DATA COMPRESSION FOR NETWORKING

DATA COMPRESSION

VIDEO CODEC

TRANSFORM CODING

IMAGE CODING

IMAGE PROCESSING

SIGNAL REPRESENTATION

VIDEO SIGNAL PROCESSING

VIDEOTELEPHONY

DIGITAL TELEVISION

AUDIO CODING

SPEECH CODING

There has been an explosive growth of multimedia communication over networks during the past two decades. Video, audio, and other continuous media data, as well as additional discrete media such as graphics, are parts of integrated network applications. For these applications, the traditional media (e.g., text, images), as well as the continuous media (e.g., video, audio), must be processed. Such processing, referred to as "data coding", often yields better and more efficient representations of text, image, graphics, audio, and video signals. The uncompressed media data often require very high transmission bandwidth and considerable storage capacity. To provide feasible and cost-effective solutions for the current quality requirements, compressed text, image, graphics, audio, and video streams are transmitted over networks.

As shown in Refs. 1–6 there exist many data coding and compression techniques that are, in part, competitive, and, in part, complementary. Most of these techniques are already used in today's products, while other methods are still undergoing development or are only partly realized. Today and in the near future, the major coding schemes are linear predictive coding, layered coding, and transform coding. The most important compression techniques are entropy coding (e.g., run-length coding, Huffman coding, and arithmetic coding), source coding (e.g., vector quantization, subsampling, and interpolation), hybrid coding (e.g., JPEG, MPEG-1, MPEG-2, MPEG-4, H.261, H.263, and H.264/AVC/Mpeg-4 Part 10), and other proprietary developed coding techniques (e.g., Intel's Indeo, Microsoft's Windows Media Audio and Video, General Instrument's DigiCipher, IBM's Ultimotion Machine, and Apple's Quick Time, etc.).

The purpose of this article is to provide the reader with a basic understanding of the principles and techniques of data coding and compression. Various compression schemes are discussed for transforming audio, image and video signals into compressed digital representations for efficient transmission or storage. Before embarking on this venture, it is appropriate to first introduce and clarify the basic terminology and methods for signal coding and compression.

BASIC TERMINOLOGY AND METHODS FOR DATA CODING

The word signal originally referred to a continuous time and continuous amplitude waveform, called an analog signal. In a general sense, people now view a signal as a function of time, where time may be continuous or discrete, and where the amplitude or values of the function may be continuous or discrete, and may be scalar or vector-valued. Thus, a signal is meant to represent a sequence or a waveform whose value at any time is a real number or real vector. In many applications, a signal also refers to an image which has an amplitude that depends on two spatial coordinates, instead of one time variable; or it can also refer to a video (moving images), where the amplitude is a function of two spatial variables and a time variable. The word data is sometimes used as a synonym for signal, but more often it refers to a sequence of numbers or more generally, vectors. Thus, data can often be viewed as a discrete time signal. During recent years, however, the word data has been increasingly been associated in most literature with the discrete or digital case, that is, with discrete time and discrete amplitude, what is called a digital signal.

Physical sources of analog signals such as speech, audio, image, video, and all observable electrical waveforms are analog and continuous time in nature. The first step to convert analog signals to digital form is sampling. An analog continuously fluctuating waveform can usually be characterized completely from the knowledge of its amplitude values at a countable set of points in time so that, in effect, one can "throw away" the rest of the signal. One does not need to observe how it behaves in between any two isolated instances of observation. This is at the same time remarkable and intuitively obvious. It is remarkable that one can discard so much of the waveform and still be able to accurately recover the missing parts. The intuitive idea is that, if one samples periodically at regularly spaced intervals, and the signal does not fluctuate too quickly so that no unexpected wiggles can appear between two consecutive sampling instants, then one can expect to recover the complete waveform by a simple process of interpolation or smoothing, where a smooth curve is drawn that passed through the known amplitude values at the sampling in-

stants.

When watching a movie, one is actually seeing 24 still pictures flashed on the screen every second. (Actually, each picture is flashed twice.) The movie camera that produced these pictures was actually photographing a scene by taking one still picture every 1/24th of a second. Yet, one has the illusion of seeing continuous motion. In this case, the cinematic process works because the brain is somehow doing the interpolation. This is an example of sampling in action in daily life. For an electrical waveform, or any other one-dimensional signal, the samples can be carried as amplitudes on a periodic train of narrow pulses. Consider a scalar time function $x(t)$, which has a Fourier transform $X(f)$. Assume there is a finite upper limit on how fast $x(t)$ can wiggle around or vary with time. Specifically, assume that $X(f) = 0$ for $|f| \geq W$. Thus, the signal has a strictly low-pass spectrum with cutoff frequency W hertz (Hz). To sample this signal, one can periodically observe the amplitude at isolated time instants $t = kT$ for $k = \ldots, -2, -1, 0, 1, 2, \ldots$. The sample rate is $f_s = 1/T$ and T is the sampling period or sampling interval in seconds.

The idealized case of the sampling model is impulse sampling with a perfect ability to observe isolated amplitude values at the sampling instants kT. The effect of such a sampling model is seen as the process of multiplying the original signal $x(t)$ by a sampling function, $s(t)$, which is the periodic train of impulses $p(t)$ (e.g., Dirac delta functions for ideal case) given by

$$s(t) = T \sum_{k=-\infty}^{\infty} p(t - kT)$$

where the amplitude scale is normalized to T so that the average value of $s(t)$ is unity. In the time domain, the effect of this multiplication operation is to generate a new impulse train whose amplitudes are samples of the waveform $x(t)$. Thus

$$y(t) = x(t)s(t) = T \sum_{k=-\infty}^{\infty} x(t)p(t - kT) \approx T \sum_{k=-\infty}^{\infty} x(kT)p(t - kT)$$

Therefore, one now has a signal $y(t)$ which contains only the sample values of $x(t)$ and all values in between the sampling instants have been discarded. Figure 1 is an example of continuous signal waveform and its sampled waveform. The complete recovery of $x(t)$ from the sampled signal $y(t)$ can be achieved if the sampling process satisfies the following fundamental theorem:

Nyquist Sampling Theorem. A signal $x(t)$ that is bandlimited to W (Hz) can be exactly reconstructed from its samples $y(t)$ when it is periodically sampled at a rate $f_s = 2W$.

This minimum sampling frequency of $2W$ (Hz) is called the Nyquist frequency or Nyquist rate. If we violate the condition of the sampling theorem, that is, the sampling rate is less than twice of the maximum frequency component in the spectrum of the signal to be sampled, then the recovered signal will be the original signal plus an additional undesired waveform whose spectrum overlaps with the high-frequency components of the original signal. This

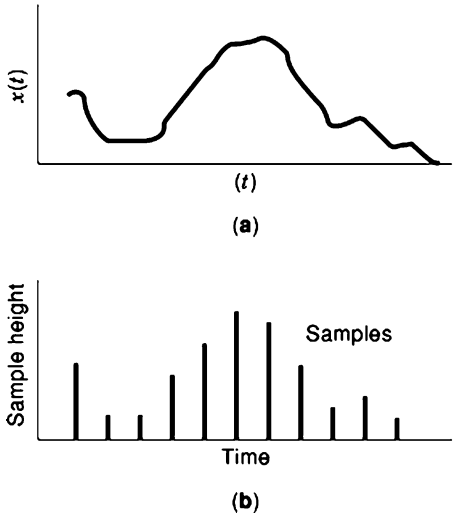

Figure 1. An example of sampling process, which shows (a) the original analog waveform and (b) its corresponding sampled waveform.

undesired component is called aliasing noise and the overall effect is referred to as aliasing, since the noise introduced here is actually a part of the signal itself but with its frequency components shifted to a new frequency.

