Chapter Six

Basics of Digital Audio

Audio information is crucial for multimedia presentations and, in a sense, is the simplest type of multimedia data. However, some important differences between audio and image information cannot be ignored. For example, while it is customary and useful to occasionally drop a video frame from a video stream, to facilitate viewing speed, we simply cannot do the same with sound information or all sense will be lost from that dimension.

6.1 Digitization of Sound

What Is Sound?

Sound is a wave phenomenon like light, but it is macroscopic and involves molecules of air being compressed and expanded under the action of some physical device..For example,

a speaker in an audio system vibrates back and forth and produces a longitudinal pressure wave that we perceive as sound.

Without air there is no sound - for example, in space. Since sound is a pressure wave, it takes on continuous values, as opposed to digitized ones with a finite range. Nevertheless, if we wish to use a digital version of sound waves, we must form digitized representations of audio information.

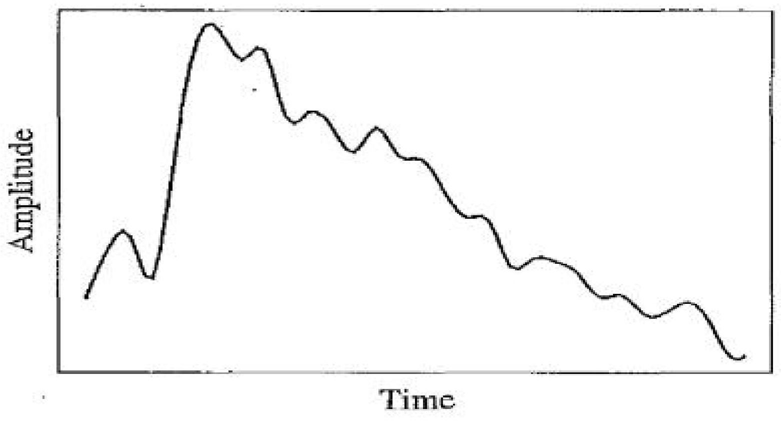

FIGURE : An analog signal: continuous measurement of pressure wave.

Figure shows the one-dimensional nature of sound. Values change over time in *amplitude:* the pressure increases or decreases with time. The amplitude value is a continuous quantity. Since we are interested in working with such data in computer storage, we must *digitize* the *analog signals* (i.e., continuous-valued voltages) produced by microphones.

For image data, we must likewise digitize the time-dependent analog signals produced by typical video-cameras. Digitization means conversion to a stream of numbers - preferably *integers* for efficiency

Since the graph in Figure is two-dimensional, to fully digitize the signal shown we have to *sample* in each dimension - in time and in amplitude. Sampling means measuring the quantity we are interested in, usually at evenly spaced intervals. The first kind of sampling - using measurements only at evenly spaced time intervals - is simply called *sampling* (surprisingly), and the rate at which it is performed is called the *sampling frequency.*

For audio, typical sampling rates are from 8 kHz (8,000 samples per second) to 48 kHz. The human ear can hear from about 20 Hz (a very deep rumble) to as much as 20 kHz; above this level, we enter the range of ultrasound. The human voice can reach approximately 4 kHz and we need to bound our sampling rate from below by at least double this frequency (see the discussion of the Nyquist sampling rate, below). Thus we arrive at the useful range about 8 to 40 or so kHz

Nyquist Theorem

Signals can be decomposed into a sum of sinusoids, if we are willing to use enough sinusoids. shows how weighted sinusoids can build up quite a complex signal. Whereas frequency is an absolute measure, pitch is a perceptual, subjective quality of sound - generally, pitch is relative.

Note that the true frequency and its alias are located symmetrically on the frequency axis with respect to the Nyquist frequency pertaining to the sampling rate used. For this reason, the Nyquist frequency associated with the sampling frequency is often called the "folding" frequency. That is to say, if the sampling frequency is less than twice the true frequency, and is greater than the true frequency, then the alias frequency equals the sampling frequency

Signal-to-Noise Ratio (SNR)

In any analog system, random fluctuations produce *noise* added to the signal, and the measured voltage is thus incorrect. The ratio of the power of the correct signal to the noise is called the *signal-fo-noise ratio (SNR).* Therefore, the SNR is a measure of the quality of the signal.

