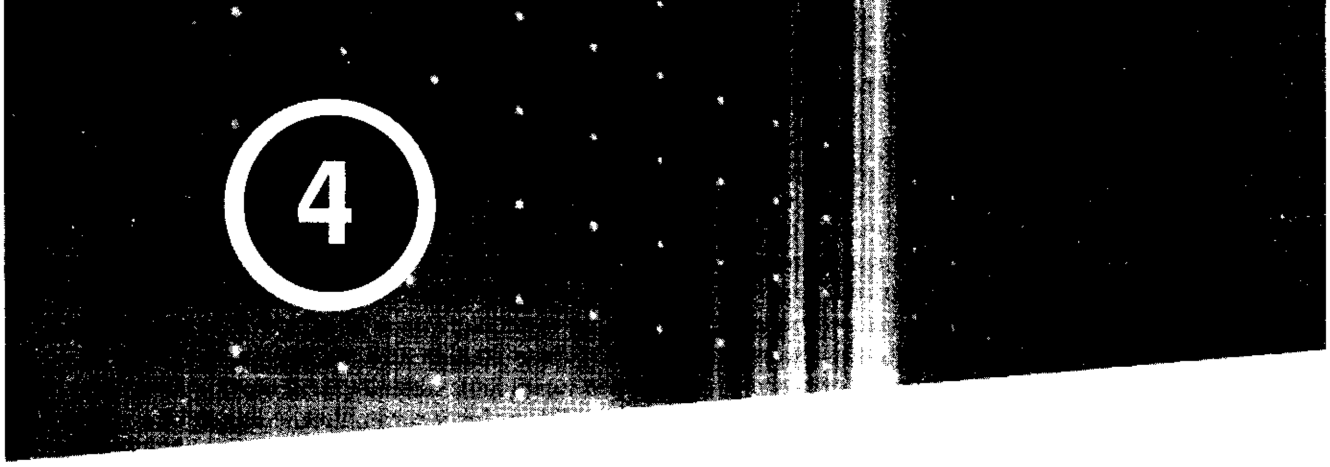


Audio and video compression

4.1 Introduction

As we described in Chapter 2, unlike text and images, both audio and most video signals are continuously varying analog signals. Hence, as we saw in Section 2.2, after digitization, both comprise a continuous stream of digital values, each value representing the amplitude of a sample of the particular analog signal taken at repetitive time intervals. As a result, the compression algorithms associated with digitized audio and video are different from those we described in the last chapter. In this chapter, therefore, we shall describe a selection of these algorithms, first those relating to audio and then those relating to video.

4.2 Audio compression

We discussed the digitization of audio signals in Section 2.5 and, as we saw, the digitization process is known as **pulse code modulation** or **PCM**. This involves sampling the (analog) audio signal/waveform at a minimum rate which is twice that of the maximum frequency component that makes up the signal. Alternatively, if the (frequency) bandwidth of the communications channel to be used is less than that of the signal, then the sampling rate is determined by the bandwidth of the communications channel. The latter is then known as a **bandlimited signal**.

In the case of a speech signal, the maximum frequency component is 10 kHz and hence the minimum sampling rate is 20 ksps. For general audio, music for example, the figures are 20 kHz and 40 ksps respectively. The number of bits per sample also varies and, typically, is 12 bits for speech and 16 bits for general audio. In addition, for a stereophonic signal, two signals must be digitized. As we saw in Example 2.4, this produces a bit rate of 240 kbps for a speech signal and 1.28 Mbps for a stereophonic music/general audio signal, the latter being the rate used with compact discs.

In practice, however, in most multimedia applications involving audio, the bandwidth of the communication channels that are available dictate rates that are less than these. This can be achieved in one of two ways: either the audio signal is sampled at a lower rate (and, if necessary, with fewer bits per sample) or a compression algorithm is used. Although the first approach is relatively simple to implement, using a lower sampling rate has the disadvantage that the quality of the decoded signal is reduced owing to the loss of the higher-frequency components from the original signal. Also, the use of fewer bits per sample results in the introduction of higher levels of quantization noise. Normally, therefore, a compression algorithm is used since, as we shall see, this achieves a comparable perceptual quality (as perceived by the ear) to that obtained with a higher sampling rate but with a reduced bandwidth requirement. In this section we consider a selection of the compression algorithms that are widely used for audio in a range of multimedia applications.

4.2.1 Differential pulse code modulation

Differential pulse code modulation (**DPCM**) is a derivative of standard PCM and exploits the fact that, for most audio signals, the range of the differences in amplitude between successive samples of the audio waveform is less than the range of the actual sample amplitudes. Hence if only the digitized difference signal is used to encode the waveform then fewer bits are required than for a comparable PCM signal with the same sampling rate. A DPCM encoder and decoder are shown in Figure 4.1(a) and a simplified timing diagram for the encoder is shown in Figure 4.1(b).

Essentially, the previous digitized sample of the analog input signal is held in the register (R) – a temporary storage facility – and the difference signal (DPCM) is computed by subtracting the current contents of the register (R_0) from the new digitized sample output by the ADC (PCM). The value in the register is then updated by adding to the current register contents the computed difference signal output by the subtractor prior to its transmission. The decoder operates by simply adding the received difference signal (DPCM) to the previously computed signal held in the register (PCM). Typical savings with DPCM, are limited to just 1 bit which, for a standard PCM voice signal, for example, reduces the bit rate requirement from 64 kbps to 56 kbps.