The rate at which a signal is sampled usually determines the amount of processing, transmission or storage that will subsequently be required. Hence, it is desirable to use the lowest possible sampling rate that will satisfy a given application. On the other hand, most physical signals are not strictly bandlimited. However, typically, the contribution of the higher frequency signal components diminishes in importance as frequency increases over certain values. For example, music often does not have a well-defined cutoff frequency, below which significant power density exists, and above which no signal power is present. But human ears are not sensitive to very-high-frequency sound. So how does one choose a meaningful sampling rate that is not higher than necessary and yet does not violate the sampling theorem?

The answer is to first decide how much of the original signal spectrum is really needed to be retained. Analog low-pass filtering is then performed on the analog signal before sampling, so that the "needless" high-frequency components are suppressed. This analog prefiltering is often called antialias filtering. For example, in digital telephony, the standard antialias filter has a cutoff of 3.4 kHz, although the speech signal contains frequency components extending well beyond this frequency. This cutoff allows the moderate sampling rate of 8 kHz to be used and retains the voice fidelity that was already achieved with analog telephone circuits, which were already limited to roughly 3.4 kHz. In summary, analog prefiltering is needed to prevent aliasing of the signal and noise components that lie outside of the frequency band that must be preserved and reproduced.

Just as a waveform is sampled at discrete times, the value of the sampled waveform at a given time is also converted to a discrete value. Such a conversion process is

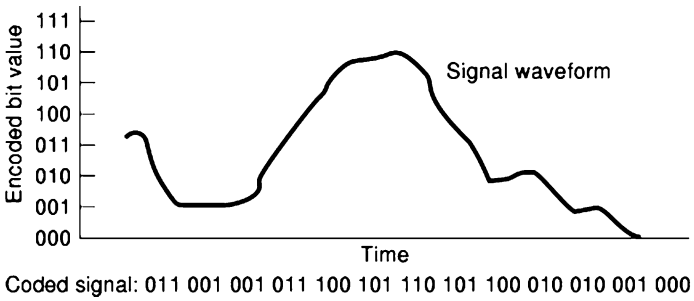

Figure 2. PCM coded signal of sampled waveform in Fig. 1(b).

called *quantization,* which will introduce loss on sampled waveform. The resolution of quantization depends on the number of bits used in measuring the height of the waveform. For example, an 8-bit quantization yields 256 possible values. Lower resolutions of quantization will result in higher losses of the digital signal. The electronic device that converts a signal waveform into digital samples is called an analog-to-digital converter (*ADC*). The reverse conversion is performed by a digital-to-analog converter (*DAC*).

The first process to sample analog signals and then quantize the sample values was *pulse code modulation* (*PCM*). PCM was invented in the 1930s, but only became prevalent in the 1960s, when transistors and integrated circuits became available. Figure 2 depicts the steps involved in PCM at a high level. PCM does not require sophisticated signal processing techniques and related circuitry. Hence, it was the first method to be employed, and is the prevalent method used today in telephone plants. PCM provides excellent quality. PCM is specified by the International Telephone and Telegraph Consultative Committee (*CCITT*). The current name of the specification is International Telecommunication Union (*ITU*) for voice coding in Recommendation G.711. A problem with PCM is that it requires a fairly high bandwidth (e.g., 64 kHz, for voice coding) to code a signal.

PCM has been around for a long time, and new technologies are beginning to demand attention. Of all the available schemes emerging from the laboratory, differential pulse code modulation (*DPCM*) and adaptive DPCM (*ADPCM*) schemes are among the most promising techniques. If a signal has a high correlation between adjacent samples, the variance of the difference between adjacent samples is smaller than the variance of the original signal. If this difference is coded, rather than the original signal, fewer bits are needed for the same desired accuracy. That is, it is sufficient to represent only the first PCM-coded sample as a whole, and all following samples as the difference from the previous one. This is the idea behind DPCM. In general, fewer bits are needed for DPCM than for PCM.

In a typical DPCM system, the input signal is bandlimited, and an estimate of the previous sample (or a prediction of the current signal value) is subtracted from the input. The difference is then sampled and coded. In the simplest case, the estimate of the previous sample is formed by taking the sum of the decoded values of all the past differences (which ideally differ from the previous sample only by a quantizing error). DPCM exhibits a significant improvement over PCM when the signal spectrum is peaked at the lower frequencies and rolls off toward the higher frequencies.

A modification of DPCM is delta modulation (*DM*). When coding the differences, it uses exactly one bit, which indicates whether the signal increases or decreases. This leads to an inaccurate coding of steep edges. This technique is particularly profitable if the coding does not depend on 8-bit grid units. If differences are small, a smaller number of bits are sufficient.

A prominent adaptive coding technique is ADPCM. It is a successive development of DPCM. Here, differences are encoded by the use of only a small number of bits (e.g., 4 bits). Therefore, either sharp transitions are coded correctly (these bits represent bits with a higher significance), or small changes are coded exactly (DPCM-encoded values are the less-significant bits). In the second case, a loss of high frequencies would occur. ADPCM adapts to this "significance" for a particular data stream as follows: the coder divides the value of DPCM samples by a suitable coefficient and the decoder multiplies the compressed data by the same coefficient, that is, the step size of the signal changes.

The value of the coefficient is adapted to the DPCM-encoded signal by the coder. In the case of a high-frequency signal, large DPCM coefficient values occur. The coder determines a high value for the coefficient. The result is a very coarse quantization of the DPCM signal in passages with steep edges. Low-frequency portions of such passages are hardly considered at all. For a signal with permanently relatively small DPCM values, the coder will determine a small coefficient. Thereby, a fine resolution of the dominant low-frequency signal portions is guaranteed. If high-frequency portions of the signal suddenly occur in such a passage, a signal distortion, in the form of a slope-overload, arises. Considering the actually defined step size, the greatest possible change by a use of the existing number of bits will not be large enough to represent the DPCM value with an ADPCM value. The transition of the PCM signal will be faded.

It is possible to explicitly change the coefficient that is adaptively adjusted to the data in the coding process. Alternatively, the decoder is able to calculate the coefficients itself from an ADPCM-encoded data stream. In ADPCM, the coder can be made to adapt to DPCM value change by increasing or decreasing the range represented by the encoded bits. In principle, the range of bits can be increased or decreased to match different situations. In practice, the ADPCM coding device accepts the PCM coded signal and then applies a special algorithm to reduce the 8-bit samples to 4-bit words using only 15 quantization levels. These 4-bits words no longer represent sample amplitudes; instead, they contain only enough information to reconstruct the amplitude at the distant end. The adaptive predictor predicts the value of the next signal on the level of the previously sampled signal. A feedback loop ensures that signal variations are followed with minimal deviation. The deviation of the predicted value, measured against the actual signal, tends to be small, and can be encoded with 4-bits.

FUNDAMENTAL COMPRESSION ALGORITHMS

The purpose of compression is to reduce the amount of data for multimedia communication. The amount of compression that an encoder achieves can be measured in two different ways. Sometimes the parameter of interest is compression ratio—the ratio between the original source data and the compressed data sizes. However, for continuous-tone images another measure, the average number of compressed bits/pixel, is sometimes a more useful parameter for judging the performance of an encoding system. For a given image, however, the two are simply different ways of expressing the same compression.

Compression in multimedia systems is subject to certain constraints. The quality of the coded and, later on, decoded data, should be as good as possible. To make a cost-effective implementation possible, the complexity of the technique should be minimal. The processing of the algorithm can not exceed certain time spans.

A natural measure of quality in a data coding and compression system is a quantitative measure of distortion. Among the quantitative measures, a class of criteria often used is the mean square criterion. It refers to some type of average or sum (or integral) of squares of the error between the sampled data $y(t)$ and decoded or decompressed data $y'(t)$. For data sequences $y(t)$ and $y'(t)$ of N samples, the quantity

$$\sigma_{ls}^2 = \frac{1}{N}\sum_{t=1}^{N} |y(t) - y'(t)|^2$$

is called the *average least squares error* (*ALSE*). The quantity

$$\sigma_{ms}^2 = E\left[|y(t) - y'(t)|^2\right]$$

is called the *mean square error* (*MSE*), where E represents the mathematical expectation. Often ALSE is used as an estimate of MSE. In many applications, the (mean square) error is expressed in terms of a *signal-to-noise ratio* (*SNR*), which is defined in decibels (*dB*) as

$$SNR = 10\log_{10}\frac{\sigma^2}{\sigma_e^2}, \qquad \sigma_e = \sigma_{ms}, \sigma_{ls}$$

where σ^2 is the variance of the original sampled data sequence.

