: Shape distributions. ACM Transactions on Graphics (TOG) 21(4), 807–832 (2002)
11. Wohlkinger, W., Vincze, M.: Ensemble of shape functions for 3D object classification. In: IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 2987–2992 (2011)

12. Milford, M.: Visual route recognition with a handful of bits. In: Proceedings of Robotics Science and Systems Conference 2012, University of Sydney (2012)
13. Fox, D., Wolfram, B., Thrun, S.: Markov localization for mobile robots in dynamic environments. Journal of Artificial Intelligence Research 11, 391–427 (1999)
14. Rahman, S., Zelinsky, A.: Mobile Robot Navigation based on localisation using Hidden Markov Models. In: Australasian Conference on Robotics and Automation, ACRA (1999)
15. Yu, S.: Hidden semi-Markov models. Artificial Intelligence 174(2), 215–243 (2010)
16. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE 77(2), 257–286 (1989)

A Feature Learning Based Approach for Automated Fruit Yield Estimation

Calvin Hung, James Underwood, Juan Nieto, and Salah Sukkarieh

Abstract. This paper demonstrates a generalised multi-scale feature learning approach to multi-class segmentation, applied to the estimation of fruit yield on tree-crops. The learning approach makes the algorithm flexible and adaptable to different classification problems, and hence applicable to a wide variety of tree-crop applications. Extensive experiments were performed on a dataset consisting of 8000 colour images collected in an apple orchard. This paper shows that the algorithm was able to segment apples with different sizes and colours in an outdoor environment with natural lighting conditions, with a single model obtained from images captured using a monocular colour camera. The segmentation results are applied to the problem of fruit counting and the results are compared against manual counting. The results show a squared correlation coefficient of $R^2 = 0.81$.

1 Introduction

A continuous growing population has generated an unprecedented demand for agriculture production. Several problems with traditional farming will need to be addressed in order to achieve the required increase in production. First of all, there is a shortage in labour in rural areas. Migration and urbanization have left rural areas with a reduced and aging population and has also reduced the available farmland for production. Secondly, the impacts of current practices to climate change provoke a necessity for innovation in order to make agriculture a sustainable practice. Reducing emissions, increasing efficiency in the use of resources such as water, reducing the use of chemical products are some of

Calvin Hung · James Underwood · Juan Nieto · Salah Sukkarieh
Australian Centre for Field Robotics, The University of Sydney
e-mail: {c.hung,j.underwood,j.nieto,salah}@acfr.usyd.edu.au

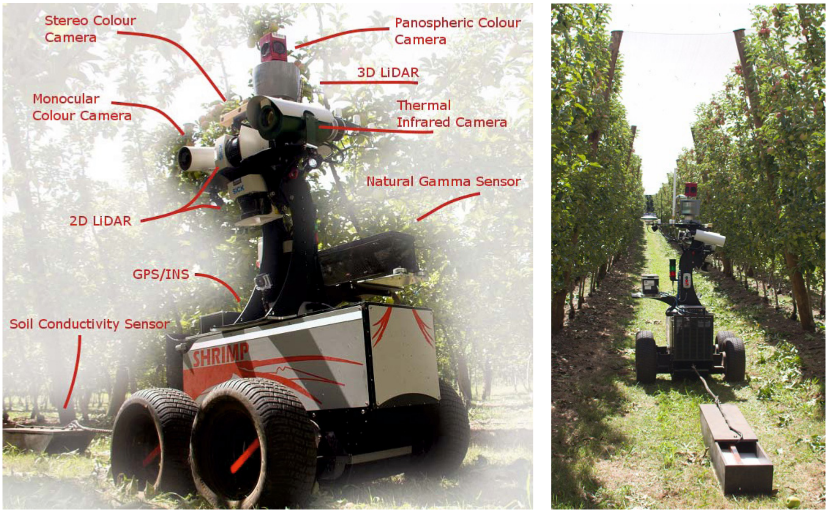
L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_33, © Springer International Publishing Switzerland 2015

the actions that will need to be addressed to reduce the environmental footprint of farming.

Robotics and automation are expected to have a significant impact on farms of the future by increasing efficiency in production and the use of resources and reducing labour costs. Information gathering systems are already being used in many farms. Geographic Information Systems (GIS) based on remote sensing data have been shown to provide good methods to solve spatial problems and support decision making [1]. The main limitation of GIS is that they provide coarse spatial and temporal information. Manual sampling suffers from the same two shortfalls, in addition to an increase in labour cost. On the other hand, field data from ground mobile sensor platforms can increase spatial and temporal resolution with improved operational and cost benefits.

The operational benefit of the autonomous system is to provide accurate yield estimation that grants better decision making such as fertiliser application across a farm, prediction for production purposes later in the supply chain. In addition the research into image processing can provide understanding of individual health and nutritional values of the fruit.

This paper demonstrates a feature learning based approach for image based fruit segmentation. We present results for apple segmentation using a camera mounted on an unmanned ground vehicle shown in Fig.1. Outdoor fruit classification using vision systems is a challenging task. An autonomous segmentation approach will have to deal with illumination changes, occlusion, variety of object sizes and in our case variety of colours. The algorithm presented here performs pixel classification using feature learning to automatically select the most important attributes of the target



(a) Sensor Configuration (b) Scanning a Row

Fig. 1 The perception research ground vehicle *Shrimp*, operating at the apple orchard

(fruit). We show that the approach can handle different illumination conditions and different fruit size and colour.

In particular this paper presents the following contributions:

- an extension for apple tree segmentation of the algorithm presented in [2] for almonds;
- extensive experiments using data collected in an apple orchard with a robotic platform. The algorithm detects both, red and green apples with a common model;
- a pipeline for fruit counting and validation against manual fruit counting performed by the farmer.

2 Related Work

There has been a relatively large amount of work presented in fruit segmentation using vision. The literature covers a variety of fruits such as grapes [3] [4], mangoes [5], oranges [6] and apples [7, 8]. A common trend in orchard tree classification is the use of manually designed features, such as intensity or colour ranges or shape features [9]. For example, the work presented in [8] uses a simple intensity threshold to segment flowers, while the work presented in [5] uses a colour threshold. Another example is the work proposed by [4] where a shape based detection followed by a colour and texture classifier is applied to grape segmentation. The main drawback of these approaches is that they are specifically designed for a particular fruit variety and will need to be re-designed for different fruits or to cope with variations such as seasonal changes.

Apart from the always present occlusion and illumination challenges, fruit classification becomes a simpler task when the fruit has a distinctive colour from the foliage. Green apples however, do not belong to that category. A pipeline for apple segmentation is presented in [7]. In order to control the illumination the authors collected data at night with an artificial source. The algorithm selects red apples using colour and green apples using specular reflection features.

The fruit classification framework presented here has been designed with the aim of automatically adapting the feature sets for different fruits; the feature extraction and classification rules are all obtained via learning. Our approach does not require domain specific assumptions and can therefore be applied to different types of trees.

3 Fruit Classification

Green apple detection using vision is challenging due to the low contrast of the fruit with the background foliage. In addition, outdoor environments present extra difficulties due to the variations in illumination caused by occlusion and different weather conditions.

This paper applies the fruit segmentation algorithm previously demonstrated on almond trees [2], to apple segmentation and counting. The learning based approach allows the algorithm to be applied on fruit of different appearance. This section provides a brief description of the algorithm, interested reader please refer to [2] for details.

3.1 Image Modelling Using Conditional Random Fields

This paper models the image data using a Conditional Random Fields (CRF) framework [10]. The graphical model for an image consists of a two dimensional lattice $\mathscr{G} =< \mathscr{V}, \mathscr{E} >$ where $\mathscr{V}$ is a set of pixels representing the vertices of the graph and $\mathscr{E}$ is a set of edges modeling the relationships between the neighbouring pixels.

Image segmentation is performed by assigning to every pixel $x_i \in \mathscr{V}$ in the image a meaningful label $l_i \in \mathscr{L}$. For multi-class image segmentation, the label set $\mathscr{L}$ may contain multiple labels up to k classes $\mathscr{L} \in \{1,...,k\}$. The optimal labeling $l^\star$ is obtained via energy minimisation on the graph structure $\mathscr{G}$ with the energy function defined as in Eq. 1

$$E(\mathbf{l}) = \sum_{i \in \mathscr{V}} \psi_i(l_i,x_i) + \sum_{(i,j) \in \mathscr{E}} \psi_{ij}(l_i,l_j,x_i,x_j) - log(Z(\mathbf{x})) \tag{1}$$

where $\psi_i(l_i,x_i)$ is the unary potential which models the likelihood of a pixel taking a certain label, $\psi_{ij}(l_i,l_j)$ is the pairwise potential which models the assumption that the neighbouring pixels should take the same label, and $log(Z(\mathbf{x}))$ is the partition function.

Conventionally the unary potential is computed using the features in the image, for example grey level intensity [11], colour [12], or texture [13]. In our approach the unary potential is generated using the multi-scale features learnt from the image dataset.

3.2 Multi-scale Feature Learning

This section provides an overview of our feature learning approach. We first apply a sparse autoencoder [14] at different scales to obtain the multi-scale feature dictionaries. A logistic regression classifier is then used to learn the label association to the multiscale responses. The classifier output is then passed into the CRF as the unary term described in Eq. 1 for multi-class image segmentation.

The first step is to use unsupervised feature learning to capture the features from an unlabelled dataset. In this paper a sparse autoencoder is used to learn the dictionaries from randomly sampled image patches. This process is then repeated with images sub-sampled at different scales.

With the low dimensional dictionary code obtained using unsupervised feature learning, an additional supervised label assignment step can be used to train a classifier. In this paper a softmax regression classifier is used. Softmax regression was chosen because it can be formulated as a single layer perceptron and trained using back-propagation.

Lastly the sparse autoencoder used for unsupervised feature learning and the logistic regression classifier are joined into one single network to fine-tune the network parameters.

3.3 Fruit Counting

With the apple segmentation output, two approaches are used to estimate the fruit count. The first is to count the total number of fruit pixels per side of the row and use the pixel count to infer actual fruit count, the second approach is to perform circle detection using the circular Hough transform [15] to estimate the actual fruit count. Prior to detection, image erosion is applied to remove the noise and dilation is applied to join partially occulded fruit.

4 Experimental Setup

The experiments were conducted using *Shrimp*; a general purpose perception research ground vehicle. The system is equipped with a variety of localisation, ranging and imaging sensors that are commonly used in machine perception, together with soil specific sensors for agricultural applications, see Fig. 1(a).

Data from all sensors were gathered from a single $0.5ha$ block at a commercial apple orchard near Melbourne, Victoria in Australia, shown in Fig. 3. The data collection was performed one week before harvest with the apple diameter ranged from 70 to 76 mm. The orchard employs a modern Güttingen V trellis structure, whereby the crop is arranged in pairs of 2D planar trellises, arranged to form a ’V’ shape. The structure was designed to maximise sunlight on the leaves and to promote efficient manual harvesting, and potentially would be an appropriate structure for future robotic harvesters. The Shrimp robot acquired data while driving between the rows, as shown in Fig. 1(b). All fifteen rows in Fig. 3 were scanned, with a trajectory length of $3.1km$ and a total of $0.5TB$ of data. The data collection took approximately 3 hours. The experiments in this paper were conducted using the single front facing camera of the Ladybug3 panospheric camera, seen at the top of the robot, because this camera has a sufficiently wide field of view to see the entire trellis face, while scanning from the centre of a row, as shown in Figs. 1(b). The data was acquired continuously at $5Hz$, with an individual image resolution of 1232×1616 pixels. Other sensors are used as part of research in tree segmentation and farm modelling.

4.1 Image Analysis

Two set of experiments were performed in this study. The experiments were designed to test both the generalisation of the algorithm for pixel classification, and to verify that the resulting fruit count did correlate to the ground truth. Multi-class segmentation was used to test the generalisation of the algorithm and binary apple/non-apple segmentation was used to evaluate the fruit counting performance.

The dataset consists of over 8000 images with resolutions of 1232 × 1616 pixels. To simplify the labelling process each image was divided into 32 sub-images with 308 × 202 pixels. Overall 90 sub-images were hand labelled with pixel accuracy with multi-class labels, and 800 sub-images were labelled with pixel accuracy with binary apple/non-apple classes. These labelled sub-images were used for the segmentation algorithm training and 2-fold cross validation. Examples of the labelled images are shown in Figure 2.

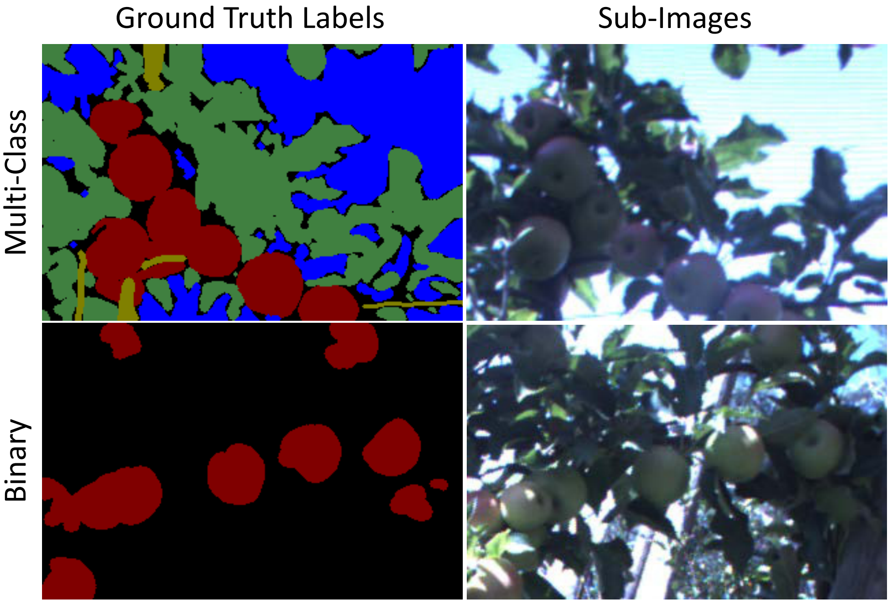


Fig. 2 Example of image labels: 90 sub-images were hand labelled with pixel accuracy with multi-class labels, and 800 sub-images were labelled with pixel accuracy with binary apple/non-apple classes

Apart from the individual class pixel classification accuracy, three evaluation metrics were applied, the global, average accuracy and the F measure. The global accuracy measures the number of correctly classified pixels of the entire dataset whereas the average accuracy measures the average performance over all classes. The F measure was computed by averaging the F-measure of the individual classes.