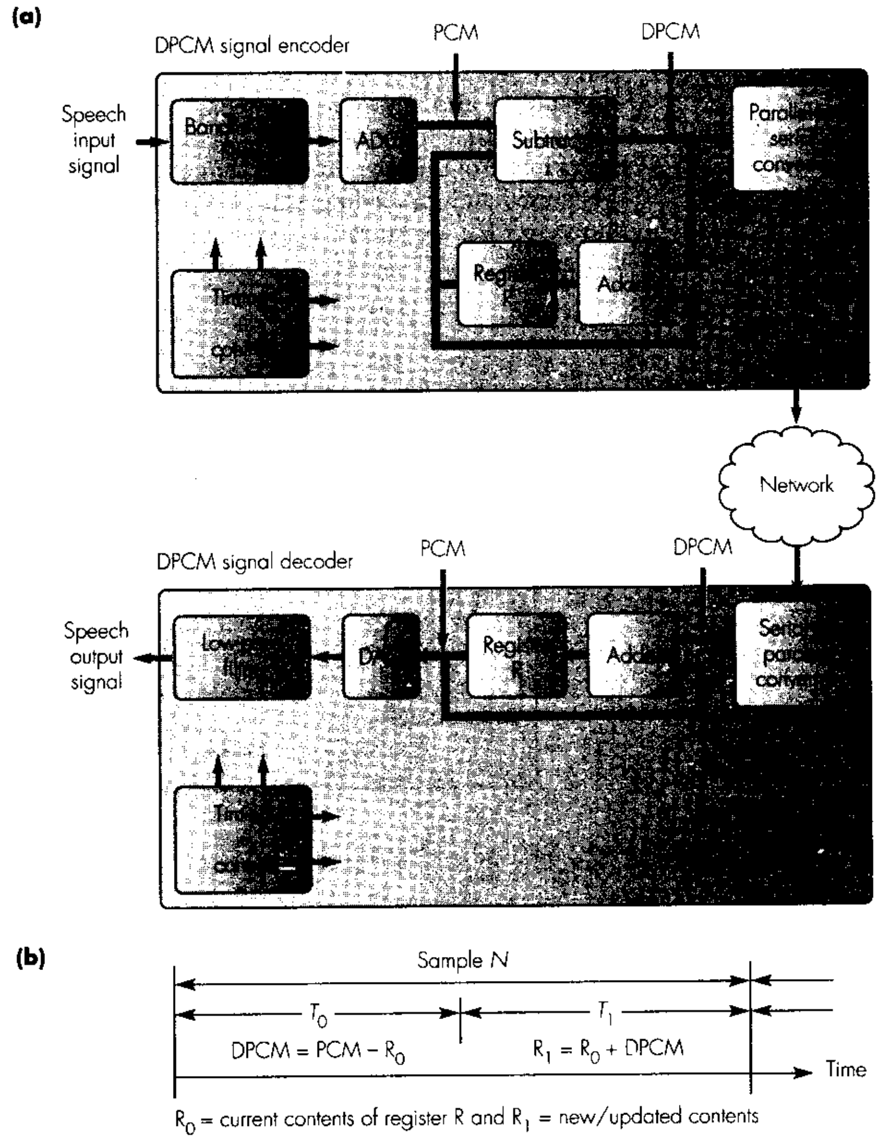


Figure 4.1 DPCM principles: (a) encoder/decoder schematic; (b) encoder timing.

As we can deduce from the circuit shown in Figure 4.1(a), the output of the ADC is used directly and hence the accuracy of each computed difference signal – also known as the **residual** (signal) – is determined by the accuracy of the previous signal/value held in the register. As we saw in Section 2.2, all ADC operations produce quantization errors and hence a string of, say, positive errors, will have a cumulative effect on the accuracy of the value that is held in the register. This means, therefore, that with a basic DPCM scheme, the previous value held in the register is only an approximation. Hence more sophisticated techniques have been developed for estimating – also known as predicting – a more accurate version of the previous signal. To achieve this, these predict the previous signal by using not only the estimate of the current signal but also varying proportions of a number of the immediately preceding estimated signals. The proportions used are determined by what are known as **predictor coefficients** and the principle is shown in Figure 4.2.