Another definition of SNR, used commonly in image and video coding applications, is

$$PSNR = 10\log_{10}\frac{(\textit{peak-to-peak value of the reference image})^2}{\sigma_e^2}$$

The PSNR value is roughly 12 to 15 dB above the value of SNR.

Another commonly used method for measuring the performance of data coding and compression system is rate distortion theory. Rate distortion theory provides some useful results, which tell us the minimum number of bits required to encode the data, while admitting a certain level of distortion, and vice versa.

The rate distortion function of a random variable x gives the minimum average rate R_D (in bits per sample) required to represent (or code) it while allowing a fixed distortion D in its reproduced value. If x is a Gaussian random variable of variance σ^2, and y is its reproduced value and if the distortion is measured by the mean square value of the difference (x-y), that is, $D = E[(x-y)^2]$, then the rate distortion function of x is defined as

$$R_D = \max\left\{0, \left(\frac{1}{2}\right)\log_2\left(\frac{\sigma^2}{D}\right)\right\}$$

Data coding and compression systems are considered optimal if they maximize the amount of compression subject to an average or maximum distortion.

As shown in Table 1, compression techniques fit into different categories. For their use in multimedia systems, one can distinguish among *entropy, source,* and *hybrid coding.* Entropy coding is a lossless process, while source encoding is a lossy process. Most multimedia systems use hybrid techniques, which are a combination of the two coding techniques.

Entropy coding is used independently of the media's specific characteristics. Any input data sequence is considered to be a simple digital sequence and the semantics of the data is ignored. Entropy encoding reduces the size of the data sequence by focusing on the statistical characteristics of the encoded data series to allocate efficient codes, independent of the characteristics of the data. Entropy encoding is an example of lossless encoding as the decompression process regenerates the data completely.

The concept of entropy is derived from classical 19th century thermodynamics. The basic ideas of entropy coding are as follows: First, one defines the term *information* by using video signals as examples. Consider a video sequence in which each pixel takes on one of K values. If the spatial correlation have been removed from the video signal, the probability that a particular level i appears will be P_i, independent of the spatial position. When such a video signal is transmitted, the information I imparted to the receiver by knowing which of the K levels is the value of a particular pixel, is $-\log_2 P_i$ bits. This value, averaged over an image, is referred to as the average information of the image, or the *entropy.* The entropy can therefore be expressed as

$$H = -\sum_{i=0}^{K-1} P_i \log_2 P_i$$

The entropy is also extremely useful for measuring the performance of a coding system. In "stationary" systems—systems where the probabilities are fixed—it provides a fundamental lower bound, called the entropy limit, for the compression that can be achieved with a given alphabet of symbols.

Entropy encoding attempts to perform efficient code allocation (without increasing the entropy) for a signal. Run-length encoding, Huffman encoding, and arithmetic encoding are well-known entropy coding methods (7) for efficient code allocation, and are commonly used in actual encoders.

Run-length coding is the simplest entropy coding. Data streams often contain sequences of the same bytes or symbols. By replacing these repeated byte or symbol sequences with the number of occurrences, a substantial reduction of data can be achieved. This is called *run-length coding,* which is indicated by a special flag that does not occur in the data stream itself. For example, the data sequence: GISSSSSSSGIXXXXXX can be run-length coded as: GIS# 7GIX# 6, where # is the indicator flag. The character "S" occurs 7 consecutive times and is "compressed" to 3 characters "S# 7", as well as the character "X" occurs 6 consecutive times and is also "compressed" to 3 characters "X# 6". Run-length coding is a generalization of zero suppression, which assumes that just one symbol appears particularly often in sequences and the coding focuses on uninterrupted sequences, or *runs,* of zeros or ones to produce an efficient encoding.

Huffman coding is an optimal way of coding with integer-length code words. Huffman coding produces a "compact" code whose definition is for a particular set of symbols and probabilities, no other integer code can be found that will give better coding performance than this code. Consider the example given in Table 2. The entropy—the average ideal code length required to transmit the weather—is given by

$$H = (1/16) \times 4 + (1/16) \times 4 + (1/8) \times 3 + (3/4) \times 0.415 = 1.186 \text{ bits/symbol}$$

However, fractional-bit lengths are not allowed, so the lengths of the codes listed in the column to the right do not match the ideal information. Since an integer code always needs at least one bit, increasing the code for the symbol "00" to one bit seems logical.

The Huffman code assignment procedure is based on a coding "tree" structure. This tree is developed by a sequence of pairing operations, in which the two least probable symbols are joined at a "node" to form two "branches" of the tree. As the tree is constructed, each node at which two branches meet is treated as a single symbol with a combined probability that is the sum of the probabilities for all symbols combined at that node.

Figure 3 shows a Huffman code pairing sequence for the four-symbol case in Table 2. In Fig. 3 the four symbols are placed on the number line from 0 to 1, in order of increasing probability. The cumulative sum of the symbol probabilities is shown at the left. The two smallest probability intervals are paired, leaving three probability intervals of size 1/8, 1/8, and 3/4. We establish the next branch in the tree by again pairing the two smallest probability intervals, 1/8 and 1/8, leaving two probability intervals, 1/4 and 3/4. Finally, the tree is completed by pairing the 1/4 and 3/4 intervals. To create the code word for each symbol, one assigns a 0 and 1, respectively (the order is arbitrary), to each branch of the tree. Then concatenate the bits assigned to these branches, starting at the "root" (at the right of the tree) and the following the branches back to the "leaf" for each symbol (at the far left). Notice that each node in this tree requires a binary decision—a choice between the two possibilities—and, therefore, appends one bit to the code word.

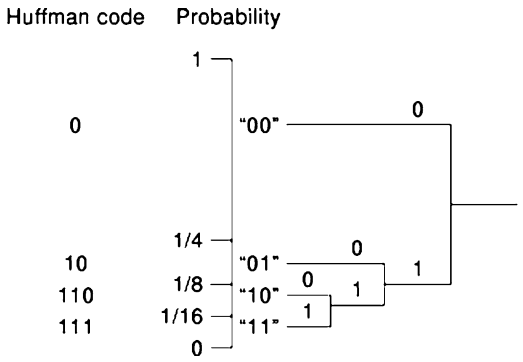

Figure 3. Huffman coding tree for the sequence symbols given in Table 2. It demonstrates the Huffman code assignment process.

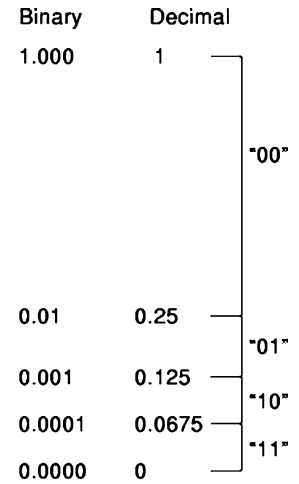

Figure 4. A process of partitioning the numbered line into subintervals for the arithmetic coding. It illustrates a possible ordering for the symbol probabilities in Table 2.

One of the problems with Huffman coding is that symbols with probabilities greater than 0.5 still require a code word of length one. This leads to less efficient coding, as can be seen for the codes in Table 2. The coding rate R achieved with Huffman codes in this case is as follows:

$$R = (1/16) \times 3 + (1/16) \times 3 + (1/8) \times 2 + (3/4) \times 1 = 1.375 \text{ bits/pixel}$$

This rate, when compared with the entropy limit of 1.186 bit/pixel, represents an efficiency of 86 percent.