4.1.1 Multi-class Fruit Segmentation

The first experiment performed multi-class segmentation with the same setting as [2] to classify fruits, leaves, branches, sky and ground. The main difference was that the fruit in this paper were apples instead of almonds. The objective of this experiment was to test the generalisation of the multi-class segmentation algorithm on different fruits, as well as the same fruit (apple) with different colours (red and green).

4.1.2 Apple Counting

The second experiment was designed to evaluate the fruit counting performance. With this objective in mind the algorithm was re-trained to specialise in apple/non-apple binary classification. This was done because in general, the classification accuracy improves with decreasing number of classes. In addition to the binary class segmentation evaluation using the hand labelled image data, the apple farmer provided us with the ground truth count and weight over several rows of the apple farm shown in Fig. 3 by using an automated post-harvest weighing and counting machine. The apple trees are grown on a V shaped trellis known as Güttingen V. Each row has two sides, the rows are numbered sequentially and the A-sides are west facing whereas the B-sides are east facing. The ground truth counts were provided as totals

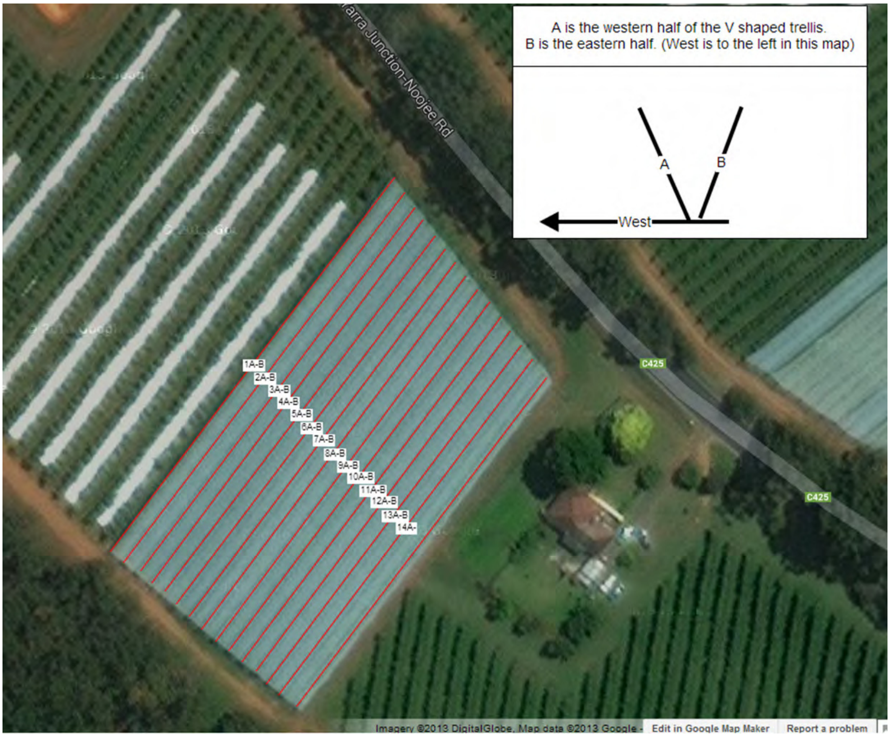

A is the western half of the V shaped trellis.
B is the eastern half. (West is to the left in this map)
A
B
West
Yarra Junction-Noojee Rd
C425
C425
1A-B
2A-B
3A-B
4A-B
5A-B
6A-B
7A-B
8A-B
9A-B
10A-B
11A-B
12A-B
13A-B
14A-
Imagery ©2013 DigitalGlobe, Map data ©2013 Google - Edit in Google Map Maker Report a problem

Fig. 3 The Map of the Farm: The apple trees are grown on a V shaped trellis (Güttingen V). The rows are numbered sequentially, each row has two sides and the A-sides are facing west whereas the B-sides are facing east. The robot surveyed the entire labelled area and the farmer provided the ground truth apple count and weight for each row face for the majority of the surveyed area.

for each side of each row, and were used to evaluate the correlation between the algorithm count and the ground truth.

To normalise the pixels counts provided by the algorithm, all the images and pixel class probabilities were first undistorted using the camera calibration data. Secondly, by using the navigation solution the algorithm sampled images every 0.5 metres along each row to minimise overlap. Finally the total pixel count per side of the rows were normalised by dividing by the number of images sampled.

5 Results

5.1 Multi-class Fruit Segmentation

The multi-class segmentation results are shown in Fig. 4 and Table 1. The dataset consists of leaves, apples and tree trunks at different lighting conditions and scales. The algorithm was able to segment various objects and also the background scene (sky and ground). The confusion matrix shows that the majority of mis-classification occurs between the apple and leaf classes, this is largely due to the colour and texture similarity between the leaves and green apples.

Table 2 also shows the overall performance in apple segmentation compared to almond segmentation shown in [2] using the same multi-scale feature learning algorithm. The segmentation algorithm performed slightly better on the apple dataset (F score 87.3) compared to the almond dataset (F score 84.8). The algorithm generalised well on two different fruit applications, and both show that fruit and leaves are the hardest to classify.

5.2 Apple Fruit Counting

Fig. 5 shows the binary apple/non-apple pixel classification results. The classification algorithm took the image collected by the robotic platform shown in (a), returned the probability of each pixel belonging to the apple class shown in (b), the class probability map was then thresholded at 95% , and erosion was applied to clean up the noise and dilation to join partially occluded apples shown in (c), followed by the apple detection using circular Hough transform shown in (d).

As expected the classification accuracy improved as the number of classes was reduced from 5 down to only apple and non-apple. For the apple counting application, the algorithm was re-trained to only perform binary classification between apple and non-apple class. The classification accuracy was 93.3 % for the apple class and 87.7 % for the non-apple class. Compared to the multi-class experiment where the apple classification accuracy was 71%.

The normalised pixel count was computed by summing the apple pixels in the images for each side of the rows and dividing by the number of images. Separately, the apple count was generated by detecting circular regions in the class map using circular Hough transform. Overall the algorithm undercounted the fruit due to shading and occlusion, however the undercounting was consistent. This was demonstrated

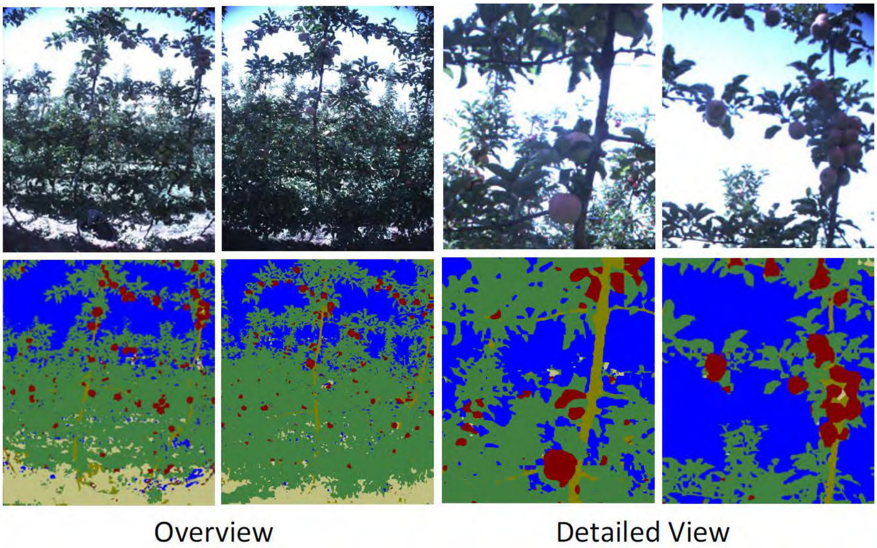


Fig. 4 Apple orchard multi-class segmentation qualitative results: This dataset was collected during an orchard surveying mission aiming to automate the yield estimation and harvesting process. The class colour code: Apples are red, leaves are green, branches are brown, sky is blue and the ground is yellow.

Table 1 Confusion matrix for pixel multi-class classification: The classifier performs well in the leaves and the sky classes. The majority of mis-classification occurs between leaves and green apples.

	leaves	**apples**	**trunk**	**ground**	**sky**
leaves	96.7	25.2	17.4	7.4	0.6
apples	2.0	71.0	7.7	0.1	0.1
trunk	0.5	2.2	71.8	1.9	0.1
ground	0.3	0.6	2.6	86.9	0.0
sky	0.4	1.1	0.5	3.6	99.1

Table 2 Multi-class segmentation performance comparison between previous work on almond and this paper on apple. The learning algorithm performed well across different types of fruit with different appearances.

	Leaves	Fruits	Trunk	Ground	Sky	Global	Average	F Measure
Almond [2]	94.0	69.8	72.8	89.1	95.8	86.9	84.3	84.8
Apple	96.7	71.0	71.9	86.9	99.1	92.5	85.1	87.3

in the comparison between the pixel and fruit count produced by the algorithm, and the ground truth fruit count and weight, shown in Fig. 6. Positive correlations can

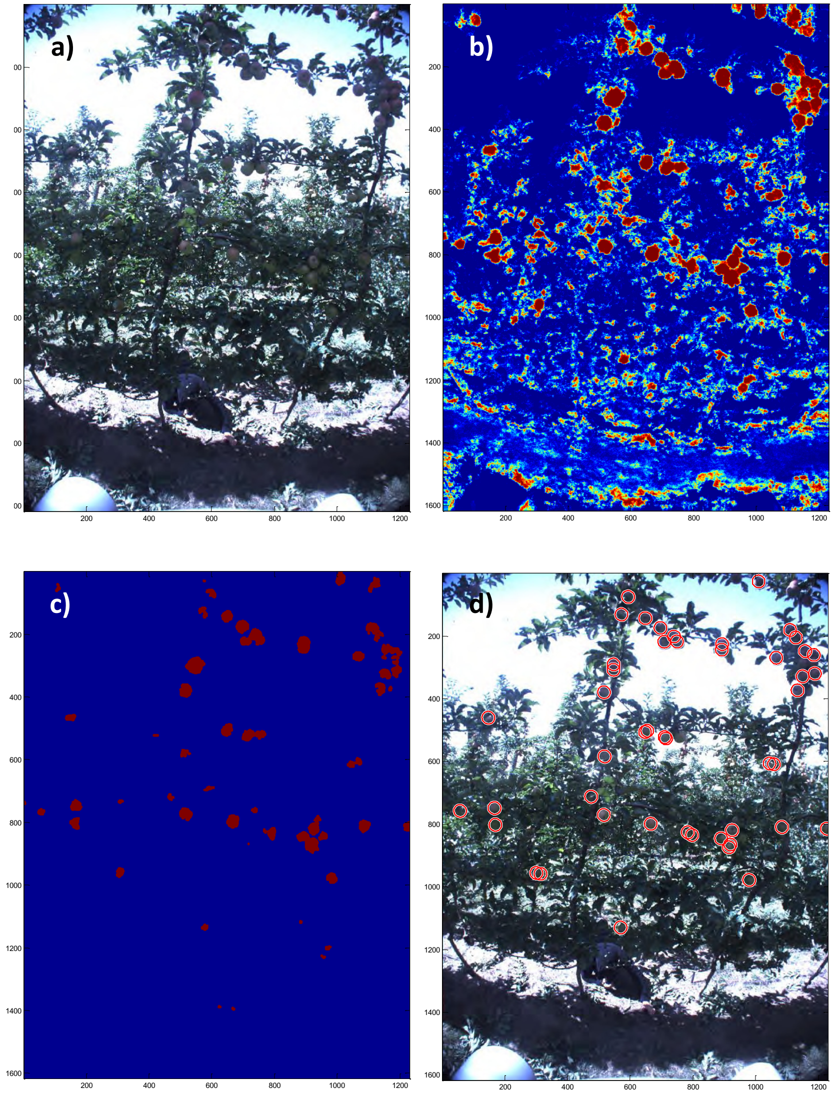


Fig. 5 Apple Pixel: a) Original Image. b) Classifier confidence output, higher confidence is shown in red and lower confidence is shown in blue. c) Confidence threshold at 0.95 and with morphological operations to clean up the noise d) Apple detection with circular Hough transform.

be observed in all plots, with the strongest correlation ($R^2 = 0.810$) between the algorithm fruit count and the ground truth fruit count. Overall higher correlation was

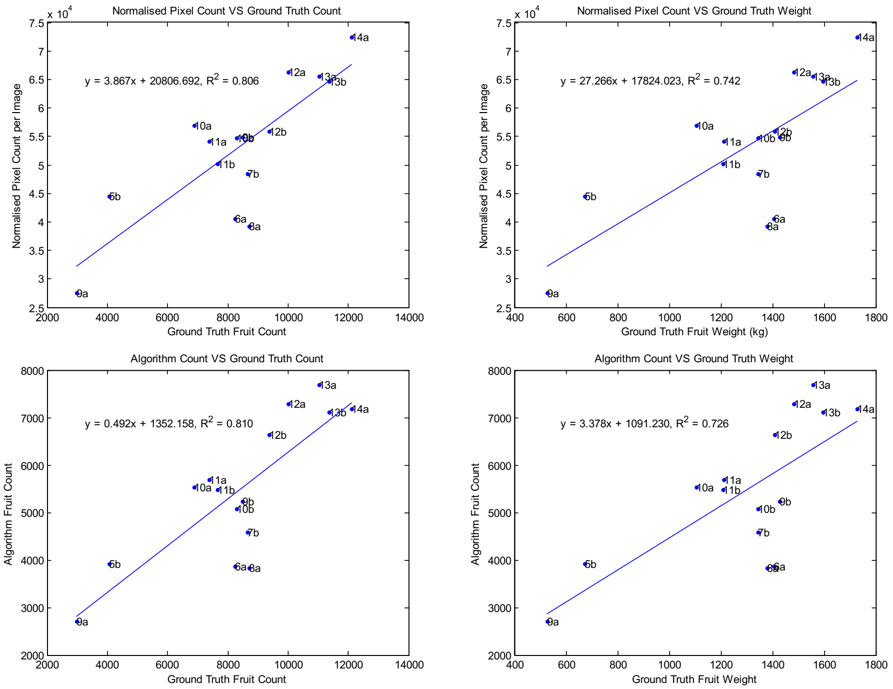


Fig. 6 Algorithm counts versus ground truth counts: Positive correlation is observed in all plots. The strongest correlation occurred between the algorithm fruit count and the ground truth fruit count, demonstrating the algorithm's ability to infer actual fruit count. The row naming convention is according to the map shown in Fig. 3.