The SNR is usually measured in *decibels (dB),* where 1 dB is a tenth of a *bel.* The SNR value, in units of dB, is defined in tenns of base-l0 logarithms of squared voltages:

$$SNR = 10\log_{10}\frac{V_{signal}^2}{V_{noise}^2} = 20\log_{10}\frac{V_{signal}}{V_{noise}}$$

Signal-to-Quantization-Noise Ratio (SQNR)

For digital signals, we must take into account the fact that only quantized values are stored. For a digital audio signal, the precision of each sample is determined by the number of bits per sample, typically 8 or 16.

Aside from any noise that may have been present in the original analog signal, additional error results from quantization. That is, if voltages are in the range of 0 to 1 but we have only 8 bits in which to store values, we effectively force all continuous values of voltage into only 256 different values. Inevitably, this introduces a roundoff error. Although it is not really "noise," it is called *quantization noise* (or *quantization error).* The association with the concept of noise is that such errors will essentially occur randomly from sample to sample.

The quality of the quantization is characterized by the *signal-to-quantization-noise ratio (SQNR).* Quantization noise is defined as the difference between the value of the analog signal, for the particular sampling time, and the nearest quantization interval value. At most, this error can be as much as half of the interval

6.2 Quantization and Transmission of Audio

To be transmitted, sampled audio information must be digitized, and here we look at some of the details of this process. Once the information has been quantized, it can then be transmitted or stored.

Coding of Audio

Quantization and transformation of data are collectively known as *coding* of the data. For audio, the ч-law technique for companding audio signals is usually combined with a simple algorithm that exploits the temporal redundancy present in audio signals.

Differences in signals between the present and a previous time can effectively reduce the size of signal values and, most important, concentrate the histogram of pixel values (differences, now) into a much smaller range.

Pulse Code Modulation

PCM in General. Audio is analog - the waves we hear travel through the air to reach our eardrums. We know that the basic techniques for creating digital signals from analog ones consist of *sampling* and *quantization.* Sampling is invariably done uniformly - we select a sampling rate and produce one value for each sampling time.
In the magnitude direction, we digitize by quantization, selecting breakpoints in magnitude and remapping any value within an interval to one representative output level. The set of interval boundaries is sometimes called *decision boundaries,* and the representative values are called *reconstruction levels*

Every compression scheme has three stages:

- **Transformation**. The input data is *transformed* to a new representation that is easier or more efficient to compress. For example, in Predictive Coding, (discussed later in the chapter) we predict the next signal from previous ones and transmit the prediction error.

- **Loss**. we may introduce *loss* of information. Quantization is the main lossy step. Here we use a limited number of reconstruction levels, fewer than in the original signal. Therefore, quantization necessitates some loss of information.

- **Coding**. Here, we assign a *codeword* (thus forming a binary bitstream) to each output level or symbol. This could be a fixed-length code or a variable-length code, such as Huffman coding (discussed in Chapter 7).

Differential Coding of Audio

Audio is often stored not in simple PCM but in a form that exploits differences. For a start, differences will generally be smaller numbers and hence offer the possibility of using fewer bits to store.

An advantage of forming differences is that the histogram of a difference signal is usually considerably more peaked than the histogram for the original signal. For example, as an extreme case, the histogram for a linear ramp signal that has constant slope is uniform, whereas the histogram for the derivative of the signal (i.e., the differences, from sampling point to sampling point) consists of a spike at the slope value.

Generally, if a time-dependent signal has some consistency over time *(temporal redundancy),* the difference signal - subtracting the current sample from the previous one - will have a more peaked histogram, with a maximum around zero. Consequently, if we then go on to assign bitstring codewords to differences, we can assign short codes to prevalent values and long codewords to rarely occurring ones.

Lossless Predictive Coding

Predictive coding simply means transmitting differences - we predict the next sample as being equal to the current sample and send not the sample itself but the error involved in making this assumption. That is, if we predict that the next sample equals the previous one, then the error is just the difference between previous and next. Our prediction scheme could also be more complex.

However, we do note one problem. Suppose our integer sample values are in the range 0 .. 255. Then differences could be as much as -255 .. 255. So we have unfortunately increased our *dynamic range* (ratio of maximum to minimum) by a factor of two: we may well need more bits than we needed before to transmit some differences.