Arithmetic coding is an optimal coding procedure that is not constrained to integer-length codes. In arithmetic coding, the symbols are ordered on the number line in the probability interval from 0 to 1 in a sequence that is known to both encoder and decoder. Each symbol is assigned a subinterval equal to its probability. Note that, since the symbol probabilities sum to one, the subintervals precisely fill the symbol probabilities in Table 2. Figure 4 illustrates a possible ordering for the symbol probabilities in Table 2.

The objective in arithmetic coding is to create a code stream that is a binary fraction pointing to the interval for the symbol being coded. Thus, if the symbol is "00", the code stream is a binary fraction greater than or equal to binary 0.01 (decimal 0.25), but less than binary 1.0. If the symbol

is "01", the code stream is greater than or equal to binary 0.001, but less than binary 0.01. If the symbol is "10", the code stream is greater than or equal to binary 0.0001, but less than binary 0.001. Finally, if the symbol is "11", the code stream is greater than or equal to binary 0, but less than 0.0001. If the code stream follows these rules, a decoder can see which subinterval is pointed to by the code stream and decode the appropriate symbol. Coding additional symbols is a matter of subdividing the probability interval into smaller and smaller subintervals, always in proportion to the probability of the particular symbol sequence. As long as one follows the rule—never allow the code stream to point outside the subinterval assigned to the sequence of symbols—the decoder will decode that sequence. For a detailed discussion of Huffman coding and arithmetic coding, interested readers should refer to (7).

Source coding takes into account the semantics of the data. The degree of compression that can be reached by source coding depends on the data contents. In the case of lossy compression techniques, a one-way relation between the original sequence and the encoded data stream exists; the data streams are similar but not identical. Different source coding techniques make extensive use of the characteristics of the specific medium. An example is sound source coding, where sound is transformed from time-dependent to frequency-dependent sound concatenations, followed by the encoding. This transformation, followed by encoding, substantially reduces the amount of data.

Predictive Coding. Prediction is the most fundamental aspect of source coding. The basis of predictive encoding is to reduce the number of bits used to represent information by taking advantage of correlation in the input signal. DPCM and ADPCM, discussed above, are among the simplest prediction coding methods. For digital video, signals exhibit correlation both between pixels within a frame (spatial correlation) and between pixels in differing frames (temporal correlation). Video compression techniques typically fall into two main types: (1) interframe prediction, which uses a combination of motion-prediction and interpolated frames to achieve high-compression ratio; (2) intraframe coding, which compresses every frame of video individually. Interframe prediction techniques take advantage of the temporal correlation, while the spatial correlation is exploited by intraframe coding methods. It is amenable also to utilize intra- and interfield prediction methods for interlaced video which scans alternate lines to distribute the pixels of a single frame across two fields.

Motion compensation (*MC*), one of the most complex prediction methods, reduces the prediction error, by predicting the motion of the imaged objects. The basic idea of MC arises from a commonsense observation: in a video sequence, successive frames (or fields) are likely to represent the same details, with little difference between one frame and the next. A sequence showing moving objects over a still background is a good example. Data compression can be effective if each component of a frame is represented by its difference with the most similar component—the predictor—in the previous frame, and by a vector—the motion vector—expressing the relative position of the two components. If an actual motion exists between the two frames, the difference may be null or very small. The original component can be reconstructed from the difference, the motion vector, and the previous frame.

A weakness of prediction-based encoding is that the influence of any errors during data transmission affects all subsequent data. In particular, when interframe prediction is used, the influence of transmission errors is quite noticeable. Since predictive encoding schemes are often used in combination with other schemes, such as transform-based schemes, the influence of transmission errors must be given due consideration.

Transform Coding. If we consider the frequency distribution of signals containing strong correlation, it appears that the signal power is concentrated in the low-frequency region. In general, it is possible to exploit for compression any systematic bias in components of the signal. The key idea behind transform coding is to transform the original signal in such a way as to emphasize the bias, making it more amenable to techniques that remove redundancy. One optimal transform is called the Karhunen–Loeve (KL) transformation. The KL transform can completely remove the statistical correlation of image data and provide a minimum mean-square-error (3). In application of the KL transform to images, there are dimensionality difficulties. The KL transform depends on the statistics as well as the size of the image. It is known that fast KL transform algorithms only exist for certain statistical image models. A number of orthogonal transforms, including the discrete Fourier transform (*DFT*) and the discrete cosine transform (*DCT*), have been used in various compression algorithms. Of these transforms, DCT is the most widely used for video compression, because the power of the transformed signal is well concentrated in the low frequencies, and it can be computed rapidly.

The following expresses a two-dimensional DCT for an $N \times N$ pixel block.

$$X(u,v) = \frac{4C(u)C(v)}{N^2} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} x(j,k) \cos\left(\frac{(2j+1)u\pi}{2N}\right) \cos\left(\frac{(2k+1)v\pi}{2N}\right)$$

$$x(j,k) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u)C(v)X(u,v) \cos\left(\frac{(2j+1)u\pi}{2N}\right) \cos\left(\frac{(2k+1)v\pi}{2N}\right)$$

where

$$C(w) = \begin{cases} \frac{1}{\sqrt{2}}, & w = 0 \\ 1, & w = 1, 2, \ldots, N-1 \end{cases}$$

After the transformation, DCT coefficients are quantized by levels specified in a quantization table. Usually, larger values of N improve the SNR, but the effect saturates above a certain block size. Further, increasing the block size increases the total computation cost required. The value of

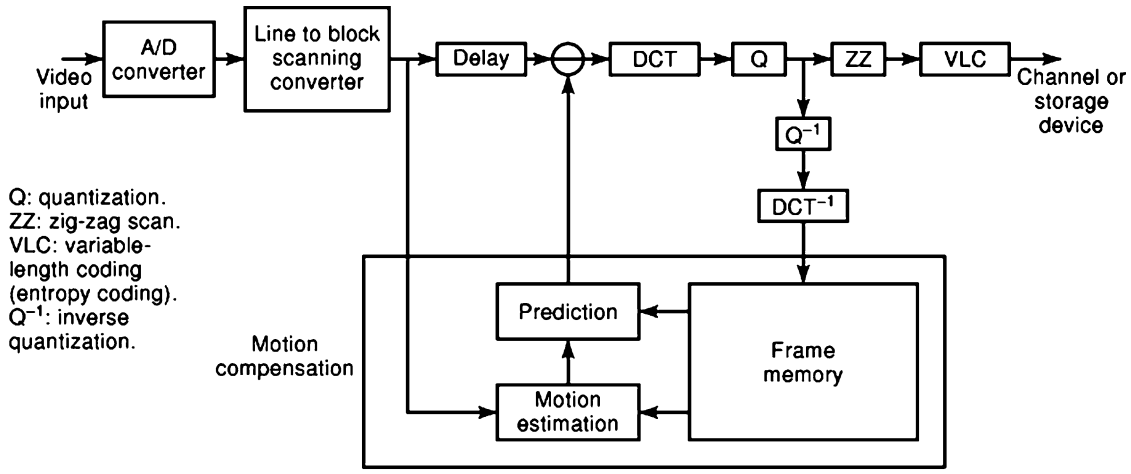

Figure 5. A block diagram of the MC + DCT coding scheme which shows the basic function blocks such as motion estimation, motion prediction, DCT variable length coding, and so on.

N is thus chosen to balance the efficiency of the transform and its computation cost, block sizes of 8 and 16 are common. For large quantization, DCT using block sizes of 8 and 16 often lead to "blocking artifacts"—visible discontinuities between adjacent blocks.

In practice, DCT is used in conjunction with other techniques, such as prediction and entropy coding. The *Motion Compensation Plus Discrete Cosine Transform* (*MC + DCT*) scheme, which will repeatedly be referred to, is a prime example of such a combination.

MC + DCT. Suppose that the video to be encoded consists of digital television or teleconferencing services. For this type of video, MC carried out on the basis of frame differences is quite effective. MC can be combined with the DCT for even more effective compression. The overall configuration of MC + DCT is illustrated in Fig. 5. The selection of block size compares its input signal with that of the previous frame (generally in units of 8×8 pixel blocks) and selects those that exhibit motion. MC operates by comparing the input signal in units of blocks against a locally decoded copy of the previous frame, extracting a motion vector, and using the motion vector to calculate the frame difference. The motion vector is extracted by, for example, shifting vertically or horizontally a region several pixels on a side and performing matching within the block or the macroblock (a 16×16 pixel segment in a frame) (8).