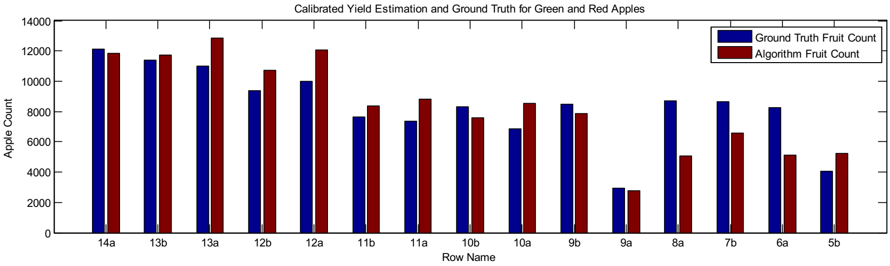


Fig. 7 Calibrated algorithm count versus ground truth count: The algorithm and ground truth fruit count matched well in the first 10 sides. The row naming convention is according to the map shown in Fig. 3.

observed between the algorithm and ground truth when comparing to fruit count rather than weight.

Finally the algorithm fruit count can be calibrated using the linear equation relating ground truth to the algorithm output. The calibrated fruit count is shown in

Fig. 7. The algorithm and ground truth match well for the first 10 sides and has a poor match for three sides 8a, 7b and 6a in particular.

6 Discussion

The results showed that the algorithm performed well for the apple fruit segmentation tasks. In this paper the multi-class segmentation achieved 71 % accuracy on the apple class, and when re-trained to specialise in apple/non-apple classification the accuracy improved to 93.3 %. This paper also showed that the apple count can be estimated by using the pixel classification output. Although the algorithm undercounted the apples due to factors such as occlusion and shading, the algorithm output was consistent. The results showed high correlation between the algorithm fruit count to the ground truth fruit count provided by the apple farmer.

There were several outliers and it is ongoing work to determine the root cause in these cases, to make the algorithm more robust. The farmer also reported logistical problems with the harvest weighing procedure, as it is not standard practice to measure the yield per row and we are working together to produce the most accurate ground truth for the next season.

This work focused on fruit segmentation and counting in outdoor natural lighting conditions. The same method used here could be applied to natural or controlled lighting, as long as training data were supplied for the chosen condition. We expect the performance of this method to be higher in controlled conditions than under natural lighting, due to the increased stability of the appearance of the fruit, but this remains to be verified experimentally.

One limitation of the algorithm is that it currently does not run in real time, instead it runs at 30 seconds per frame. This is acceptable for yield estimation tasks but will need further optimisation for future autonomous harvesting applications. There are two potential solutions. The first is to down sample the image, the current image resolution of 1232×1616 pixels is more than enough to resolve large fruits such as apples. The second is to process a smaller subset of frames with reduces overlap, while still maintaining full coverage.

7 Conclusion

This paper presented an approach for multi-class image segmentation for an apple fruit segmentation application. This algorithm has been successfully applied to almond segmentation previously, and this paper proved that by providing new training data, the algorithm can be generalised to apple segmentation without modification. In addition, the experiments showed that by sampling the classification output with the aid of the navigation data, the algorithm is able to provide reliable apple yield estimation.

Compared to the existing work, the algorithm developed in this paper is able to work in natural lighting conditions and is able to detect both red and green apples using images from a monocular camera.

Acknowledgements. We would like to thank Kevin Sanders for his support during the field trials at his apple orchard. This work is supported in part by the Australian Centre for Field Robotics at the University of Sydney and Horticulture Australia Limited through project AH11009 Autonomous Perception Systems for Horticulture Tree Crops.

References

1. Zhang, Q., Pierce, F.J.: Agricultural Automation: Fundamentals and Practices. CRC Press LLC (2013)
2. Hung, C., Nieto, J., Taylor, Z., Underwood, J., Sukkarieh, S.: Orchard Fruit Segmentation using Multi-spectral Feature Learning. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2013)
3. Dey, D., Mummert, L., Sukthankar, R.: Classification of plant structures from uncalibrated image sequences. In: 2012 IEEE Workshop on Applications of Computer Vision (WACV), pp. 329–336. IEEE (2012)
4. Nuske, S., Achar, S., Bates, T., Narasimhan, S., Singh, S.: Yield estimation in vineyards by visual grape detection. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2352–2358. IEEE (2011)
5. Payne, A., Walsh, K., Subedi, P., Jarvis, D.: Estimation of mango crop yield using image analysis–segmentation method. Computers and Electronics in Agriculture 91, 57–64 (2013)
6. Bulanon, D., Burks, T., Alchanatis, V.: Image fusion of visible and thermal images for fruit detection. Biosystems Engineering 103(1), 12–22 (2009)
7. Wang, Q., Nuske, S., Bergerman, M., Singh, S.: Automated crop yield estimation for apple orchards. In: 13th International Symposium on Experimental Robotics (2012)
8. Aggelopoulou, A., Bochtis, D., Fountas, S., Swain, K.C., Gemtos, T., Nanos, G.: Yield prediction in apple orchards based on image processing. Precision Agriculture 12(3), 448–456 (2011)
9. Jimenez, A., Ceres, R., Pons, J., et al.: A survey of computer vision methods for locating fruit on trees. Transactions of the ASAE-American Society of Agricultural Engineers 43(6), 1911–1920 (2000)
10. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., Rother, C.: A comparative study of energy minimization methods for markov random fields with smoothness-based priors. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1068–1080 (2007)
11. Boykov, Y., Jolly, M.: Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In: Proceedings of the Eighth IEEE International Conference on Computer Vision, ICCV 2001, vol. 1, pp. 105–112. IEEE (2001)
12. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. ACM Transactions on Graphics (TOG) 23(3), 309–314 (2004)

13. Ladicky, L., Russell, C., Kohli, P., Torr, P.: Associative hierarchical crfs for object class image segmentation. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 739–746. IEEE (2009)
14. Lee, H., Ekanadham, C., Ng, A.: Sparse deep belief net model for visual area v2. Advances in Neural Information Processing Systems 19 (2007)
15. Atherton, T.J., Kerbyson, D.J.: Size invariant circle detection. Image and Vision Computing 17(11), 795–803 (1999)

Part X
Search and Rescue Robots

Visual and Inertial Odometry for a Disaster Recovery Humanoid

Michael George, Jean-Philippe Tardif, and Alonzo Kelly

Abstract. Disaster recovery robots must operate in unstructured environments where wheeled or tracked motion may not be feasible or where it may be subject to extreme slip. Many industrial disaster scenarios also preclude reliance on GNSS or other external signals as robots are deployed indoors or underground. Two of the candidates for precise positioning in these scenarios are visual odometry and inertial navigation. This paper presents some practical experience in the design and analysis of a combined visual and inertial odometry system for the Carnegie Mellon University Highly Intelligent Mobile Platform (CHIMP); a humanoid robot competing in the DARPA Robotics Challenge.

1 Introduction

Odometry is a form of dead reckoning which infers position from measurements of wheel or track rotation [20]. The canonical odometry techniques in 2D robotics involve one or two wheel encoders, whose measurements, over a given interval are passed through a simple kinematic model to produce an estimate of change in position. The term has been extended to other techniques which make similar measurements using different sensors, most notably visual odometry [15] which constructs position estimates from streams of images.

Odometry systems are often used in combination with absolute positioning sensors such as the GPS [21] or to seed SLAM algorithms [17]. In certain scenarios odometry alone is sufficient. One such scenario is tele-operation where a human is in the loop and performs the high level spatial reasoning and navigation while

Michael George · Jean-Philippe Tardif
National Robotics Engineering Center, Carnegie Mellon University,
10 40th St., Pittsburgh PA. U.S.A. 15221
e-mail: {mgeorge,tardifj}@rec.ri.cmu.edu

Alonzo Kelly
Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave.,
Pittsburgh PA. U.S.A. 15213
e-mail: alonzo@cmu.edu

L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_34, © Springer International Publishing Switzerland 2015

the odometry system works at a lower level to facilitate tasks like laser and image stitching or control feedback.

Odometry is normally an open loop process. This is sufficient in simple applications or when an absolute positioning system is able to correct accumulated drift but it breaks down in the presence of extreme wheel slip or terrain that is impassable to wheeled vehicles. CHIMP contains both traditional and visual odometry systems along with a high precision IMU and a GPS receiver. These subsystems are combined into a single best estimate of the robot's motion for use in planning, control and perception under the control of a remote operator.

This paper describes the design and initial results for the combination of inertial navigation and visual odometry on CHIMP.

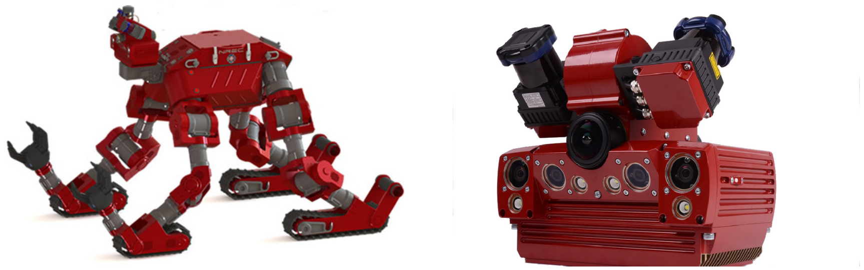
2 CHIMP Robot

The Carnegie Mellon University Highly Intelligent Mobile Platform is the Tartan Rescue (`http://www.rec.ri.cmu.edu/projects/tartanrescue/`) team entry in the ongoing DARPA Robotics Challenge (DRC). CHIMP is a statically stable humanoid robot that can also drive on two or four legs. The DRC event is designed to test competing robots in typical disaster recovery tasks like clearing rubble and closing valves and in ambitious locomotion strategies like climbing ladders and driving utility vehicles. CHIMP has a visual odometry system that uses a custom stereo camera pair composed of Pixim Inc. Seawolf cameras, a Honeywell Inc. HG9900 navigation grade IMU, a Novatel Inc. OEM628 GPS receiver and a mix of relative and absolute encoders on its joints and tracks. The visual odometry subsystem runs on a desktop grade processor as part of CHIMP's ROS based computing platform while the inertial navigation and data fusion algorithms run on an embedded processor in CHIMP's torso. The two subsystems communicate via user datagram protocol over CHIMP's onboard network.

3 Positioning System Design for CHIMP

CHIMP's positioning system is designed primarily for reliability and redundancy during tele-operation and supervised autonomy. It prioritizes locally consistent position information over global accuracy and it prioritizes decentralized and redundant sources of this position information over centralized and optimal fusion.

Disaster recovery scenarios are characterized by challenging and unpredictable environments where the robot may encounter collapsed buildings, mine shafts, unstructured debris and radiation or biological hazards. Global position systems are unreliable or completely unavailable indoors and underground, odometry is subject to gross errors due to slip or debris and cameras may be obscured by dust, gases or just a lack of ambient light. An inertial navigation system, in contrast, is self-contained and unaffected by environmental conditions and provides a fall-back in all conditions.



(a) (b)

Fig. 1 A rendering of CHIMP in four limb driving mode (a). CHIMP's tracks and joints are fitted with encoders for odometry. Its torso contains a Honeywell Inc. HG9900 IMU. Close-up of CHIMP's head (b), containing Pixim Inc. Seawolf cameras in a stereo configuration for visual odometry along with other perception sensors.

The work presented here fits into a category of visual inertial fusion characterized by an aided inertial navigation Kalman filter as the central component [4] [14]. This arrangement gracefully degrades if subsystems, which provide measurements to the Kalman filter, go offline or fail. It can also be extremely computationally efficient, as demonstrated in [4] where a design for a similar filter for spacecraft navigation of small interplanetary bodies is presented. Our own implementation is deployed in real-time on an embedded 400MHz processor. We add some additional modifications to these works. CHIMP's positioning system automatically detects periods of zero motion (In translation and rotation, relative to Earth) and corrects positioning error accordingly. The design also accounts for large delays (100+ ms) in the processing of the visual odometry solution when it arrives at the Kalman filter.

There are alternative approaches to integrating inertial and visual sensors which relax the Kalman filter assumptions and potentially extract better performance [12] [13]. The trade-off is computational expense and less graceful degradation with component failure.

Due to space constraints we concentrate on the inertial and visual components of CHIMP's positioning system and specifically their integration. The GPS and odometry components are simple extensions of this framework and some results including odometry measurements are also presented.