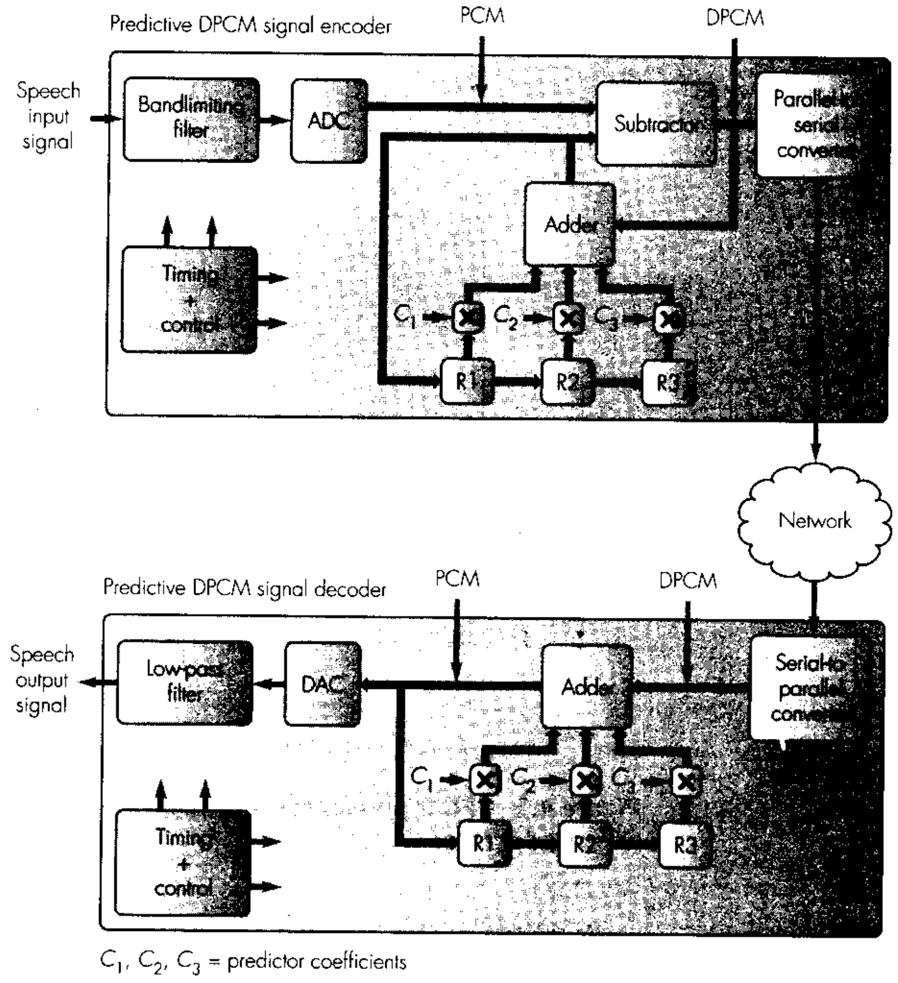


Figure 4.2 Third-order predictive DPCM signal encoder and decoder schematic.

In this example, the difference signal is computed by subtracting varying proportions of the last three predicted values from the current digitized value output by the ADC. For example, if the three predictor coefficients have the values $C_1 = 0.5$ and $C_2 = C_3 = 0.25$, then the contents of register R1 would be shifted right by 1 bit (thereby multiplying its contents by 0.5) and registers R2 and R3 by 2 bits (multiplying their contents by 0.25). The three shifted values are then added together and the resulting sum subtracted from the current digitized value output by the ADC (PCM). The current contents of register R1 are then transferred to register R2 and that of register R2 to register R3. The new predicted value is then loaded into register R1 ready for the next sample to be processed. The decoder operates in a similar way by adding the same proportions of the last three computed PCM signals to the received DPCM signal. By using this approach, a similar performance level to standard PCM is obtained by using only 6 bits for the difference signal which produces a bit rate of 32 kbps.

4.2.2 Adaptive differential PCM

Additional savings in bandwidth – or improved quality – can be obtained by varying the number of bits used for the difference signal depending on its amplitude; that is, using fewer bits to encode (and hence transmit) smaller difference values than for larger values. This is the principle of **adaptive differential PCM (ADPCM)** and an international standard for this is defined in **ITU-T Recommendation G.721**. This is based on the same principle as DPCM except an eight-order predictor is used and the number of bits used to quantize each difference value is varied. This can be either 6 bits – producing 32 kbps – to obtain a better quality output than with third-order DPCM, or 5 bits – producing 16 kbps – if lower bandwidth is more important.

A second ADPCM standard, which is a derivative of G.721, is defined in **ITU-T Recommendation G.722**. This provides better sound quality than the G.721 standard at the expense of added complexity. To achieve this, it uses an added technique known as **subband coding**. The input speech bandwidth is extended to be from 50 Hz through to 7 kHz – compared with 3.4 kHz for a standard PCM system – and hence the wider bandwidth produces a higher-fidelity speech signal. This is particularly important in conferencing applications, for example, in order to enable the members of the conference to discriminate between the different voices of the members present. The general principle of the scheme is shown in Figure 4.3.

To allow for the higher signal bandwidth, prior to sampling the audio input signal, it is first passed through two filters: one which passes only signal frequencies in the range 50 Hz through to 3.5 kHz, and the other only frequencies in the range 3.5 kHz through to 7 kHz. By doing this, the input (speech) signal is effectively divided into two separate equal-bandwidth signals, the first known as the **lower subband signal** and the second the **upper subband signal**. Each is then sampled and encoded independently using