The motion-compensated frame-difference signal is then discrete cosine transformed, in order to remove spatial redundancy. A variety of compression techniques are applied in quantizing the DCT coefficients; the reader is directed to the references for details (8). A leading method is zig-zag scan, which has been standardized in JPEG, H.261, H.263, MPEG-1, -2, and -4, for video transmission encoding (8). Zig-zag scan, which transforms two-dimensional data into one dimension, is illustrated in Fig. 6. Because the dc component of the coefficients is of critical importance, ordinary linear quantization is employed for them. Other components are scanned, for example, in zig-zag fashion, from low to high frequency, linearly quantized, and variable-length-encoded by the use of run-length and Huffman coding.

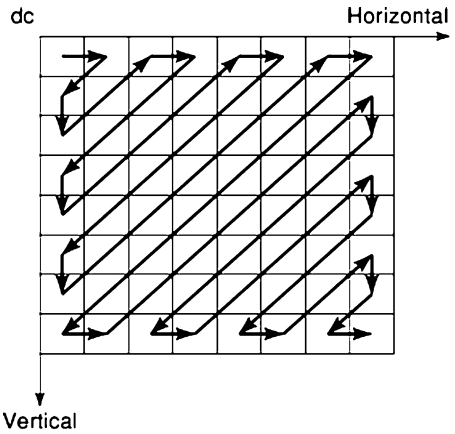

Figure 6. The zig-zag scan pattern for a 8×8 block.

Subband Coding. Subband coding (5) refers to the compression methods that divide the signal into multiple bands to take advantage of a bias in the frequency spectrum of the video signal. That is, efficient encoding is performed by partitioning the signal into multiple bands and taking into account the statistical characteristics and visual significance of each band.

The general form of a subband coding system is shown in Fig. 7. In the encoder, the analyzing filters partition the input signal into bands. Such a process is called subband decomposition. Each band is separately encoded, and the encoded bands are multiplexed and transmitted. The decoder reverses this process. Subband encoding does offer several advantages. Unlike DCT compression techniques, it is not prone to blocking artifacts. Furthermore, subband encoding is the most natural coding scheme when hierarchical processing is needed for video coding. The main technological features to be determined in subband encoding are the subband analysis method (2- or 3-dimensional), the structure of the analyzing filters, the bit allocation method, and the compression method within each band. In particular, there are quite a number of candidates for the form of the

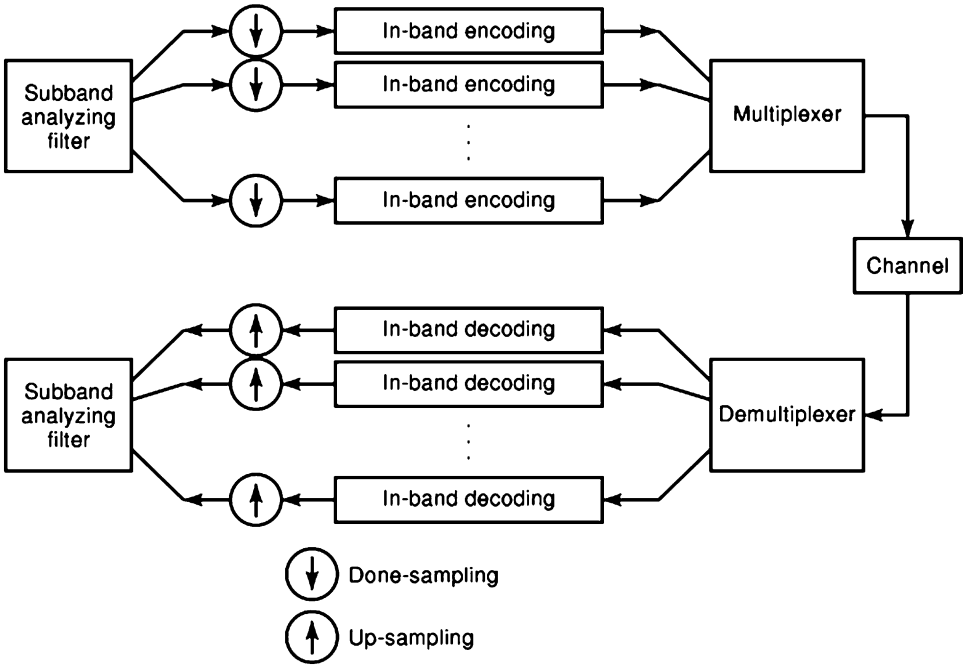

Figure 7. A simplified block diagram of subband coding scheme.

analysis and the structure of the filters. The filters must not introduce distortion due to aliasing in-band analysis and synthesis.

Figure 8 shows a two-band analysis and synthesis system. Consider the following analyzing filter as an example:

$$H_l(z) = \frac{1}{4}(1 + 2z^{-1} + z^{-2})$$
$$H_h(z) = \frac{1}{4}(-1 - 2z^{-1} + 6z^{-2} - 2z^{-3} - z^{-4})$$

For these analyzing filters, the characteristics of the synthesizing filters are

$$G_l(z) = H_h(-z) \qquad G_h(z) = H_l(-z)$$

The relationship between the input and output is then

$$Y(z) = \frac{1}{2}[H_l(z)X(z) + H_l(-z)X(-z)]G_l(z) - \frac{1}{2}[H_h(z)X(z) + H_h(-z)X(-z)]G_h(z) = z^{-3}X(z)$$

Clearly, the aliasing components completely cancel. The basic principles illustrated hold unchanged when two-dimensional filtering is used in a practical application.

Figure 9 illustrates how the two-dimensional frequency domain may be partitioned either uniformly or in an octave parent. If one recalls that signal power will be concentrated in the low-frequency components, then the octave method seems the most natural. Since this corresponds to constructing the analyzing filters in a tree structure, it lends itself well to implementation with filter banks.

In practical applications, one of the most important decomposition filters is what is called discrete wavelet transform (DWT). Wavelet theory provides a unified framework for multiresolution image compression. DWT-based compression enables coding of still image textures with a high coding efficiency as well as scaleable spatial resolutions at fine granularity.

The organization of a subband codec is similar to the DCT-based codec. The principal difference is that encoding and decoding are each broken out into a number of independent bands. Quality can be fixed at any desired value by adjusting the compression and quantization parameters of the encoders for each band. Entropy coding and predictive coding are often used in conjunction with subband coding to achieve high-compression performance.

If one considers quality from the point of view of the rate-distortion curve, then, at any given bit rate, the quality can be maximized by distributing the bits such that distortion is constant for all bands. A fixed number of bits is allocated, in advance, to each band's quantizer, based on the statistical characteristics of the band's signal. In contrast, adaptive bit distribution adjusts the bit count of each band according to the power of the signal. In this case, either the decoder of each subband must also determine the bit count for inverse quantization, using the same criterion as is used by the encoder, or the bit count information must be transmitted along with the quantized signal. Therefore, the method is somewhat lacking in robustness.

Vector Quantization. As opposed to scalar quantization, in which sample values are independently quantized one at a time, vector quantization (*VQ*) attempts to remove redundancy between sample values by collecting several sample values and quantizing them as a single vector. Since the input to a scalar quantizer consists of individual sample values, the signal space is a finite interval of the real number line. This interval is divided into several regions, and each region is represented in the quantized outputs by a

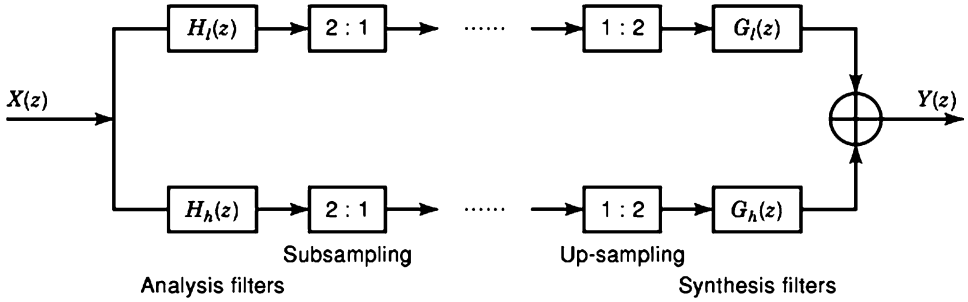

Figure 8. A two-band subband coding system. It demonstrates the in-band encoding and the in-band decoding blocks in Fig. 7.