4 Inertial Navigation on CHIMP

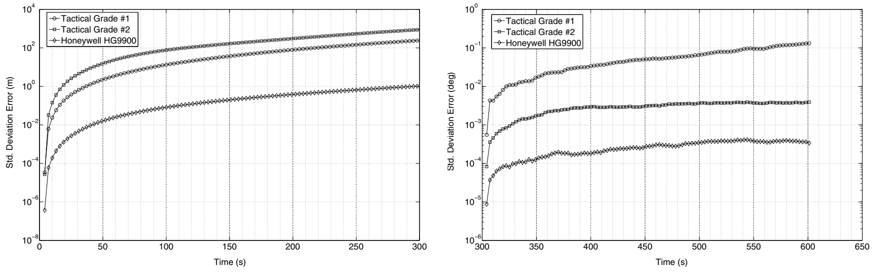
Inertial navigation is a mature field in its traditional areas of application: aircraft, ships and military components. There are many excellent texts that develop the details [18] [21]. However, these references typically assume access to absolute Earth referenced initial conditions: heading, latitude, height above the Earth's reference ellipsoid and a database of local gravity anomalies. For most field robots at least some of these will be unknown or known only approximately. The more precise the IMU the more important these initial conditions become.

An approximate summary of IMU categories is given in Table 1. Most robotics applications that require an IMU use a tactical grade unit with cost and size being the major factors. Tactical grade IMUs are capable of tracking orientation for extended periods but cannot be used to track position for more than a few seconds in unaided operation. Commercial grade sensors may be found in small robots where size and cost must be strictly minimized such as micro air vehicles [5]. With clever calibration they may also be capable of tracking orientation in some applications.

Table 1 Approximate IMU categories

IMU grade	Size	Error Characteristics	Cost
Navigation	$1600cm^3$	$<$ 1600 meters per hour in position. $<$ $1/3000$ degrees per hour in orientation.	\$70,000+
Tactical	$100cm^3$	$<$ 10^7 meters per hour in position. $<$ 50 degrees per hour in orientation.	\$2,000+
Commercial	$1cm^3$	$\sim$ 10^9 meters per hour in position. $<$ 3600 degrees per hour in orientation.	\$15+

A Monte Carlo simulation of a typical scenario for CHIMP was performed before selecting the IMU. Position and orientation errors for three devices that were considered in a trade study are shown in Figure 2.



(a) Position errors

(b) Orientation errors

Fig. 2 Standard deviation of position and orientation errors for 5 minutes in simulated DRC scenario using an unaided IMU. Results generated from Monte Carlo analysis based on simulated IMU and robot trajectories.

The navigation grade Honeywell HG9900 [1] was selected for its accuracy in the worst case scenario that all other aiding systems fail. The resulting requirement for absolute initial conditions is satisfied in two ways: the IMU can be used to gyrocompass for initial heading and the remote operator must enter a configuration containing approximate latitude and altitude of operation. This information is available from public sources like Google Earth and US government agencies [2].

4.1 Inertial Navigation Equations

The fundamental equations of inertial navigation are repeated here to introduce the notation used in following sections [21]. They consist of an orientation differential equation, integrated once, and a velocity differential equation, integrated twice.

$$\dot{\mathbf{C}}_b^n = \mathbf{C}_b^n[\omega_{ib}^b \times] - [\omega_{in}^n \times]\mathbf{C}_b^n \tag{1}$$

$$\dot{\mathbf{v}}^n = \mathbf{C}_b^n \mathbf{f}^b - (\omega_{en}^n + 2\omega_{ie}^n) \times \mathbf{v}^n + \mathbf{g}_p^n, \tag{2}$$

where the following definitions are used

$\mathbf{x}^z$	Vector $\mathbf{x}$ expressed in frame z
$\mathbf{C}_y^z$	Rotation matrix mapping frame y to frame z such that $\mathbf{x}^z = \mathbf{C}_y^z \mathbf{x}^y$
$[\mathbf{x}\times]$	Cross product matrix version of $\mathbf{x}$ such that $[\mathbf{x}\times]\mathbf{y} = \mathbf{x} \times \mathbf{y}$
ω_{xy}^z	Angular rate vector of frame y relative to frame x expressed in frame z
$\mathbf{g}_p^n$	Plumb-bob gravity, *i.e.* gravitation plus centripetal effects
$\mathbf{v}$	IMU velocity vector
$\mathbf{f}$	Non-gravitational acceleration *a.k.a.* specific force.
i	Inertial frame of reference
b	Body frame of reference, assumed to coincident with the IMU frame
n	Navigation frame of reference, a design choice with subtle implications
ω_{ib}^b	Angular rate vector of IMU measured by gyroscopes
ω_{ie}^n	Angular rate vector of Earth
ω_{en}^n	Angular rate vector of navigation frame relative to Earth, in this work zero

5 Visual Odometry

CHIMP's visual odometry system is largely inspired by the work of Nister *et al.* [15]. It produces a solution based on incremental structure from motion with key frame selection and sparse local bundle adjustment [8]. It makes use of a stereo pair to eliminate scale ambiguity in the resulting solution.

Like most visual odometry systems, it solves consecutive relative pose problems from image correspondences and scene structure estimates. Details of an earlier implementation with many similarities can be found in [19] and the references therein.

The algorithm produces relative pose solutions at frame rate. Key frames are selected when motion between consecutive frames is sufficiently large. New scene points are added, old scene points are re-triangulated and bundle adjustment is applied only on key frames. On other frames, only tracking and pose estimation based on the reconstructed scene is performed. Generally speaking, using key frames results in drift being proportional to distance rather than time. In addition, key frames eliminate trajectory drift when the robot is standing still but oscillating which happens frequently while CHIMP is performing manipulation tasks.

Stereo tracking relies on Harris corners computed with sub-pixel accuracy and matched using normalized cross-correlation. We develop an algorithm that combines both temporal and stereo information by doing tracking and stereo matching together.

Pose estimation given a set of reconstructed points and corresponding image locations is generally referred to as the PnP problem [10]. Robust estimation is achieved using random sampling (RANSAC) with solution candidates computed with subsets of 3 correspondences. This is the so-called P3P problem and well established solutions exist [16]. Given our high frame rate and relatively small angular velocity, we implement a simpler and faster solution that assumes small orientation changes. For simplicity, assume a coordinate system centered on the left camera. We first transform the reconstructed points in the coordinate frame of left camera, c at frame $k-1$. At frame k, we then consider a projection function for the left camera of the form:

$$\begin{aligned}\mathbf{x}(k) &\propto \mathbf{P}\mathbf{X}(k-1)\\ &\propto \mathbf{C}_{c(k-1)}^{c(k)}\left(\mathbf{I}_3 \;\; -\mathbf{p}_{c(k)}^{c(k-1)}\right)\mathbf{X}(k-1)\end{aligned}$$

where $\mathbf{P}$ is a 3×4 projection matrix, $\mathbf{C}_{c(k-1)}^{c(k)}$ is a rotation matrix of small angles

$$\mathbf{C}_{c(k-1)}^{c(k)} \approx \begin{pmatrix} 1 & u & v \\ -u & 1 & w \\ -v & -w & 1 \end{pmatrix},$$

$\mathbf{p}_{c(k)}^{c(k-1)}$ is a 3×1 translation vector representing the camera position at time k in the camera frame at time $k-1$, $\mathbf{x}(k)$ is the correspondence in image coordinates, $\mathbf{X}(k-1)$ is the 3D location of the correspondence and $\propto$ is equality up to scale.

Assuming measurements uncorrupted by noise and a perfect rotation approximation, three linear constraints over the unknowns can be obtained. These are given by

$$[\mathbf{x}(k)\times]\mathbf{P}\mathbf{X}(k-1) = \mathbf{0}.$$

Only two out of the three constraints are linearly independent, so a minimum of three correspondences are required to obtain a solution. Similar constraints can be derived for the right camera. Since measurements are corrupted by noise, we estimate the unknowns by minimizing the constraints under least-squares. In theory, only two stereo correspondences are necessary to obtain a solution. However, we have found that using three provides more reliable results. Once a solution is found, $\mathbf{C}_{c(k-1)}^{c(k)}$ is converted to a orthogonal matrix by using u, v, w as Euler angles. Finally, the solution provided by random sampling is improved using robust iterative refinement.

The number of features being tracked is automatically adjusted so it is maximized while ensuring real-time performance. In practice, the system can comfortably track 300 stereo features at 30 Hz on full resolution 720×487 stereo images. The most computationally intensive steps are the corner detection at 6ms for a stereo pair and bundle adjustment at 5ms. Pose estimation, stereo and tracking each take under 2ms. As a result, key frames are processed in under 20ms and non-key frames in under 15ms.

Integration with the rest of the pose system can be accomplished in multiple ways. At each frame k, the following potential measurements are available to the Kalman filter.

1- Relative position and orientation increments, *i.e.* the immediate outputs of the visual odometry

$$\mathbf{p}_{c(k)}^{c(k-1)} \tag{3}$$

$$\mathbf{C}_{c(k)}^{c(k-1)} = \left(\mathbf{C}_{c(k-1)}^{c(k)}\right)^{\mathsf{T}} \tag{4}$$

2- Global pose. Given initial conditions $\mathbf{p}_{c(0)}^{n}$ and $\mathbf{C}_{c(0)}^{n}$ from the inertial navigation system, global pose at increment k can be calculated via

$$\mathbf{p}_{c(k)}^{n} = \mathbf{p}_{c(0)}^{n} + \sum_{i=1}^{k} \mathbf{C}_{c(i-1)}^{n} \mathbf{p}_{c(i)}^{c(i-1)} \tag{5}$$

$$\mathbf{C}_{c(k)}^{n} = \mathbf{C}_{c(0)}^{n} \prod_{i=1}^{k} \mathbf{C}_{c(i)}^{c(i-1)} \tag{6}$$

3- Velocity. Given our high frame rate (30 Hz) the translation component can be numerically differentiated to produce a velocity signal

$$\mathbf{v}^{c(k-\frac{1}{2})} = \frac{\mathbf{p}_{c(k)}^{c(k-1)}}{t(k) - t(k-1)} \tag{7}$$

where the $k - \frac{1}{2}$ notation indicates the velocity is valid at the mid-point of the frame interval.

Using pose increments allows a decoupled integration between the Kalman filter and the visual odometry [19]. However, because the global pose of the visual odometry is not considered, the filtered solution can drift even when the robot is standing still. For that reason, the visual odometry estimates whether the robot is under motion or not and provides that information to the filter. A couple of criteria are used. First, it is verified that tracked features move by less than one pixel over the last second. Then the average relative position of the camera within the last second is compared to a small threshold which is determined empirically. This is combined with a similar determination from the IMU and a pseudo-measurement of zero velocity is applied to the Kalman filter.

6 Kalman Filter for Visual and Inertial Odometry

Visual and inertial estimates are blended in a modified extended Kalman filter. The Kalman filter has several advantages in this approach, it's a well understood algorithm with a large body of literature, in particular as it relates to aided inertial navigation. The filter uses the common indirect inertial navigation state model [21] which keeps subsystems decoupled and allows the inertial navigation and visual odometry to continue operating normally in the event of filter failure. In addition it can be embedded on low power and hardened processors for demanding environments.

The standard Kalman filter equations are suitable for incorporating an absolute visual odometry pose or a velocity measurement (Equations 5-6 and Equation 7). They are not suitable for incorporating the fundamental visual odometry outputs (Equations 3-4), the relative state measurements. A modified version able to incorporate this type of measurement can be found in [6]. The modified Kalman update equations take the following form

$$\begin{aligned}
\mathbf{z}_k &= \mathbf{h}(\mathbf{x}_k, \mathbf{x}_{k-1}) \\
\mathbf{H}_k &= \frac{\partial \mathbf{h}(\mathbf{x}_k, \mathbf{x}_{k-1})}{\partial \mathbf{x}_k} \\
\mathbf{J}_k &= \frac{\partial \mathbf{h}(\mathbf{x}_k, \mathbf{x}_{k-1})}{\partial \mathbf{x}_{k-1}} \\
\mathbf{L}_k &= \mathbf{H}_k \mathbf{P}_k \mathbf{H}_k^\mathsf{T} + \mathbf{R}_k + \mathbf{J}_k \mathbf{P}_{k-1} \boldsymbol{\Phi}_{k-1}^\mathsf{T} \mathbf{H}_k + \mathbf{H}_k \boldsymbol{\Phi}_{k-1} \mathbf{P}_{k-1} \mathbf{J}_k^\mathsf{T} + \mathbf{J}_k \mathbf{P}_{k-1} \mathbf{J}_k^\mathsf{T} \\
\mathbf{K}_k &= (\mathbf{P}_k \mathbf{H}_k^\mathsf{T} + \boldsymbol{\Phi}_{k-1} \mathbf{P}_{k-1} \mathbf{J}_k^\mathsf{T}) \mathbf{L}_k^{-1} \\
\mathbf{P}_k^+ &= \mathbf{P}_k - \mathbf{K}_k \mathbf{L}_k \mathbf{K}_k^\mathsf{T} \\
\mathbf{x}_k^+ &= \mathbf{x}_k + \mathbf{K}_k (\mathbf{z}_k - \mathbf{h}(\mathbf{x}_k, \mathbf{x}_{k-1}))
\end{aligned}$$

where the following definitions apply

$\mathbf{z}$	Measurement vector
$\mathbf{h}$	Measurement function mapping states to measurements
$\mathbf{H}$	Measurement matrix, jacobian of $\mathbf{h}$ w.r.t $\mathbf{x}_k$
$\mathbf{J}$	Measurement matrix, jacobian of $\mathbf{h}$ w.r.t $\mathbf{x}_{k-1}$
$\mathbf{P}$	State covariance matrix
$\boldsymbol{\Phi}$	State transition matrix mapping state evolution over time
$\mathbf{K}$	Kalman gain matrix
$\mathbf{x}_k$	State mean at time k

These equations are functions of variables at time k and $k-1$, as indicated by the subscripts. For clarity this notation is slightly different from other sections where time indices are indicated in parentheses. The $k-1$ parameters are stored in a memory buffer or retrodicted explicitly using the current $\mathbf{x}$ and $\mathbf{P}$ variables with saved versions of $\boldsymbol{\Phi}_{k-1}$ and the noise inputs [3].