DPCM

Differential Pulse Code Modulation is exactly the same as Predictive Coding, except that it incorporates a quantizer step. Quantization is as in PCM and can be uniform or nonuniform. One scheme for analytically determining the best set of nonuniform quantizer steps is the *Lloyd-Max* quantizer, named for Stuart Lloyd and Joel Max, which is based on a least squares minimization of the error term.

DM

DM stands for *Delta Modulation,* a much-simplified version of DPCM often used as a quick analog-to-digital converter.

ADPCM

Adaptive DPCM takes the idea of adapting the coder to suit the input much further. Basically, two pieces make up a DPCM coder: the quantizer and the predictor. Above, in Adaptive DM,we adapted the quantizer step size to suit the input. In DPCM, we can *adaptively modify the quantizer,* by changing the step size as well as decision boundaries in a nonuniform quantizer.

We can carry this out in two ways: using the properties of the input signal *(called forward adaptive quantization),* or the properties of the quantized output. For if quantized errors become too large, we should change the nonuniform Lloyd-Max quantizer (this is called *backward adaptive quantization).*

MIDI: MUSICAL INSTRUMENT DIGITAL INTERFACE

Wave-table files provide an accurate rendering of real instrument sounds but are quite large. For simple music, we might be satisfied with PM synthesis versions of audio signals that could easily be generated by a sound card. A sound card is added to a PC expansion board and is capable of manipulating and outputting sounds through speakers connected to the board, recording sound input from a microphone connected to the computer, and manipulating sound stored on a disk. If we are willing to be satisfied with the sound card's defaults for many of the sounds we wish to include in a multimedia project, we can use a simple scripting language and hardware setup called MIDI.

MIDI Overview

MIDI, which dates from the early 1980s, is an acronym that stands for *Musical Instrument Digital Interface.* It forms a protocol adopted by the electronic music industry that enables computers, synthesizers, keyboards, and other musical devices to communicate with each other. A synthesizer produces synthetic music and is included on sound cards, using one of the two methods discussed above. The MIDI standard is supported by most synthesizers, so sounds created on one can be played and manipulated on another and sound reasonably close. Computers must have a special MIDI interface, but this is incorporated into most sound cards. The sound card must also have both DA and AD converters. MIDI is a scripting language - it codes "events" that stand for the production of certain sounds. Therefore, MIDI files are generally very small. For example, a MIDI event might include values for the pitch of a single note, its duration, and its volume.

Terminology. A *synthesizer* was, and still may be, a stand-alone sound generator that can vary pitch, loudness, and tone color. (The pitch is the musical note the instrument plays - a C, as opposed to a G, *say.)* It can also change additional music characteristics, such as attack and delay time. A good (musician's) synthesizer often has a microprocessor, keyboard, control panels, memory, and so on. However, inexpensive synthesizers are now included on PC sound cards. Units that generate sound are referred to as tone modules or sound modules. . A *sequencer* started off as a special hardware device for storing and editing a *sequence* of musical events, in the form of MIDI data. Now it is more often a software *music editor* on the computer.

A *MIDI keyboard* produces no sound, instead generating sequences of MIDI instructions called MIDI messages.

Hardware Aspects of MIDI

The MIDI hardware setup consists of a 31.25 kbps (kilobits per second) serial connection, with the 10-bit bytes including a 0 start and stop bit. Usually, MIDI-capable units are either input devices or output devices, not both.

Figure shows a traditional synthesizer. The modulation wheel adds vibrato. Pitch bend alters the frequency, much like pulling a guitar string over slightly. There are often other controls, such as foots pedals, sliders, and so on.

The physical MIDI ports consist of 5-pin connectors labeled IN and OUT and a third connector, THRU. This last data channel simply copies data entering the IN channel. MIDI communication is half-duplex. MIDI IN is the connector via which the device receives all MIDI data. MIDI OUT is the connector through which the device transmits all the MIDI data it generates itself. MIDI THRU is the connector by which the device echoes the data it receives from MIDI IN (and only that - all the data generated by the device itself is sent via MIDI OUT). These ports are on the sound card or interlace externally, either on a separate card on a PC expansion card slot or using a special interlace to a serial or parallel port