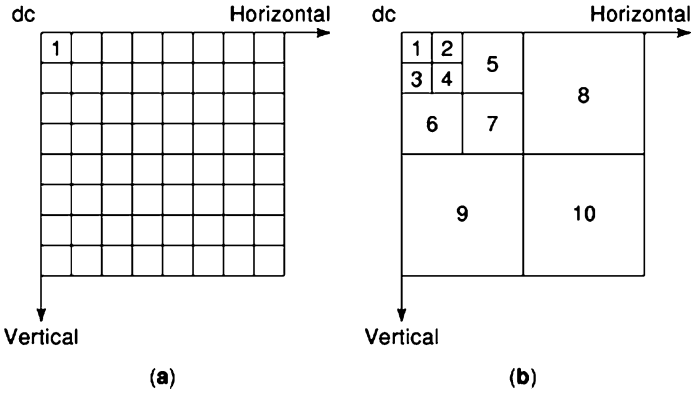

Figure 9. Subband splitting patterns in two-dimensional frequency domain: (a) uniform split (8×8); (b) octave split.

single value. The input to a vector quantizer is typically an n-dimensional vector, and the signal space is likewise an n-dimensional space. To simplify the discussion, consider only the case where $n = 2$. In this case, the input to the quantizer is the vector x_j, which corresponds to the pair of samples (s_j^1, s_j^2). To perform vector quantization, the signal space is divided into a finite number of nonoverlapping regions, and a single vector to represent each region is determined. When the vector $\boldsymbol{x}_j$ is input, the region containing $\boldsymbol{x}_j$ is determined, and the representative vector for that region, $\boldsymbol{y}_j$, is output. This concept is shown in Fig. 10. If we phrase the explanation explicitly in terms of encoding and decoding, the encoder determines the region to which the input $\boldsymbol{x}_j$ belongs and outputs j, the index value which represents the region. The decoder receives this value j, extracts the corresponding vector $\boldsymbol{y}_j$ from the representative vector set, and outputs it. The set of representative vectors is called the *codebook*.

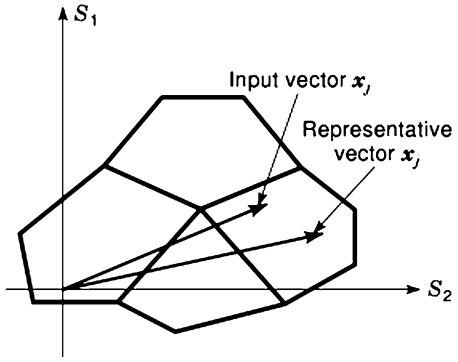

Figure 10. An example of VQ with two-dimensional vectors. To perform VQ, the signal space is divided into a finite number of nonoverlapping region, and a single vector is used to represent all vectors in each region.

The performance of vector quantization is evaluated in the same manner as for other schemes, that is, by the relationship between the encoding rate and the distortion. The encoding rate R per sample is given by the following equation

$$R = \lceil \log_2 N \rceil / K$$

where K is the vector dimensionality, and N is the number of quantization levels. The notation $\rceil \cdot \lceil$ represents the smallest integer greater than or equal to x (the "ceiling" of x).

We define the distortion as the distance between the input vector x_j and the output vector y_j. In video encoding, the square of the Euclidean distance is generally used as a distortion measure, because it makes analytic design of the vector quantizer for minimal distortion more tractable. However, it is not necessarily the case that subjective distortion perceived by a human observer coincides with the squared distortion.

To design a high-performance vector quantizer, the representative vectors and the regions they cover must be chosen to minimize total distortion. If the input vector probability density function is known in advance, and the vector dimensionality is low, it is possible to perform an exact optimization. However, in an actual application it is rare for the input vector probability density to be known in advance. The well-known LBG algorithm is widely used for adaptively designing vector quantizers in this situation (9). LBG is a practical algorithm that starts out with some rea-

sonable codebook and, by adaptively iterating the determination of regions and representative vectors, converges on a better codebook.

Figure 11 shows the basic structure of an image codec based on vector quantization. The image is partitioned into M-pixel blocks, which are presented, one at a time, to the VQ encoder as the Al-dimensional vector x_j. The encoder locates the closest representative vector in its prepared codebook and transmits the representative vector's index. The decoder, which need only perform a simple table lookup in the codebook to output the representative vector, is an extremely simple device. The simplicity of the decoder makes VQ coding very attractive for distribution-type video services. VQ coding, combining with other coding methods, has been adopted in many high-performance compression systems.

Table 1 shows examples of coding and compression techniques that are applicable in multimedia applications in relation to the entropy, source, and hybrid coding classification. Hybrid compression techniques are a combination of well-known algorithms and transformation techniques that can be applied to multimedia systems. For a better and clearer understanding of hybrid schemes to be identified in all schemes (entropy, source, and hybrid) a set of typical processing steps is described.

This typical sequence of operations has been shown in Fig. 5, which is performed in the compression of still images and video sequences. The following four steps describe the compression of one image:

1. Preparation includes analog-to-digital conversion and generating an appropriate digital representation of the information. An image is divided into blocks of 8×8 pixels, and represented by a fixed number of bits per pixel.
2. Processing is actually the first step of the compression process which makes use of sophisticated algorithms. A transformation from the time to the frequency domain can be performed by a use of DCT. In the case of motion video compression, interframe coding uses a motion vector for each 16×16 macroblock or 8×8 block.
3. Quantization processes the results of the previous step. It specifies the granularity of the mapping of real numbers into integers. This process results in a reduction of precision. In a transformed domain, the coefficients are distinguished according to their significance. For example, they could be quantized using a different number of bits per coefficient.
4. Entropy encoding is usually the last step. It compresses a sequential digital data stream without loss. For example, a sequence of zeros in a data stream can be compressed by specifying the number of occurrences followed by the zero itself.

In the case of vector quantization, a data stream is divided into blocks of n bytes each. A predefined table contains a set of patterns. For each block, a table entry with the most similar pattern is identified. Each pattern in the table is associated with an index. Such a table can be multidimensional; in this case, the index will be a vector. A decoder uses the same table to generate an approximation of the original data stream.

In the following sections the most relevant work in the standardization bodies concerning image and video coding is outlined. In the framework of International Standard Organization (*ISO/IEC/JTC1*), three subgroups were established in May 1988: the Joint Photographic Experts Group (*JPEG*) is working on coding algorithms for still images; the Joint Bilevel Image Experts Group (*JBIG*) is working on the progressive processing of bilevel coding algorithms, and the Moving Picture Experts Group (*MPEG*) is working on representation of motion video. In the International Telecommunication Union (*ITU*), H.261 and H.263 are also developed for video conference and telephone applications. The results of these standard activities are presented next.

JPEG

The ISO 10918-1 JPEG International Standard (1992) Recommendation T.81 is a standardization of compression and decompression of still natural images (4). JPEG provides the following important features:

- JPEG implementation is independent of image size.
- JPEG implementation is applicable to any image and pixel aspect ratio.
- Color representation is independent of the special implementation.
- JPEG is for natural images, but image content can be of any complexity, with any statistical characteristics.
- The encoding and decoding complexities of JPEG are balanced and can be implemented by a software solution.
- Sequential decoding (slice-by-slice) and progressive decoding (refinement of the whole image) should be possible. A lossless, hierarchical coding of the same image with different resolutions is supported.
- The user can select the quality of the reproduced image, the compression processing time, and the size of the compressed image by choosing appropriate individual parameters.