7 Delayed Measurements

Visual odometry is performed in its own computing cores separate from the embedded Kalman filter. While the computation is achieved in real-time (20ms) additional communications delays and image capture through embedded hardware introduce a non-trivial time synchronization problem. Total delays of up to 100ms can occur between capturing an image and that data being available to the Kalman filter. In this case the previous relative state update relations can be modified for the absolute or velocity forms of the visual odometry (Equations 5-6 and Equation 7) by setting the current measurement jacobian to zero, giving

$$\begin{aligned}
\mathbf{H}_k &= \mathbf{0} \\
\mathbf{z}_k &= \mathbf{h}(\mathbf{x}_{k-1}) \\
\mathbf{L}_k &= \mathbf{R}_k + \mathbf{J}_k \mathbf{P}_{k-1} \mathbf{J}_k^\mathsf{T} \\
\mathbf{K}_k &= (\boldsymbol{\Phi}_{k-1} \mathbf{P}_{k-1} \mathbf{J}_k^\mathsf{T}) \mathbf{L}_k^{-1} \\
\mathbf{P}_k^+ &= \mathbf{P}_k - \mathbf{K}_k \mathbf{L}_k \mathbf{K}_k^\mathsf{T} \\
\mathbf{x}_k^+ &= \mathbf{x}_k + \mathbf{K}_k (\mathbf{z}_k - \mathbf{h}(\mathbf{x}_{k-1}))
\end{aligned}$$

These now represent the equations necessary to apply a delayed measurement that was valid at time $k-1$ but arrived at the filter at time k. It should be noted that the interval between k's is not required to be constant.

8 Calibration

It is necessary to calibrate the relative position ($\mathbf{p}_{b\to c}^b$) and orientation ($\mathbf{C}_c^b$) of the visual odometry cameras and IMU on CHIMP. Assuming a rigid link the following relation holds, where k and j are time indices with $k > j$ and b represents the IMU frame while c is the camera frame

$$\mathbf{C}_{b(k)}^{b(j)} = \mathbf{C}_c^b \mathbf{C}_{c(k)}^{c(j)} \mathbf{C}_c^{b\mathsf{T}}$$

This can be converted to a quaternion form and solved for the relative rotation matrix between camera and IMU $\mathbf{C}_c^b$. Details can be found in [7].

For the translation, most methods involve a calibration Kalman filter [11]. These methods are necessarily complex because they are designed for low-cost IMUs where position estimates drift rapidly. For a navigation grade IMU a more direct method is possible. The following relation maps the relative pose increments generated by the visual odometry system with similar increments calculated from the unaided inertial navigation system

$$\mathbf{p}_{c(k)}^{c(k-1)} = \mathbf{C}_b^c \mathbf{C}_n^{b(k-1)} (\mathbf{p}_{b(k)}^n + \mathbf{C}_{b(k)}^n \mathbf{p}_{b\to c}^b - \mathbf{p}_{b(k-1)}^n - \mathbf{C}_{b(k-1)}^n \mathbf{p}_{b\to c}^b)$$

Re-arranging for $\mathbf{p}_{b\to c}^b$, the fixed position offset between IMU and camera

$$\mathbf{C}_b^c (\mathbf{C}_{b(k)}^{b(j)} - \mathbf{I}) \mathbf{p}_{b\to c}^b = \mathbf{p}_{c(k)}^{c(j)} + \mathbf{C}_b^c \mathbf{C}_n^{b(j)} (\mathbf{p}_{b(j)}^n - \mathbf{p}_{b(k)}^n)$$

which stacked over many j, k intervals has the form

$$\mathbf{A}\mathbf{p}^b_{b\to c} = \mathbf{b}$$

A least squares solution is given by $\mathbf{p}^b_{b\to c} = (\mathbf{A}^\intercal\mathbf{A})^{-1}\mathbf{A}^\intercal\mathbf{b}$. This can be transformed into a weighted least squares with the standard inertial error model from the Kalman filter prediction steps used to generate covariances as the IMU solution drifts. Given these covariances, **P**, the weighted least squares solution then becomes

$$\mathbf{p}^b_{b\to c} = (\mathbf{A}^\intercal\mathbf{P}^{-1}\mathbf{A})^{-1}\mathbf{A}^\intercal\mathbf{P}^{-1}\mathbf{b}$$

9 Results

The preceding algorithms were tested on CHIMP in three stages of construction: camera and IMU sensors on a test cart, CHIMP's completed torso on a mobile frame, Figure 3(a) and the finished robot under its own motion, Figure 3. Since CHIMP is designed to operate without access to global positioning, obtaining ground truth is challenging. These results were generated using a combination of the following three techniques

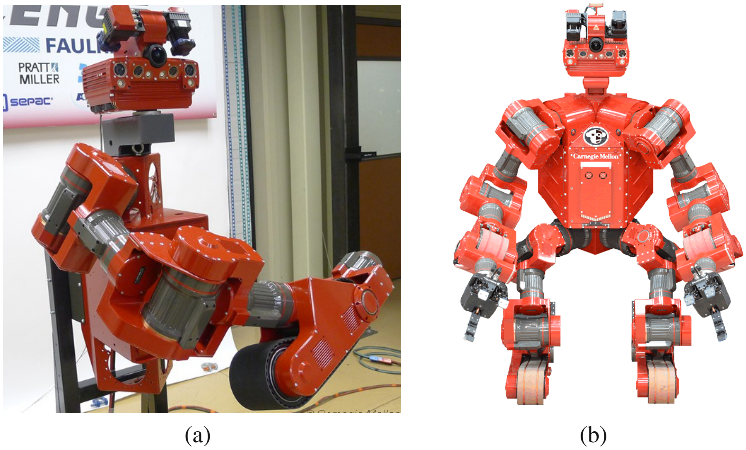
1 In small volumes, CHIMP's positioning system can be compared to motion capture from a Natural Point Inc. Optitrack system.

2 Over longer trajectories a laser line striper is attached to CHIMP to align precisely with known locations on the ground. CHIMP is returned to these locations, typically at the start and end of a test and and error can be read directly as the difference between calculated and true position.

3 Virtual objects are placed in identifiable locations in a 3D reconstruction generated with CHIMP's laser scanners and positioning data. Accumulated error in the positioning data shifts the 3D voxels from the virtual objects. The shift due to positioning error is measurable in an operator control GUI.

CHIMP is tested in scenarios reflecting the DRC tasks, broadly divided into mobility, driving over smooth and rough terrain and manipulation, arm motion and small movements but no overall locomotion. Figures 4 and 5 demonstrate CHIMP a simple mobility scenario where the torso-only test rig was moved through a variety of motions (sweeping arcs, point turns, straight lines etc.). Total accumulated error on returning to the known initial location is 0.72m or 0.6% of distrance traveled for the filtered result using visual odometry velocity measurements (Equation 7).

Figure 6 demonstrates live results from CHIMP in a simulated manipulation scenario where the robot is in a typical manipulation posture (Similar to Figure 3) and is interacting with objects in close proximity. This represents a challenging scenario where the visual odometry system is throttling its CPU usage (By dropping corner features) to accomodate planning, control and perceptions priorities and is viewing a scene at close proximity with self occlusion by moving arms. The left panel of Figure 6 compares three trajectories, visual odometry only, the combined IMU and visual odometry solution and a motion capture ground truth reference. The bottom panels plot horizontal errors (North and East) for the combined solution. The right hand panel illustrates the further addition of velocity data from CHIMPs


(a)

(b)

Fig. 3 A partially constructed CHIMP on a test-rig (a). The sensor head containing visual odometry sensors and a single arm are shown. The IMU is located in a temporary enclosure at the base of a mobile frame (not shown). Fully constructed CHIMP (b). The visual odometry sensors are unchanged. The IMU is located in CHIMP's torso (not visible). For the results in this section CHIMP is remotely operated by a human.

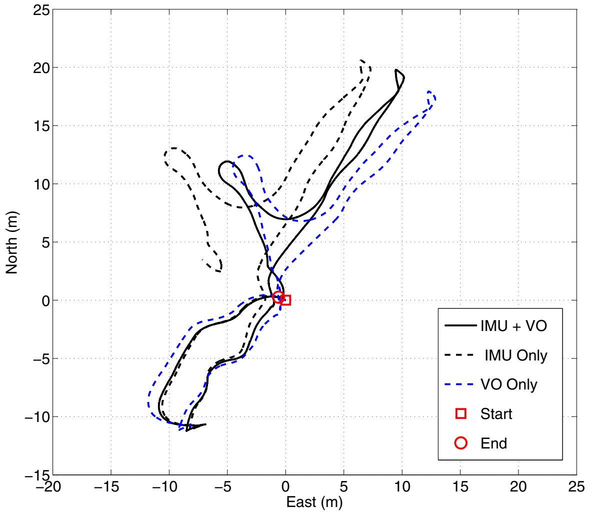


Fig. 4 Indoor path of 105m, with error assessed as closure discrepancy when returning to the origin. 2D error is 72cm or 0.6% of distance. VO Only has an arbitrary initial alignment. IMU only results drift 8m.

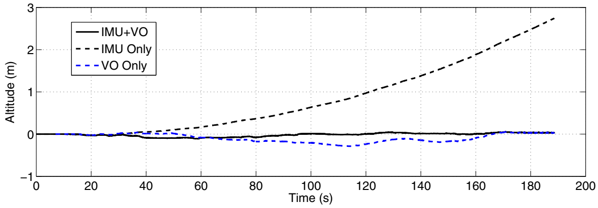


Fig. 5 Indoor path of 105m, with error assessed as closure discrepancy when returning to the origin. Altitude error is 3cm at loop closure. Ground truth altitude is not available in the middle of the segment.

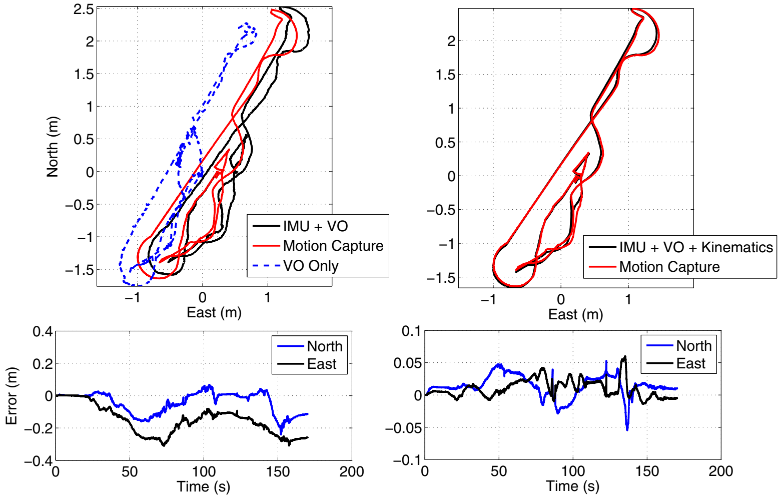


Fig. 6 Short duration manipulation scenario with motion capture ground truth reference. Left, the visual odometry running live on CHIMP is resource constrained and is not able to use as many features in its pose estimation and tracking as that running on the test rig. As a result the combined solution is also less accurate. With the addition of kinematic data from the robots tracks and joints (right) the errors are much reduced. Bottom panels show horizontal errors compared to motion capture ground truth.

joints and tracks which serves to compensate most of the residual error in this scenario.

Figure 7 shows a typical colorized 3D point cloud generated from the positioning, laser and camera sensors. This point cloud is generated open loop with no scan matching or other improvements. For a tele-operated control scenario only the last 10 seconds of the 3D reconstruction are presented to the operator for clarity.

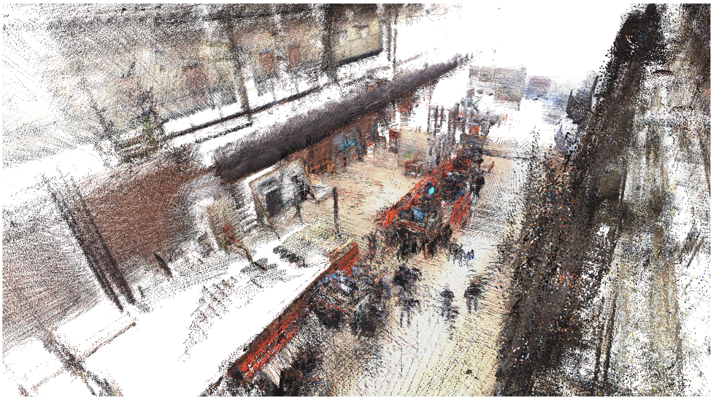


Fig. 7 3D point cloud generated by colorizing laser scans using image data and accumulating points in North, East, Down world frame. No corrections are applied using point cloud data itself, it is stitched open loop using the positioning system described in this paper. The view point is from an elevated mezzanine floor looking down on CHIMPs test bay. Test fixtures like framed doors and an cinder block obstacle course are visible along with red safety barriers running diagonally through the image separating the robot's area behind the barriers from engineers and their desks and computers in front.

10 Ongoing Work

CHIMP is scheduled to compete in the first year of the DRC in late 2013. Further refinement of the current system is expected to continue beyond that based on feedback from this event. An improved 3D mapping capability based on the laser scan matching and a pose graph optimization framework is anticipated.

Acknowledgements. Michel Laverne and Dane Bennington designed and built much of the positioning hardware. Clark Haynes implemented the track and joint odometry subsystem and many others contributed to CHIMPs development.