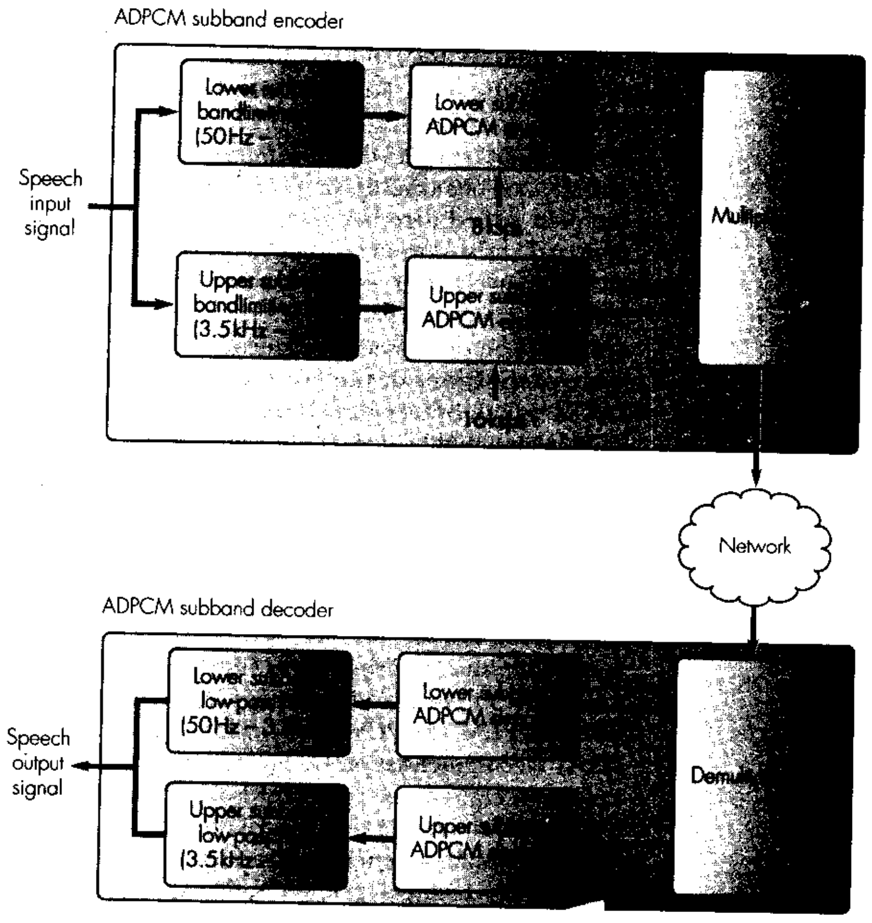


Figure 4.3 ADPCM subband encoder and decoder schematic.

ADPCM, the sampling rate of the upper subband signal being 16 ksps to allow for the presence of the higher frequency components in this subband.

The use of two subbands has the advantage that different bit rates can be used for each. In general, the frequency components that are present in the lower subband signal have a higher perceptual importance than those in the higher subband. The operating bit rate can be 64, 56, or 48 kbps. With a bit rate of 64 kbps, for example, the lower subband is ADPCM encoded at 48 kbps and the upper subband at 16 kbps. The two bitstreams are then multiplexed (merged) together – to produce the transmitted (64 kbps) signal – in such a way that the decoder in the receiver is able to divide them back again into two separate streams for decoding.

A third standard based on ADPCM is also available. This is defined in **ITU-T Recommendation G.726**. This also uses subband coding but with a speech bandwidth of 3.4 kHz. The operating bit rate can be 40, 32, 24, or 16 kbps.

4.2.3 Adaptive predictive coding

Even higher levels of compression – but at higher levels of complexity – can be obtained by also making the predictor coefficients adaptive. This is the principle of **adaptive predictive coding** (**APC**) and with this, the predictor coefficients continuously change.

In practice, the optimum set of predictor coefficients continuously vary since they are a function of the characteristics of the audio signal being digitized; for example, the actual frequency components that make up the signal at a particular instant of time.

To exploit this property, the input speech signal is divided into fixed time segments and, for each segment, the currently prevailing characteristics are determined. The optimum set of coefficients are then computed and these are used to predict more accurately the previous signal. This type of compression can reduce the bandwidth requirements to 8 kbps while still obtaining an acceptable perceived quality.

4.2.4 Linear predictive coding

All the algorithms we have considered so far are based on sampling the time-varying speech waveform and then either sending the quantized samples directly (PCM) or sending the quantized difference signal (DPCM and its derivatives). With the advent of inexpensive digital signal processing circuits, an alternative approach has become possible which involves the source simply analyzing the audio waveform to determine a selection of the perceptual features it contains. These are then quantized and sent and the destination uses them, together with a sound synthesizer, to regenerate a sound that is perceptually comparable with the source audio signal. This is the basis of the **linear predictive coding** (**LPC**) technique and, although with this the generated sound – normally speech – can often sound synthetic, very high levels of compression (and hence low bit rates) can be achieved.

Clearly, the key to this approach is to identify the set of perceptual features to be used and, in terms of speech, the three features which determine the perception of a signal by the ear are its:

- pitch: this is closely related to the frequency of the signal and is important because the ear is more sensitive to frequencies in the range 2–5 kHz than to frequencies that are higher or lower than these;
- period: this is the duration of the signal;
- loudness: this is determined by the amount of energy in the signal.

In addition, the origins of the sound are important. These are known as **vocal tract excitation parameters** and classified as:

- voiced sounds: these are generated through the vocal chords and examples include the sounds relating to the letters m, v, and l;

- unvoiced sounds: with these the vocal chords are open and examples include the sounds relating to the letters f and s.