The key steps of the JPEG compression are DCT (8×8), quantization, zig-zag scan, and entropy coding. Both Huffman coding and arithmetic coding are options of entropy coding in JPEG. The JPEG decompression just reverses its compression process. A fast coding and decoding of still images also used for video sequences is known as Motion JPEG. Today, JPEG software packages, together with specific hardware support, are already available in many products.

ISO 11544 JBIG is specified for lossless compression of binary and limited bits/pixel images (4). The basic structure of the JBIG compression system is an adaptive binary arithmetic coder. The arithmetic coder defined for JBIG is identical to the arithmetic-coder option in JPEG.

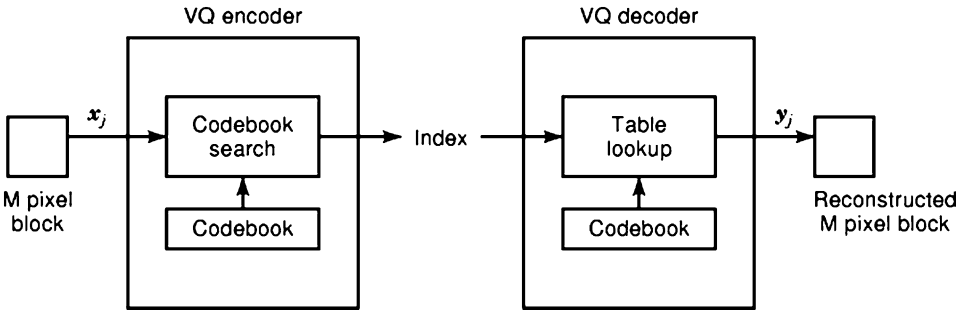

Figure 11. The basic structure of a VQ codec.

Table 1. A Classification of Coding/Compression Techniques

Entropy Coding	Source Coding		Hybrid Coding
Huffman Coding	Predictions	DPCM DM	JPEG
Arithmetic Coding	Transformation	FFT DCT	MPEG-1, MPEG-2, MPEG-4
Run-length Coding	Layered Coding	Bit Position Subsampling Sub-banding Coding	H.261 H.263
	Vector Quantization		Others

Table 2. Sequence Symbols with (1) Ideal Code Length, (2) Integer Code Length, and (3) Variable-Length (Huffman) Codes

Symbols	Probability (P)	Information (Ideal length $\log_2 P$)	Integer Code Length	Huffman Code
00	3/4	0.415 bits	1 bit	0
01	1/8	3 bits	2 bits	10
10	1/16	4 bits	3 bits	110
11	1/16	4 bits	3 bits	111

H.261 AND H.263

ITU Recommendations H.261 and H.263 (6) are digital video compression standards, developed for video conferencing and videophone applications, respectively.

Both H.261 and H.263 are developed for real-time encoding and decoding. For example, the maximum signal delay of both compression and decompression for H.261 is specified to be 150 ms for the end-to-end delay of targeted applications. Unlike JPEG, H.261 specifies a very precise image format. Two resolution formats each with an aspect ratio of 4:3 are specified. The so-called Common Intermediate Format (*CIF*) defines a luminance component (Y) of 288 lines, each with 352 pixels. The chrominance components (C_b and C_r) each have a resolution of 144 lines and 176 pixels per line to fulfill the 2:1:1 requirement. Quarter-CIF (*QCIF*) has exactly half of the CIF resolution, that is, 176×144 pixels for the luminance and 88×72 pixels for the other components. All H.261 implementations must be able to encode and decode QCIF.

In H.261 and H.263, data units of the size 8×8 pixels are used for the representation of the Y, as well as the C_b and C_r components. A macroblock is the result of combining four Y blocks with one block of the C_b and C_r components. A group of blocks is defined to consist of 33 macroblocks. Therefore, a QCIF-image consists of three groups of blocks, and a CIF-image comprises twelve groups of blocks. Two types of pictures are considered in the H.261 coding. These are I-pictures (or intraframes) and P-pictures (or interframes). For I-picture encoding, each macroblock is intracoded. That is, each block of 8×8 pixels in a macroblock is transformed into 64 coefficients by a use of DCT and then quantized. The quantization of dc-coefficients differs from that of ac-coefficients. The next step is to apply entropy encoding to the dc- and ac-parameters, resulting in a variable-length encoded word. For P-picture encoding, the macroblocks are either MC + DCT coded or intracoded. The prediction of MC + DCT coded macroblocks is determined by a comparison of macroblocks from previous images and the current image. Subsequently, the components of the motion vector are entropy encoded by a use of a lossless variable-length coding system. To improve the coding efficiency for low bit-rate applications, several new coding tools are included in H.263. Among them are the PB-picture type and overlapped motion compensation, and so on.

MPEG

The ISO/IEC/JTC1/SC29/WG11 MPEG working group has produced three specifications, ISO 11172 MPEG-1, ISO 13818 MPEG-2, and ISO 14496 MPEG-4 (8), for coding of combined video and audio information. MPEG-1 is intended for image resolutions of approximately CIF or SIF (360×240) and bit rates of about 1.5 Mbit/s for both video and audio. MPEG-2 is specified for higher resolutions (including interlaced video) and higher bit rates (4 Mbit/s to 15 Mbit/s, or more). MPEG-4 was originally targeted for very low bitrate coding applications. The targeted applications were modified after MPEG-4 compression was found to be effective over a wide range of bitrates. In addition, a completely new concept of encoding a scene as separate "AV" objects was developed in MPEG-4. There are three major parts composed in the MPEG-1, -2, and -4 specifications: Part 1, Systems; Part 2, Video; and Part 3, Audio. The system part specifies a system coding layer for combining coded video and audio and also provides the capability of combining private data streams and streams that may be defined at a later date. The specification describes the syntax and semantic rules of the coded data stream.

MPEG's system coding layer specifies a multiplex of elementary streams such as audio and video, with a syntax that includes data fields directly supporting synchronization of the elementary streams. The system data fields also assist in the following tasks:

1. Parsing the multiplexed stream after a random access
2. Managing coded information buffers in the decoders
3. Identifying the absolute time of the coded information

The system semantic rules impose some requirements on the decoders; however, the encoding process is not specified in the ISO document and can be implemented in a variety of ways, as long as the resulting data stream meets the system requirements.

MPEG-1, -2 and -4 video often use three types of frames (or pictures): Intra (I) frames; Predicted (P) frames; and Bidirectional (B) frames. Similar to H.261, I-type frames are compressed using only the information provided by the DCT algorithm. P-frames are derived from the preceding I frames (or from other P frames) by using MC (predicting motion forward in time) + DCT; P frames are compressed to approximately 60:1. Bidirectional B interpolated frames are derived from the previous I or P frame and the future I or P frame. B frames are required to achieve the low average data rate. Field-block-based DCT and MC were developed in MPEG-2 for efficient coding of interlaced video. MPEG-1 and -2 video can yield compression ratios of 50:1 to 200:1. It can provide 50:1 compression for broadcast quality at 6 Mbit/s. It also can provide 200:1 compression to yield VHS quality at 1.2 Mbit/s to 1.5 Mbit/s. MPEG-2 can also provide high-quality video for High Definition Television at about 18 Mbit/s.

Note that the MPEG video coding algorithms are asymmetrical. Namely, in general, it requires more computational complexity to compress full-motion video than to decompress it. This is useful for applications where the signal is produced at one source but is distributed to many.