References

1. Honeywell Aerospace, HG9900 IMU (2009), `http://www51.honeywell.com/aero/common/documents/myaerospacecatalog-documents/MilitaryAC/HG9900_IMU.pdf`
2. National Oceanic and Atmospheric Administration, Marine & Global Geophysical Data Gravity (2013), `http://www.ngdc.noaa.gov/mgg/gravity/`
3. Bar-Shalom, Y.: Update with Out-of-Sequence Measurements in Tracking: Exact Solution. IEEE Trans. Aerosp. Electron. Syst. 38(3), 769–778 (2002)
4. Bayard, D., Brugarolas, P.: On-Board Vision-Based Spacecraft Estimation Algorithm for Small Body Exploration. IEEE Trans. Aerosp. Electron. Syst. 44(1), 243–260 (2008)
5. Bristeau, P.-J., Callou, F., et al.: The Navigation and Control technology inside the AR.Drone micro UAV. In: Proc IFAC World Congress, pp. 1477–1484 (August 2011)

6. Brown, R., Hwang, P.: Introduction to Random Signals and Applied Kalman Filtering, 3rd edn. John Wiley & Sons, New York (1997)
7. Dong-Si, T., Mourikis, A.: Estimator Initialization in Vision-aided Inertial Navigation with Unknown Camera-IMU Calibration. In: Proc IEEE/RSJ IROS, October 7-12, pp. 1064–1071 (2012)
8. Engels, C., Stewenius, H., Nister, D.: Bundle adjustment rules. Photogrammetric Computer Vision 2 (2006)
9. Farrell: Aided Navigation: GPS with High Rate Sensors. McGraw Hill, New York (2008)
10. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge University Press, Cambridge (2000)
11. Hol, J., Schon, T., Gustafsson, F.: A New Algorithm for Calibrating a Combined Camera and IMU Sensor Unit. In: Proc. IEEE ICARCV, December 17-20, pp. 1857–1862 (2008)
12. Indelman, V., Williams, S., et al.: Information fusion in navigation systems via factor graph based incremental smoothing. J. Robot. Auton. Syst. 61(8), 721–738 (2013)
13. Jones, E., Soatto, S.: Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. Int. J. Robot. Res. 30(4), 407–430 (2011)
14. Mourikis, A., Roumeliotis, S.: On the Treatment of Relative-Pose Measurements for Mobile Robot Localization. In: Proc. IEEE ICRA, pp. 2277–2284 (May 2006)
15. Nister, D., Naroditsky, O., Bergen, J.: Visual odometry for ground vehicle applications. J. Field Robot. 23(1), 3–20 (2006)
16. Quan, L., Lan, Z.: Linear N-point camera pose determination. IEEE Trans. Pattern Anal. Mach. Intell. 21(8), 774–780 (1999)
17. Olson, E., Agarwal, P.: Inference on networks of mixtures for robust robot mapping. In: Proc. Robotics: Science and Systems (July 2012)
18. Savage, P.: Strapdown Analytics, 2nd edn. Strapdown Associates Inc., Maple Plain (2007)
19. Tardif, J., George, M., et al.: A new approach to vision-aided inertial navigation. In: Proc. IEEE/RSJ IROS, October 18-22, pp. 4161–4168 (2010)
20. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press, Cambridge (2005)
21. Titterton, D., Weston, J.: Strapdown Inertial Navigation Technology, 2nd edn. IEE Radar, Sonar and Navigation Series 17. IEE, Reston (2004)

Precise Velocity Estimation for Dog Using Its Gait

Naoki Sakaguchi, Kazunori Ohno, Eijiro Takeuchi, and Satoshi Tadokoro

Abstract. We aimed to record and visualize the investigation activities of search and rescue dogs. The dog's trajectory is required to create this visualization, and the dog's velocity needs to be determined to estimate its trajectory. In this study, we examined a method for velocity estimation that uses a dog's gait. We measured a Labrador dog's gaits (walk and trot) and analyzed the gait data. From the gait data, we found that there are cyclic moments when the dog's velocity vector faces its heading direction. This fact enables the reconstruction of the velocity vector $\mathbf{v} = (v_{\mathrm{x}}, v_{\mathrm{y}}, v_{\mathrm{z}})^{\mathrm{T}}$ from the dog's speed $|\mathbf{v}|$ and pose. We devised a precise estimation method for a dog's velocity and evaluated its accuracy. From the evaluation results, we confirmed that the gait-based velocity estimation was more accurate than velocity estimation based on the extended Kalman filter when $|\mathbf{v}|$ was obtained at 1, 5, and 10 Hz. This result can pave the way for using a mobile phone to estimate a dog's trajectory.

1 Introduction

We have researched methods for recording and visualizing the investigation activities of search and rescue (SAR) dogs. SAR dogs can investigate and find victims in forests and at disaster sites using their keen sense of smell. SAR dogs are trained to continue barking when they find victims. A human handler then determines whether a victim is located in the place where the dog is barking, by calling out and looking around for victims. The handler then informs other rescue workers about the place and situation in which the victims are, by using a radio or handwritten memorandum.

However, verbal communication or a handwritten memorandum has limitations in relation to sharing accurate information about a victim and the investigation activ-

Naoki Sakaguchi · Kazunori Ohno · Eijiro Takeuchi · Satochi Tadokoro
Tohoku University, 6-6-01 Aramaki Aza Aoba, Aobaku, Sendai, 980-8579, Japan
e-mail: {sakaguchi,kazunori,
takeuchi,tadokoro}@rm.is.tohoku.ac.jp

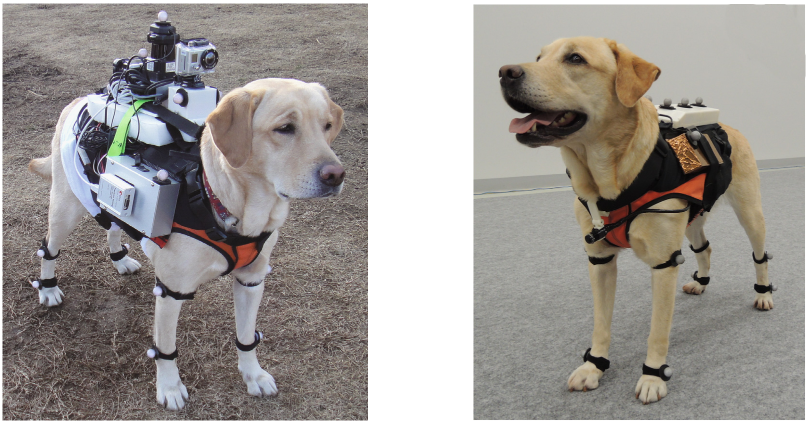
L. Mejias, P. Corke, and J. Roberts (eds.), *Field and Service Robotics*,
Springer Tracts in Advanced Robotics 105,
DOI: 10.1007/978-3-319-07488-7_35, © Springer International Publishing Switzerland 2015

ities. Therefore, we conducted research on methods for recording the investigation activities of SAR dogs.

Figure 1 shows the equipment we developed for recording an SAR dog's investigation activities. The device can record what the dog is looking at, along with its trajectory. Using wireless tablet devices, handlers can share the images and trajectory online. Other researchers have used similar approaches [1, 2, 3]. These studies showed that the use of a camera and GPS is an efficient way to record an SAR dog's investigation activities. In our case, the GPS position data are insufficient as trajectory data. A more continuous and accurate trajectory is needed for sharing accurate information. Therefore, we recorded the investigation activities using a camera, a GPS, and an internal measurement unit (IMU).

The trajectory was estimated by integrating the sensor data from the IMU. Its cumulative error was canceled using the velocity $\mathbf{v}$ and position data $\mathbf{x}$. The velocity vector of a dog, $\mathbf{v}$, consists of three components $(v_{\mathrm{x}}, v_{\mathrm{y}}, v_{\mathrm{z}})$. It is difficult to directly measure the velocity $\mathbf{v}$ for an SAR dog because of the size and weight of the required sensor device. Therefore, we measured the scalar of the velocity vector, $|\mathbf{v}|$, by using a GPS or an optical sensor. We wanted to determine a velocity collection method by using $|\mathbf{v}|$. In this paper, we propose a velocity collection method that analyzes a dog's motion. We found that there were cyclic moments when the dog's velocity vector faced its heading direction. In these moments, the velocity vector could be recovered from its scalar speed $|\mathbf{v}|$ and the dog's pose.

In this study, we analyzed a dog's motion data and developed a velocity collection method. We explain the related works in Sec. 2. The dog's motion was recorded using motion capture. In Sec. 3, we show how the motion was analyzed to find a new rule for velocity estimation. We propose a velocity collection method in Sec. 4, and evaluate its efficiency in Sec. 5. In Sec. 6, we discuss the results.



(a) Prototype Device

(b) Current Recording Device

Fig. 1 Recording Equipment of Dog's Investigation Activities

2 Related Works

Several studies have recorded the images that an animal is looking at, along with its trajectory [1, 2, 3, 4]. In the CAT project, the investigation activity of an SAR dog was recorded using a camera and GPS mounted on it [1, 2, 3]. In another study, a camera and GPS receiver were attached onto a cat and cat's-eye-view images and its motion were recorded [4]. We had the goal of recording the investigation activities of SAR dogs by using a camera, a GPS receiver, and an IMU sensor. We developed the recording devices by referring to these studies. Our goal was to obtain more accurate data for an SAR dog's velocity and trajectory than could be obtained from GPS data alone. Thus, we developed the velocity estimation method reported in this paper.

Several studies have proposed methods for human position estimation that use an IMU and/or GPS. Bebek et al. proposed a method for human position estimation that uses an IMU sensor and a pressure sensor [5]. They equipped the inner sole of a shoe with these sensors to measure the zero velocity point (ZVP). Using ZVP makes it possible to cancel the cumulative error caused by the double integration of the acceleration. ZVP is useful for canceling the cumulative error. Therefore, we considered applying ZVP to SAR dogs' position estimation. An IMU sensor and a pressure sensor could be attached to a dog's paw by using shoes. However, this was not convenient for SAR dogs, because they disliked having something on their paws. In addition, the handlers worried that the shoes would come off or get stuck when the SAR dog moved on rubble. Thus, we needed to consider another method.

Many methods have been proposed to measure the velocity of mobile robots. Odometry is the most popular method, which calculates the velocity from the number of rotations of a wheel. Visual odometry can be used to calculate the velocity from the optical flow between different stereo camera images [6]. Scan matching calculates the velocity by matching several data scans measured using three-dimensional light detection and ranging [7]. We considered measuring the velocity $\mathbf{v} = (v_\mathrm{x}, v_\mathrm{y}, v_\mathrm{z})^\mathrm{T}$ using optical flow or scan matching. However, it was too difficult to apply these methods to SAR dogs because these devices are not lightweight and not small enough to be carried by an SAR dog. In addition, the dog's motion would cause blurring and distortion of the image and the scan data.

DGPS (Differential GPS) and RTK-GPS (Real Time Kinematic GPS) could be used to directly and accurately to measure the position of a dog. However, these GPS devices are too large and heavy to attach to an SAR dog. Therefore, we needed to develop another method.

Our goal then was to develop a method for estimating a dog's velocity using the features of its gait. We analyzed a dog's gaits and found their characteristics. Using these characteristics, we estimated the dog's velocity from its speed and pose.

3 Analysis of Dog's Gait and Its Velocity

3.1 Hypothesis about the Dog's Velocity

We analyzed a dog's gait and its velocity to establish a new principle for estimating this velocity. Several studies have analyzed the relationship between a dog's gait and its velocity. Maes described the relationship between a dog's body size and its velocity (speed) at each gait [8]. The average velocity can be estimated when the dog's gait and body size are obtained. That study suggested that the gait affects the velocity. However, these data are insufficient for estimating a dog's velocity vector at a certain moment. Therefore, we needed to analyze the relationship between a dog's gait and its velocity.

It is difficult to directly measure the velocity vector $\mathbf{v} = (v_\mathrm{x}, v_\mathrm{y}, v_\mathrm{z})^\mathrm{T}$ (see Sec. 2). Its scalar value $|\mathbf{v}|$ can be measured by using Doppler-effect-based speed sensors or optical flow sensors. A dog's pose can be measured using an IMU sensor. We wanted to find a method to calculate the velocity $\mathbf{v}$ from $|\mathbf{v}|$ and the dog's pose.

One assumption that we considered was that during each gait, there is a certain moment when the dog's velocity direction is equal to its heading direction. Actually, when a dog moves ahead, its heading direction faces the moving direction.

To evaluate the above assumption, we defined the dog's velocities as shown in Figure 2. Here, Σ_G is the world coordinate, whose x-y plane is parallel to the ground and whose z-axis is perpendicular to the x-y plane. $\mathbf{x} = (x, y, z, \theta_\mathrm{roll}, \theta_\mathrm{pitch}, \theta_\mathrm{yaw})^\mathrm{T}$ is the dog's position and pose on Σ_G. $\mathbf{v}$ is the dog's velocity on Σ_G. Σ_D is the dog's coordinate, where the origin is equal to the center of the dog's shoulder; the x axis is equal to the heading direction; the y axis is equal to the shoulder; and the z axis is perpendicular to the x-y plane. $\mathbf{n}$ is a unit vector of the x axis on Σ_D. $\mathbf{v}^\mathrm{D}$ is the dog's velocity on Σ_D. $\mathbf{R}$ and $\mathbf{t}$ are rotation and translation matrices that are used to convert from Σ_G to Σ_D. θ is the angle between $\mathbf{v}$ and $\mathbf{n}$. If θ is equal to 0, we can calculate $\mathbf{v}$ from $|\mathbf{v}^\mathrm{D}|$ and $\mathbf{R}$. Therefore, we analyze the relationship between dog's gait and its velocity by looking at changes in θ.

Σ_D's coordinate center and the heading direction are calculated from the positions of motion capture marker. Concrete definition of Σ_D is described at Sec. 3.2.