Once these have been obtained from the source waveform, it is possible to use them, together with a suitable model of the vocal tract, to generate a synthesized version of the original speech signal. The basic features of an LPC encoder/decoder are shown in Figure 4.4.

The input speech waveform is first sampled and quantized at a defined rate. A block of digitized samples – known as a segment – is then analyzed to determine the various perceptual parameters of the speech that it contains. The speech signal generated by the vocal tract model in the decoder is a function of the present output of the speech synthesizer – as determined by the current set of model coefficients – plus a linear combination of the previous set of model coefficients. Hence the vocal tract model used is adaptive

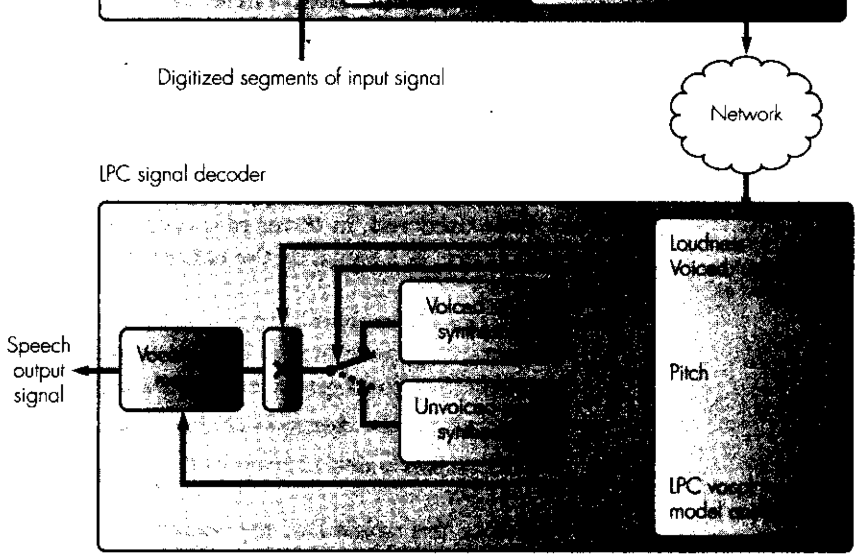


Figure 4.4 Linear predictive coding (LPC) signal encoder and decoder schematic.

and, as can be seen, the encoder determines and sends a new set of coefficients for each quantized segment.

As we can see from the above, the output of the encoder is a string of frames, one for each segment. Each frame contains fields for pitch and loudness – the period is determined by the sampling rate being used – a notification of whether the signal is voiced or unvoiced, and a new set of computed model coefficients. Some LPC encoders use up to ten sets of previous model coefficients to predict the output sound (**LPC-10**) and use bit rates as low as 2.4 kbps or even 1.2 kbps. As indicated, however, the generated sound at these rates is often very synthetic and hence LPC coders are used primarily in military applications in which bandwidth is all-important.

4.2.5 Code-excited LPC

The synthesizers used in most LPC decoders are based on a very basic model of the vocal tract. A more sophisticated version of this, known as a **code-excited linear prediction (CELP) model**, is also used and, in practice, is just one example of a family of vocal tract models known as **enhanced excitation (LPC) models**. These are also intended primarily for applications in which the amount of bandwidth available is limited but the perceived quality of the speech must be of an acceptable standard for use in various multimedia applications.

In the CELP model, instead of treating each digitized segment independently for encoding purposes, just a limited set of segments is used, each known as a **waveform template**. A precomputed set of templates are held by the encoder and decoder in what is known as a **template codebook**. Each of the individual digitized samples that make up a particular template in the codebook are differentially encoded. Each codeword that is sent selects a particular template from the codebook whose difference values best match those quantized by the encoder. In this way, there is continuity from one set of samples to another and, as a result, an improvement in sound quality is obtained.

There are now four international standards available that are based on this principle. These are **ITU-T Recommendations G.728, 729, 729(A),** and **723.1** all of which give a good perceived quality at low bit rates.

All coders of this type have a delay associated with them which is incurred while each block of digitized samples is analyzed by the encoder and the speech is reconstructed at the decoder. The combined delay value is known as the coder's **processing delay**. In addition, before the speech samples can be analyzed, it is necessary to buffer – store in memory – the block of samples. The time to accumulate the block of samples is known as the **algorithmic delay** and, in some CELP coders, this is extended to include samples from the next successive block, a technique known as **lookahead**. These delays occur in the coders, of course, and hence are in addition to the end-to-end transmission delay over the network. Nevertheless, the combined delay value of a coder is an important parameter as it often determines the suitability of the coder for a specific application. For example, in a conventional telephony application, a low-delay coder is required since a large delay can impede the flow of a conversation. In contrast, in an interactive application that involves the output of speech stored in a file, for example, a delay of several seconds before the speech starts to be output is often acceptable and hence the coder's delay is less important. Other parameters of the coder that are considered are the **complexity** of the coding algorithm and the **perceived quality** of the output speech and, in general, a compromise has to be reached between a coder's speech quality and its delay/complexity.