The MPEG standards also specify efficient compression algorithms for high-performance audio (8). For example, MPEG-1 audio coding uses the same sampling frequencies as Compact Disc Digital Audio and Digital Audio Tape, that is, 44.1 kHz and 48 kHz, additionally, 32 kHz is available, all at 16 bits. Three layers of an encoder are shown in Fig. 12. An implementation of a higher layer must be able to decode the MPEG audio signals of lower layers. Similar to the use of the two-dimensional DCT for video, a transformation into the frequency domain is applied for audio. The *Fast Fourier Transform* (*FFT*) is suitable for audio coding, and the spectrum is split into 32 noninterleaved subbands. For each subband, the amplitude of the audio signal is calculated. Also, for each subband, the noise level is determined simultaneously to the actual FFT by using a psychoacoustic model. At a higher noise level, a coarse quantization is performed, and at a lower noise level, a finer quantization is applied. The quantized spectral portions of layers one and two are PCM-encoded and those of layer three are Huffman-encoded. The audio coding can be performed with a single channel, two independent channels, or one stereo signal. In the definition of MPEG, there are two different stereo modes: two channels that are processed either independently or as joint stereo. In the case of joint stereo, MPEG exploits redundancy of both channels to achieve a higher compression ratio.

Each layer specifies 14 fixed bit rates for the encoded audio data stream, which, in MPEG, are addressed by a bit rate index. The minimal value is always 32 kbit/s. These layers support different maximal bit rates: layer one allows for a maximal bit rate of 448 kbit/s, layer two for 384 kbit/s, and layer three for 320 kbit/s. For layers one and two, a decoder is not required to support a variable bit rate. In layer three, a variable bit rate is specified by switching the bit rate index. For layer two, not all combinations of bit rate and mode are allowed:

- 32 kbit/s, 48 kbit/s, 56 kbit/s, and 80 kbit/s are only allowed for a single channel.
- 64 kbit/s, 96 kbit/s, 112 kbit/s, 128 kbit/s, 160 kbit/s, and 192 kbit/s are 192 kbit/s are allowed for all modes.
- 224 kbit/s, 256 kbit/s, 320 kbit/s, and 384 kbit/s are allowed for the modes stereo, joint stereo, and dual channel modes.

H.264/AVC/JVT

The latest video codec was developed as a joint effort between the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Motion Picture Experts Group (MPEG) (10, 11). The intent of this effort was to create a standard that would produce good video quality at half the bitrates that previous video standards such as MPEG-2 and H.263 required. The techniques used in this new standard were to be constructed in a manner to allow the new standard to be applicable over a very wide range of bitrates and resolutions. The tradeoff compared to previous standards was an increase in complexity that could be eased by Moore's law and other technological advances. The first version of this

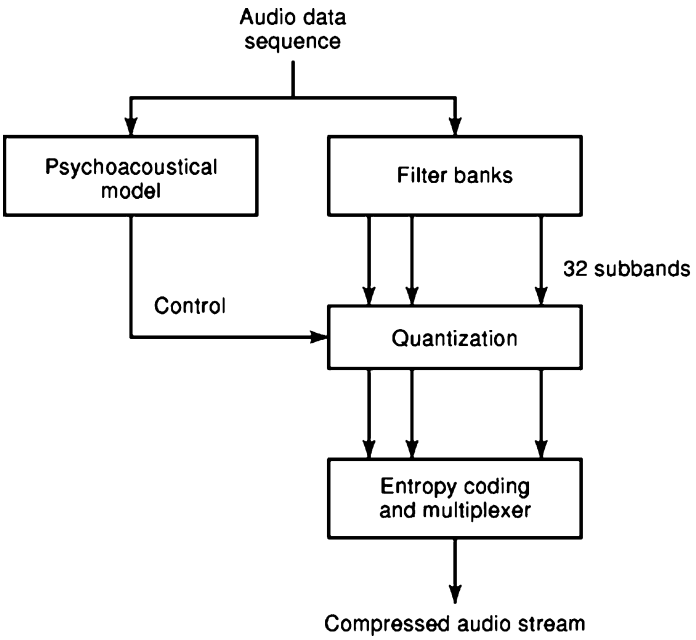

Figure 12. The key functional blocks of audio encoding in MPEG.

standard was completed in 2003. Additional extensions known as the Fidelity Range Extensions (FRExt) were finished in 2004 to support higher-fidelity video coding beyond the support of 8-bit 4:2:0 video (e.g., 10-bit, 12-bit, 4:2:2 and 4:4:4 video). The same syntax has been published by both organizations: the ITU-T H.264 standard and the ISO/IEC MPEG-4 Part 10 standard. Note that MPEG-4 Part 10 is not the same as MPEG-4 Part 2, the original video codec in the MPEG-4 suite of standards. This codec may also be referred to as the AVC (Advanced Video Coding) standard or the JVT standard which references the joint partnership between VCEG and MPEG which was known as the Joint Video Team.

H.264/AVC contains many new features for more effective video compression than older standards. Some of these features include:

- Multi-picture inter-picture prediction. Previous standards had limited inter-picture prediction to one (for P-pictures) or two (for B-pictures) reference pictures. H.264/AVC allows for up to 16 reference pictures to be used. In addition, there are fewer restrictions on the pictures that can be used for prediction. For example, in MPEG-2, B-pictures were not allowed to be used as reference pictures for the prediction of other pictures. This restriction is not present in H.264/AVC.
- Variable block-size motion compensation. Block sizes ranging from 4×4 pixels to 16×16 pixels can be chosen to match the size of objects and regions in the video content.
- Motion compensation using increased fractional-pixel precision. While half-sample precision was used in MPEG-1, MPEG-2 and H.263, quarter-sample precision is used for luma and eighth-sample preceision is used for chroma in H.264/AVC.
- Spatial prediction from neighboring blocks for intra-coding. For example, only DC coefficients were predicted in MPEG-2. In H.264/AVC, spatial prediction using neighboring blocks is performed for AC coefficients.
- An exact integer transform similar to the DCT is specified to allow for exact decoding. Previous standards specified approximations to the ideal DCT which may result in drift when the encoder and decoder implementations differed.
- In-loop deblocking filter to reduce the blocking artifacts common to DCT-based compression algorithms.
- Context-adaptive binary arithmetic coding and context-adaptive variable-length coding that are more efficient than previous entropy coding.

For evaluation of video, image and audio quality, subjective criteria are often used. The subjective criteria employ rating scales such as goodness scales and impairment scales. A goodness scale may be a global scale or a group scale. The overall goodness criterion rates perceptual quality on a scale ranging from excellent to unsatisfactory. A training set is used to calibrate such a scale. The group goodness scale is based on comparisons within a set of data. The impairment scale rates an image, video or audio sequence on the basis of the level of degradation present when compared with a reference image, video or audio sequence. It is useful in applications such as video coding, where the encoding process might introduce degradation in the output images.

BIBLIOGRAPHY

1. N. S. Jayant and P. Noll, *Digital Coding of Waveform*, Englewood Cliffs, NJ: Prentice-Hall, 1984.
2. K. R. Rao and P. Ypi, *Discrete Cosine Transform*, San Diego: Academic Press, 1990.
3. A. K. Jain, *Fundamentals of Digital Image Processing*, Englewood Cliffs, NJ: Prentice-Hall, 1989.

4. W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*, New York: Van Nostrand Reinhold, 1993.
5. J. W. Woods (ed.), *Subband Image Coding*, Boston: Kluwer, 1991.
6. K. Jack, *Video Demystified*, 2nd ed., San Diego: HighText Interactive, 1996.
7. T. M. Cover and J. A. Thomas, *Elements of Information Theory*, New York: Wiley, 1991.
8. B. G. Haskell, A. Puri, and A. N. Netravali, *Digital Video: An Introduction to MPEG-2*, New York: Chapman & Hall, 1997.
9. A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Boston: Kluwer, 1992.
10. G. J. Sullivan and T. Wiegand,"Video Compression – From Concepts to the H.264/AVC Standard", *Proceedings of the IEEE*, Vol. 93, Issue 1, December 2004, p. 18–31.
11. A. Puri, X. Chen and A. Luthra,"Video Coding Using the H.264/MPEG-4 AVC Compression Standard", *Signal Processing: Image Communication*,Vol. 19, Issue 9,October 2004, p. 793–849.

WADE WAN
XUEMIN CHEN
Broadcom Corporation, Irvine, CA