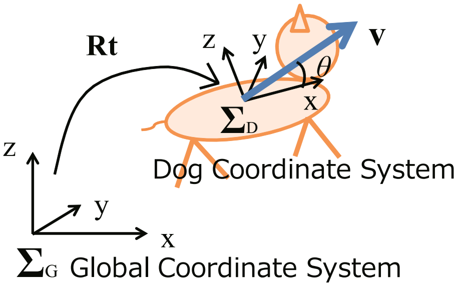


Fig. 2 Definition of Coordinate Systems (Σ_G, Σ_D) and Dog's Velocity on Each Coordinate System ($\mathbf{v}$, $\mathbf{v}^\mathrm{D}$)

3.2 Dog's Gait Measurement Using Motion Capture System

A Labrador was used for the measurement because it is a breed commonly adopted for SAR purposes. The dog's gaits (walk and trot) were measured using a motion capture system (eight cameras). Figure 3 shows the experimental setup for the gait measurement, along with the dog's trajectories while walking and trotting. Two cones were placed on the ground at a distance of 4 m. While walking, the dog walked three times around these cones. During trotting, the dog ran three times between these cones. The solid line shows the outward trip, and the dotted line shows the return trip. The outward trip data are almost the same as the return trip data. We explain our observations by using the outward trip data.

Figure 4 shows the layout of the motion capture markers. The numbers in Figure 4 show the IDs of the markers. A total of 14 markers were used. IDs 10, 11, 12, 13, and 14 were located on the other side of the dog's body. The center of Σ_D is located at ID 2. The x axis of Σ_D is parallel to the vector from ID 3 to ID 2. The y axis of Σ_D is parallel to the vector from ID 5 to ID 10 (which is located at the shoulder on the other side). $\mathbf{v}^D$ is the velocity of ID 2 on Σ_D.

Figure 5 shows each component of $\mathbf{v}^D$ during walk and trot. The walk data and trot data are analyzed in Sec. 3.3 and Sec. 3.4, respectively.

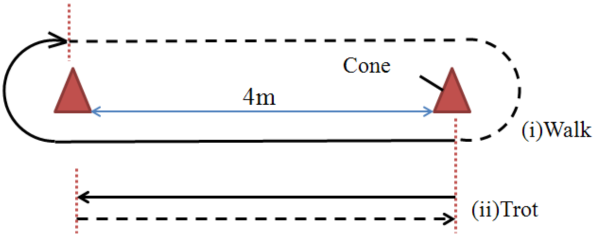


Fig. 3 Experimental Setup for Gait Measurement

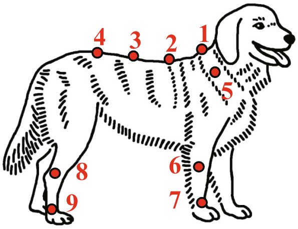
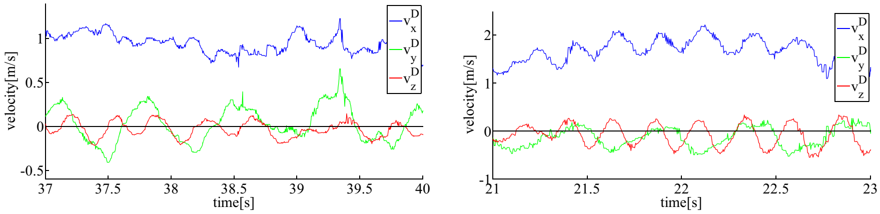


Fig. 4 Positions of Motion Capture Markers



(a) During Walk Gait (b) During Trot Gait

Fig. 5 Graph of Dog's Velocity Components (v_x^D, v_y^D, v_z^D)

3.3 Analysis of Walk Gait

Figure 5(a) shows the components of $\mathbf{v}^D$ at walk. The graph shows that there are cyclic moments where v_y^D and v_z^D are almost equal to 0 m/s. At these moments, the dog's velocity $\mathbf{v}^D$ almost faces the x axis of Σ_D. Then, we analyzed the angle between $\mathbf{v}$ and $\mathbf{v}^D$ using θ. Figure 6 shows the graph of θ. This graph shows that there are cyclic moments where θ is almost equal to 0°. Although there were offsets, θ was close to 0° at these moments (the average offset was 6.2° (standard deviation $\sigma_\theta = 2.2°$)).

We considered reasons why the offset occurred in the same direction. One possible reason is the SAR dog's vest. The markers were placed on a hard vest. The angle offset might have occurred during the gait because this hard vest affected the markers.

On the basis of these facts, we consider that the dog's velocity $\mathbf{v}$ can be reconstructed from the dog's speed $|\mathbf{v}|$ or $|\mathbf{v}^D|$ and the dog's pose at the above moments, when the dog is walking. This fact is important for online velocity estimation because heavy and large sensor devices are not required for the estimation.

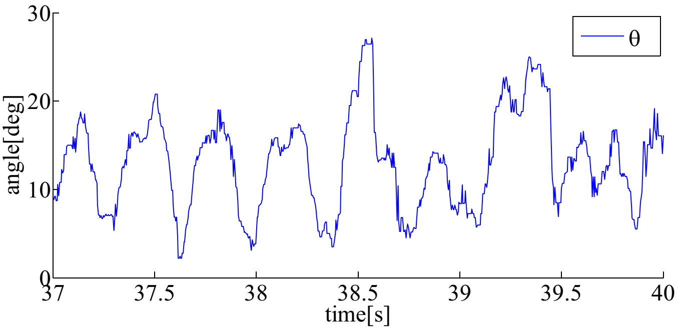


Fig. 6 Graph of Angle θ during its Walk Gait

We desired to detect the moments when θ was nearly equal to 0° (v_y^D and v_z^D were nearly equal to 0 m/s). We observed the relationships between the dog's four paws and θ. The velocities of the dog's four paws were used for the observation. Figure 7 shows graphs of these velocities and θ. During the walk, θ becomes approximately 0° at several moments (t = 37.6, 38.0, 38.4, and 38.8 s in Figure 7). From Figure 7, we confirmed that the dog's rear paws were just leaving the ground at these moments. The graph is slightly difficult to observe. Therefore, Figure 8 shows graphs of the velocities of the dog's rear paws and θ. When θ becomes approximately 0°, the velocities of the dog's rear paws are increasing rapidly.

We evaluated the accuracy of the detection on the basis of the motion of the dog's rear paws. In this evaluation, those moments ware detected st which the dog's rear paws left the ground, from motion capture data. The average error was 11.7°, and its standard deviation was 6.82°. Although the error was not very small, this result suggests that we can detect the moments when θ becomes 0° by using the motion of the dog's rear paws.

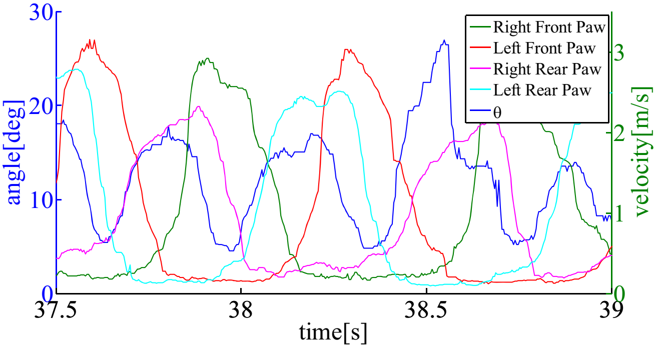


Fig. 7 Graph of Dog's Paw Velocities and Angle θ during Walk

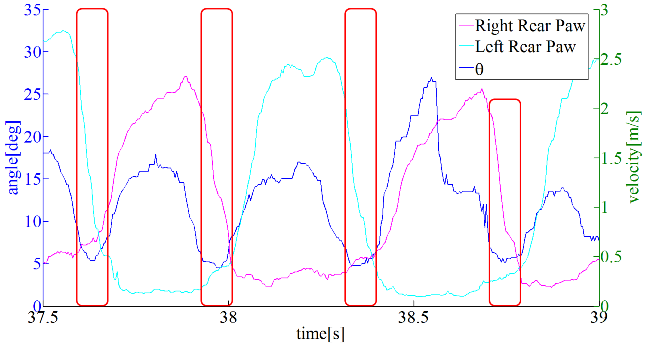


Fig. 8 Graph of Dog's Rear Paw Velocities and Angle θ during Walk

3.4 Analysis of Trot Gait

Figure 5(b) shows a graph of each component of $\mathbf{v}^{\mathrm{D}}$. The evaluation results for the trot gait are similar to those for the walk gait. $v_{\mathrm{y}}^{\mathrm{D}}$ and $v_{\mathrm{z}}^{\mathrm{D}}$ periodically become 0 m/s at the same time. Figure 9 shows a graph of θ. This graph shows that there are cyclic moments where θ is almost equal to 0°, just as with the walk gait. The offsets of these moments were 5.1° ($\sigma_\theta = 2.1°$). The average error for the trot gait is smaller than that for the walk gait. This is beacause the variation in $v_{\mathrm{y}}^{\mathrm{D}}$ was small during the trot gait.

We wanted to detect the moments when θ was nearly equal to 0°($v_{\mathrm{y}}^{\mathrm{D}}$ and $v_{\mathrm{z}}^{\mathrm{D}}$ were both nearly equal to 0 m/s) for the trot gait. Figure 10 shows the relationships between the velocity of the dog's four paws and θ. During the trot gait, θ becomes approximately 0° (t = 21.5, 21.8, 22, and 22.3 s in Figure 10) at the moments when the dog's rear paws just leave the ground. Figure 11 shows graphs of the velocities of the dog's rear paws and θ. When θ becomes approximately 0°, the velocities of the dog's rear paws are approximately 0 m/s.

We evaluated the accuracy of the detection on the basis of the motion of the dog's rear paws for the trot gait. In this evaluation, those moments were detected at which the dog's rear paws left the ground from motion capture data. The average error was 9.2°, and its stadard deviation was 4.6°. These results suggest that we can detect the moments when θ is close to 0° by using the motion of the dog's rear paws.

3.5 Consideration

We analyzed the motion capture data measured with a Labrador and found that there were cyclic moments when $v_{\mathrm{y}}^{\mathrm{D}}$ and $v_{\mathrm{z}}^{\mathrm{D}}$ became 0 m/s simultaneously for the walk and trot gaits. At these moments, the dog's velocity vector faced its heading direction. Therefore, the dog's velocity vector could be reconstructed from the dog's speed $|\mathbf{v}|$ and the pose at these moments.

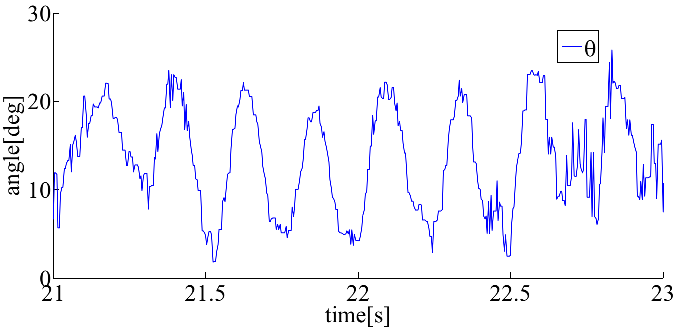


Fig. 9 Graph of Angle θ during its Trot Gait

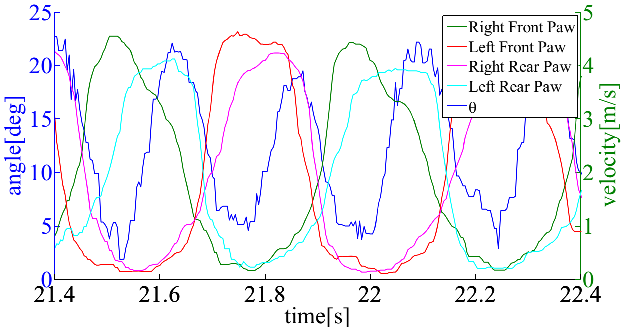


Fig. 10 Graph of Dog's Paw Velocities and Angle θ during Trot

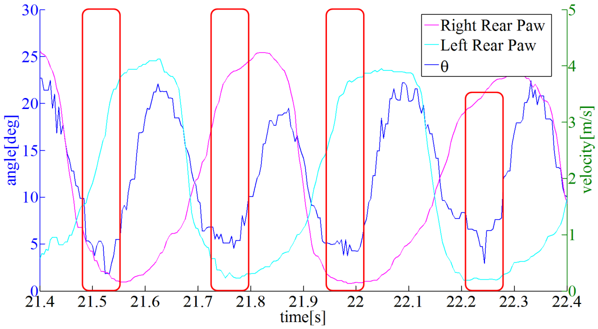


Fig. 11 Graph of Dog's Rear Paw Velocities and Angle θ during Trot

The walk and trot gait data show that at these moments, the dog's rear paws just leave the ground. When the dog's rear paws kick the ground, the dog's acceleration becomes reaches its maximum. Accelerating in the moving direction can lead to an efficient walk. Therefore, θ becomes close to 0°. This suggests that these moments can be detected by using the motion of the dog's rear paws.

For application of the velocity estimation method in engineering, it was necessary to evaluate its accuracy. In Sec. 4 and Sec. 5, we describe the gait-based velocity estimation method and an evaluation of its accuracy, respectively.

4 Gait-Based Velocity Estimation

We found that there were cyclic moments when the dog's velocity vector faced its heading direction. The dog's velocity consists of three components, $\mathbf{v}^{\mathrm{D}} = (\mathbf{v}_{\mathrm{x}}^{\mathrm{D}}, \mathbf{v}_{\mathrm{y}}^{\mathrm{D}}, \mathbf{v}_{\mathrm{z}}^{\mathrm{D}})^{\mathrm{T}}$. However, at these moments, the dog's velocity is equal to $(|\mathbf{v}^{\mathrm{D}}|, 0, 0)^{\mathrm{T}}$. Therefore, the dog's velocity vector $\mathbf{v}$ can be reconstructed from the dog's speed $|\mathbf{v}^{\mathrm{D}}|$ and its pose $\mathbf{R}$ ($\mathbf{v} = \mathbf{R}^{-1}(|\mathbf{v}^{\mathrm{D}}|, 0, 0)^{\mathrm{T}}$). We use this calculated velocity vector $\mathbf{v}$ to modify the cumulative errors.