The delay associated with a basic PCM coder is very small as it is equal to the time interval between two successive samples of the input waveform. Hence at the basic PCM sampling rate of 8 ksps the delay is equal to 0.125 ms. This same delay also applies, of course, to ADPCM coders. In contrast, the four CELP-based standards have delay values in excess of these as multiple samples are involved. These are summarized in Table 4.1 which also includes the bit rate(s) associated with each standard and the principal application for which each has been developed. The use of the extension .1 with G.723.1 is used to discriminate this standard from the earlier G.723 standard which has now been integrated with G.721 into the G.726 standard.

Table 4.1 Summary of CELP-based standards.

Standard	Bit rate	Total coder delay	Example application domain
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

4.2.6 Perceptual coding

Both LPC and CELP are used primarily for telephony applications and hence the compression of a speech signal. Perceptual encoders, however, have been designed for the compression of general audio such as that associated with a digital television broadcast. They also use a model but, in this case, it is known as a **psychoacoustic model** since its role is to exploit a number of the limitations of the human ear.

Using this approach, sampled segments of the source audio waveform are analyzed – as with CELP-based coders – but only those features that are perceptible to the ear are transmitted. For example, although the human ear is sensitive to signals in the range 15 Hz through to 20 kHz, the level of sensitivity to each signal is non-linear; that is, the ear is more sensitive to some signals than others. Also, when multiple signals are present – as is the case with general audio – a strong signal may reduce the level of sensitivity of the ear to other signals which are near to it in frequency, an effect known as **frequency masking**. In addition, when the ear hears a loud sound, it takes a short but finite time before it can hear a quieter sound, an effect known as **temporal masking**. A psychoacoustic model is used to identify those signals that are influenced by both these effects. These are then eliminated from the transmitted signals and, in so doing, this reduces the amount of information to be transmitted.

Sensitivity of the ear

We defined the dynamic range of a signal in Section 2.2.2. It is the ratio of the maximum amplitude of the signal to the minimum amplitude and is measured in decibels (dB). In the case of the ear, its dynamic range is the ratio of the loudest sound it can hear to the quietest sound and is in the region of 96 dB. As we have just seen, however, the sensitivity of the ear varies with the frequency of the signal and, assuming just a single-frequency signal is present at any one time, the perception threshold of the ear – that is, its minimum level of sensitivity – as a function of frequency is shown in Figure 4.5(a).

As we can see, the ear is most sensitive to signals in the range 2–5 kHz and hence signals within this band are the quietest the ear is sensitive to. The vertical axis, therefore, indicates the amplitude level of all the other signal frequencies relative to this level – measured in dB – that are required for them to be heard. Hence in the figure, although the two signals A and B have the same relative amplitude, signal A would be heard – that is, it is above the hearing threshold – while signal B would not.

Frequency masking

When an audio sound consists of multiple frequency signals is present, the sensitivity of the ear changes and varies with the relative amplitude of the signals. For example, the curve shown in Figure 4.5(b) shows how the sensitivity of the ear changes in the vicinity of a loud signal. In this example, signal B is larger in amplitude than signal A and, as we can see, this causes the basic sensitivity curve of the ear to be distorted in the region of signal B. As a result, signal A will no longer be heard even though on its own, it is above the hearing threshold of the ear for a signal of that frequency. This is the origin of the term frequency masking and, in practice, the masking effect also varies with frequency as we show in Figure 4.6.

The various curves show the masking effect of a selection of different frequency signals – 1, 4, and 8 kHz – and, as we can see, the width of the masking curves – that is, the range of frequencies that are affected – increase with increasing frequency. The width of each curve at a particular signal level is known as the **critical bandwidth** for that frequency and experiments have shown that, for frequencies less than 500 Hz, the critical bandwidth remains constant at about 100 Hz. For frequencies greater than 500 Hz, however, the critical bandwidth increases (approximately) linearly in multiples of 100 Hz. For example, for a signal of 1 kHz (2 × 500 Hz), the critical bandwidth is