We propose a velocity estimation method based on the extended Kalman filter (EKF). The velocity and its error are estimated from IMU sensor data using the following equations:

$$\begin{aligned}\overline{\mathbf{v}}_t &= \mathbf{v}_{t-1} + \mathbf{R}_{t-1}^{-1}\boldsymbol{a}_t\,dt\\ \overline{\boldsymbol{\Sigma}}_{vt} &= \boldsymbol{\Sigma}_{vt-1} + \mathbf{J}_a\boldsymbol{\Sigma}_a\mathbf{J}_a^{\mathrm{T}} + \boldsymbol{\Sigma}_{\mathrm{N}}\end{aligned} \tag{1}$$

The cumulative error of the velocity estimation is modified at these cyclic moments using the following equations:

$$\begin{aligned}\mathbf{v}_t &= \overline{\mathbf{v}}_t + \mathbf{K}_{vt}(\mathbf{z}_{vt} - \overline{\mathbf{v}}_t)\\ \boldsymbol{\Sigma}_{vt} &= (\mathbf{I} - \mathbf{K}_{vt})\overline{\boldsymbol{\Sigma}}_{vt}\\ \mathbf{K}_{vt} &= \overline{\boldsymbol{\Sigma}}_{vt}(\overline{\boldsymbol{\Sigma}}_{vt} + \mathbf{Q}_v)^{-1}\\ \mathbf{z}_v(t) &= \mathbf{R}_{t-1}^{-1}\begin{bmatrix} v_{\mathrm{measure}} \\ 0 \\ 0 \end{bmatrix}\end{aligned} \tag{2}$$

Here, $\mathbf{v}_t$ is the dog's velocity on Σ_{G}. $\boldsymbol{\Sigma}_{vt}$ is the covariance matrix, and $\boldsymbol{a}_t$ is the dog's acceleration. $\mathbf{R}_t$ is the rotation matrix that is used to convert Σ_{G} to Σ_{D}. $\mathbf{J}_a$ is a Jacobian of $\overline{\mathbf{v}}_t$ about $\boldsymbol{a}$. $\boldsymbol{\Sigma}_a$, $\mathbf{Q}_v$, and $\boldsymbol{\Sigma}_{\mathrm{N}}$ are the covariance matrices of the acceleration, observed velocity, and system noise, respectively. v_{measure} is the measured speed $|\mathbf{v}^{\mathrm{D}}|$.

The cycle of the velocity modification depends on the dog's gait. We detect the cyclic moments ($\theta = 0°$) on the basis of the motion of the dog's rear paw smotion in each gait. During the walk gait, the modification cycle is approximately 3 Hz, whereas in the trot gait, the cycle is approximately 4 Hz.

5 Evaluation of Velocity Estimation Method Based on Dog's Gait

5.1 Evaluation Method

We evaluated the accuracy of our proposed estimation method for a dog's velocity. In this evaluation, we used the IMU data and motion capture data during walk, which were described in Sec. 3.2. The acceleration $\boldsymbol{a}$ and pose $\mathbf{R}$ were obtained from the IMU sensor attached to the dog. The speed $|\mathbf{v}|$ was obtained from the motion capture data. We detected the moments when the dog's rear paws left the ground using the motion capture data.

We compared the accuracy of the gait-based velocity estimation with that of non-gait-based velocity estimation. In the gait-based velocity estimation, the velocity $\mathbf{v}$ is modified using Eq. (2) at the moments when the dog's rear paws leave the ground. In the non-gait-based velocity estimation, the velocity $\mathbf{v}$ is modified using Eq. (2) at the moments when the sensor data are obtained. In the latter case, the velocity direction does not face the dog's heading direction.

We consider that the accuracy changes depending on the frequency of sensor data acquisition. Therefore, we evaluated the proposed method using different frequencies for sensor data acquisition (1 Hz, 5 Hz, and 10 Hz). These frequencies were selected on the basis of the frequencies of commercial GPS sensors. In the gait-based velocity estimation, the speed was interpolated linearly.

5.2 Evaluation of Velocity Estimation During Dog'S Walk

Figure 12 shows the estimation results when using the dog's gait, along with the raw speed data. Figure 13 shows the estimation result without any modification. The cumulative error in Figure 12 is smaller than that in Figure 13. We confirmed that the cumulative error caused by the integration of the acceleration was canceled by using the speed data.

Figure 14 shows the error of the velocity estimation at each frequency. The error of the gait-based estimation is smaller than that of the non-gait-based estimation at each frequency. When the frequency was 10 Hz, the average errors in $v_{\mathrm{y}}^{\mathrm{D}}$ and $v_{\mathrm{z}}^{\mathrm{D}}$ were 0.11 m/s and 0.06 m/s for the gait-based estimation, and 0.14 m/s and 0.09 m/s for the non-gait-based estimation, respectively.

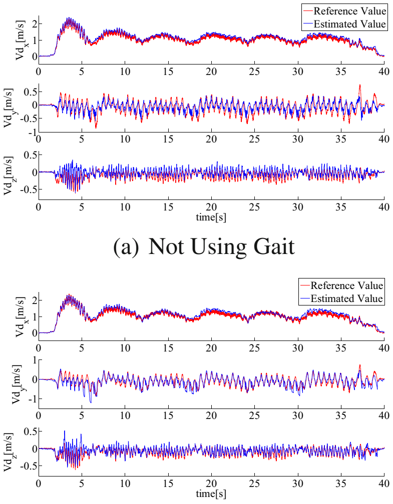
When the frequency was 1 Hz, the average estimation error decreased for the gait-based method in comparison to that for the non-gait-based method. $v_{\mathrm{x}}^{\mathrm{D}}$ decreased by 0.04 m/s; $v_{\mathrm{y}}^{\mathrm{D}}$ decreased by 0.22 m/s; and $v_{\mathrm{z}}^{\mathrm{D}}$ decreased by 0.03 m/s. At the low frequency, the proposed method performed accurate velocity estimation.

5.3 Evaluation of Velocity Estimation at Dog's Trot

Figure 15 shows the estimation results when using the dog's gait, along with the raw speed data. Figure 16 shows the error of the velocity estimation at each frequency. In the case of the trot data, accurate velocity ($v_{\mathrm{x}}^{\mathrm{D}}$, $v_{\mathrm{y}}^{\mathrm{D}}$, $v_{\mathrm{z}}^{\mathrm{D}}$) could be estimated by using the proposed method. In particular, the error of $v_{\mathrm{y}}^{\mathrm{D}}$ and $v_{\mathrm{z}}^{\mathrm{D}}$ became small when using the proposed method (Figures 16(b) and (c)). The errors in $v_{\mathrm{x}}^{\mathrm{D}}$ at 5Hz and 10Hz were almost the same for the gait-based approach and non-gait-based approach.

When the frequency was 10 Hz, the average errors in $v_{\mathrm{y}}^{\mathrm{D}}$ and $v_{\mathrm{z}}^{\mathrm{D}}$ were 0.12 m/s and 0.24 m/s for the gait-based estimation, and 0.13 m/s and 0.37 m/s for the non-gait-based estimation, respectively.

When the frequency was 1 Hz, the average estimation error decreased for the gait-based method in comparison to that for the non-gait-based method, as was observed in the case of walk gait. $v_{\mathrm{x}}^{\mathrm{D}}$ decreased by 0.15 m/s; $v_{\mathrm{y}}^{\mathrm{D}}$ decreased by 0.09 m/s; and $v_{\mathrm{z}}^{\mathrm{D}}$ decreased by 0.16 m/s.



(a) Not Using Gait

(b) Using Gait

Fig. 12 Graph of Dog's Velocity Estimation using 10 Hz Speed Data in Walk Gait: (a) Velocity Modified using 10 Hz Speed Data. (b) Velocity Modified Based on Gait.

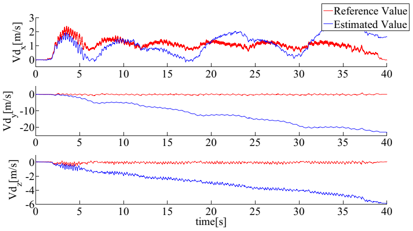
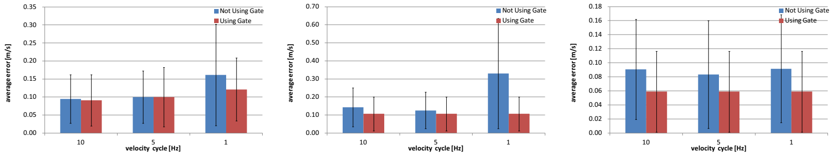


Fig. 13 Graph of Velocity Estimation without Any Modification in Walk

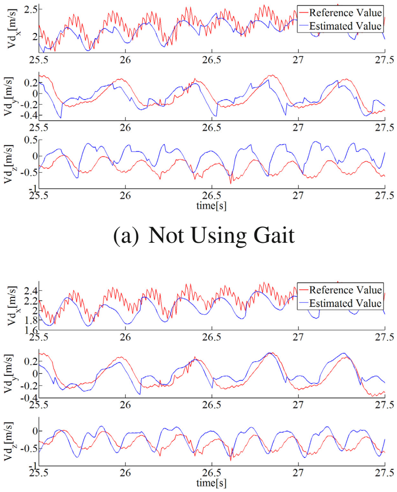


(a) $\mathbf{v}_x^D$

(b) $\mathbf{v}_y^D$

(c) $\mathbf{v}_z^D$

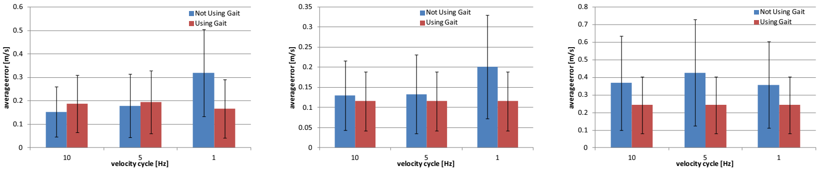
Fig. 14 Error of Velocity Estimation: Gait-based Velocity Estimation vs. Non-gait-based Velocity Estimation during Walk Gait



(a) Not Using Gait

(b) Using Gait

Fig. 15 Graph of Dog's Velocity Estimation using 10 Hz Speed Data in Trot Gait: (a) Velocity Modified using 10 Hz Speed Data. (b) Velocity Modified Based on Gait.



(a) $\mathbf{v}_x^D$ (b) $\mathbf{v}_y^D$ (c) $\mathbf{v}_z^D$

Fig. 16 Error of Velocity Estimation: Gait-based Velocity Estimation vs. Non-gait-based Velocity Estimation during Trot Gait

6 Discussion

From the analysis of the dog's gaits, we found the cyclic moments when the dog's velocity faces its heading direction ($\theta \simeq 0°$). At these moments, the dog's velocity can be reconstructed from its speed and pose. During the walk and trot gaits, the cyclic moments were synchronized with the dog's rear paws. There were small offsets between the cyclic moments and the dog's rear paw motions. The offset for the walk gait was larger than that for the trot gait. For accurate velocity estimation, it is essential to estimate the offsets during the gaits. In addition, we believe that the hard vest caused θ to be unequal to 0°. In future work, we need to determine whether this was actually the reason.

We confirmed that the accuracy of the velocity estimation improved by using the gait-based velocity estimation. The proposed method worked well at a low frequency for sensor data acquisition (1 Hz). This result can pave the way for the development of embedded devices that can obtain the speed at 1 Hz (e.g., mobile phone), for estimating a dog's trajectory.

In this study, we evaluated the walk and trot gaits. In future work, we intend to evaluate not only the walk and trot gaits but also the gallop gait. Motion capture data were used to detect the moments when the dog's rear paws left the ground. With regard to the dog's wearable equipment, we need to develop a walk-detection method for a dog. Currently, we are trying to detect walk by using force sensors, microphones, or IMU sensors. The results of this work will be reported in the near future.

7 Conclusion

In this study, we analyzed a dog's gait data (walk and trot) and found the moments when the dog's velocity faced its heading direction during walk and trot. Using the above facts, we proposed a velocity estimation method based on the dog's gait. We confirmed that the use of gait permits the improvement of the accuracy of velocity estimation during the walk and trot gaits. This result will facilitate the acquisition of a dog's trajectory.

References

1. Ferworn, A., Sadeghian, A., Barnum, K., Rahnama, H., Pham, H., Erickson, C., Ostrom, D., Dell'Agnese, L.: Urban Search and Rescue with Canine Augmentation Technology. In: 2006 IEEE/SMC International Conference on System of Systems Engineering, Los Angeles, CA, USA (2006)
2. Tran, J., Ferworn, A., Ribeiro, C., Denko, M.: Enhancing Canine Disaster Search. In: IEEE SoSE 2008, Monterey, CA, USA, June 2-5 (2008)
3. Ferworn, A., Sadeghian, A., Barnum, K., Ostrom, D., Rahnama, H., Woungang, I.: Canine as Robot in Directed Search. In: 2006 IEEE/SMC International Conference on System of Systems Engineering, Los Angeles, CA, USA (2006)
4. Yonezawa, K., Miyaki, T., Rekimoto, J.: Cat@Log: Sensing Device Attachable to Pet Cats for Supporting Human-Pet Interaction. In: International Conference on Advances in Computer Entertainment Technology (ACE 2009), pp. 149–156 (2009)
5. Bebek, O., et al.: Personal Navigation via Shoe Mounted Inertial Measurement Units. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, pp. 1052–1058 (2010)
6. Nister, D., et al.: Visual odometry for ground vehicle applications. Journal of Field Robotics 23(1), 3–20 (2006)
7. Nüchter, A., Surmann, H., Lingemann, K., Hertzberg, J., Thrun, S.: 6D SLAM with an Application in Autonomous Mine Mapping. In: Proceedings of the IEEE International Conference on Robotics and Automation, New Orleans, USA, pp. 1998–2003 (April 2004)
8. Maes, L.D., Herbin, M., Hackert, R., Bels, V.L., Abourachid, A.: Steady locomotion in dogs:Temporal and associated spatialcoordination patterns and the effect of speed. The Journal of Experimental Biology 211, 138–149 (2008)

Author Index