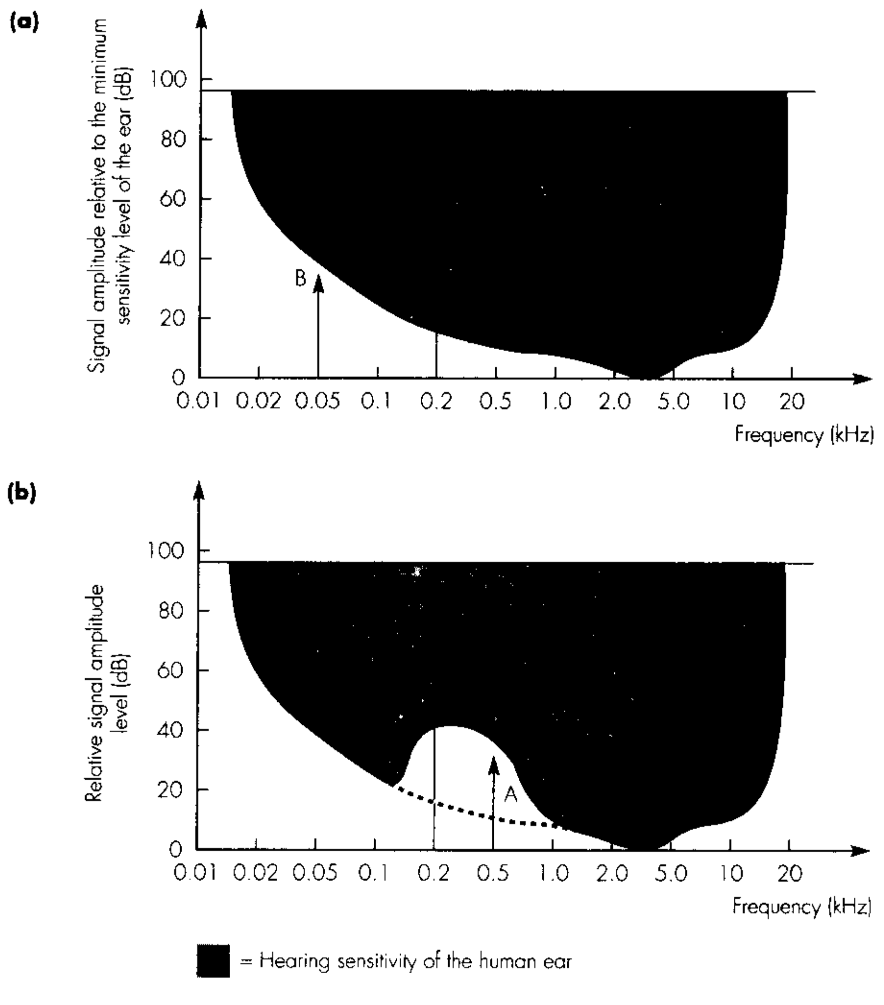


Figure 4.5 Perceptual properties of the human ear: (a) sensitivity as a function of frequency; (b) frequency masking.

about 200 (2 × 100) Hz while at 5 kHz (10 × 500 Hz) it is about 1000 (10 × 100) Hz. Hence if the magnitude of the frequency components that make up an audio sound can be determined, it becomes possible to determine those frequencies that will be masked (and hence inaudible) and do not therefore need to be transmitted.

Temporal masking

As indicated earlier, after the ear hears a loud sound, it takes a further short time before it can hear a quieter sound. This is known as temporal masking and the general effect is shown in Figure 4.7. As we can see, after the loud

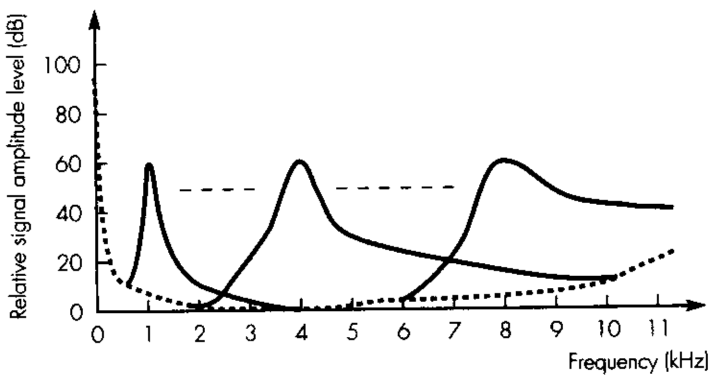


Figure 4.6 Variation with frequency of effect of frequency masking.

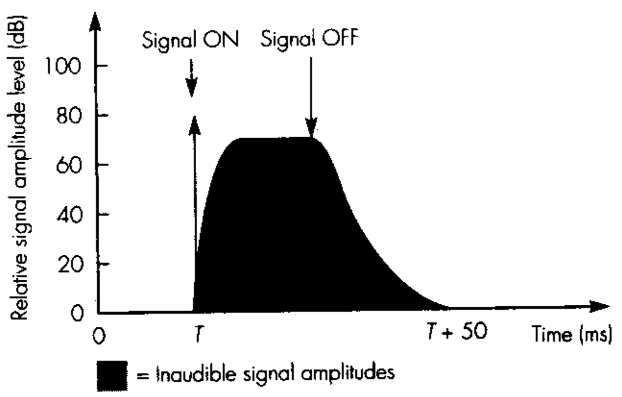


Figure 4.7 Temporal masking caused by a loud signal.

sound ceases it takes a short period of time (in the order of tens of milliseconds) for the signal amplitude to decay. During this time, signals whose amplitudes are less than the decay envelope will not be heard and hence need not be transmitted. Clearly, however, in order to exploit this phenomenon, it is necessary to process the input audio waveform over a time period that is comparable with that associated with temporal masking.

4.2.7 MPEG audio coders

In practice, perceptual coding is used in a range of different audio compression applications. For example, as we shall describe later in Section 4.3.4, the **Motion Pictures Expert Group (MPEG)** was formed by the ISO to formulate a set of standards relating to a range of multimedia applications that involve the use of video with sound. The coders associated with the audio compression part of these standards are known as **MPEG audio coders** and a number of these use perceptual coding.

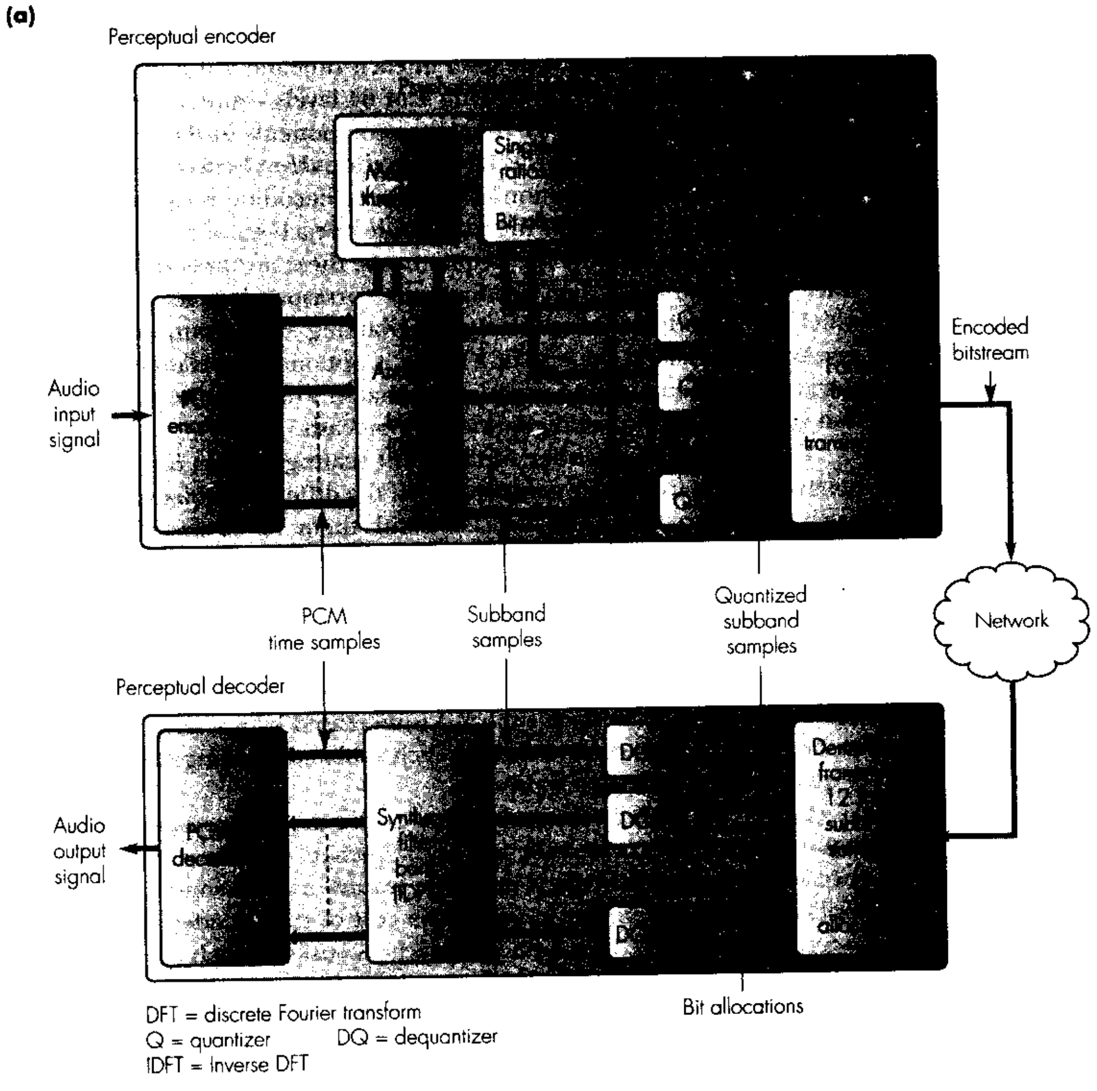
All the signal processing operations associated with a perceptual coder are carried out digitally and a schematic diagram of a basic encoder and decoder is shown in Figure 4.8(a).

The time-varying audio input signal is first sampled and quantized using PCM, the sampling rate and number of bits per sample being determined by the specific application. The bandwidth that is available for transmission is divided into a number of **frequency subbands** using a bank of **analysis filters** which, because of their role, are also known as **critical-band filters**. Each frequency subband is of equal width and, essentially, the bank of filters maps each set of 32 (time-related) PCM samples into an equivalent set of 32 frequency samples, one per subband. Hence each is known as a **subband sample** and indicates the magnitude of each of the 32 frequency components that are present in a segment of the audio input signal of a time duration equal to 32 PCM samples. For example, assuming 32 subbands and a sampling rate of 32 ksps – that is, a maximum signal frequency of 16 kHz – each subband has a bandwidth of 500 Hz.

In a basic encoder, the time duration of each sampled segment of the audio input signal is equal to the time to accumulate 12 successive sets of 32 PCM – and hence subband – samples; that is, a time duration equal to 384 (12 x 32) PCM samples.

In addition to filtering the input samples into separate frequency subbands, the analysis filter bank also determines the maximum amplitude of the 12 subband samples in each subband. Each is known as the **scaling factor** for the subband and these are passed both to the psychoacoustic model and, together with the set of frequency samples in each subband, to the corresponding quantizer block.

The processing associated with both frequency and temporal masking is carried out by the psychoacoustic model which is performed concurrently with the filtering and analysis operations. The 12 sets of 32 PCM samples are first transformed into an equivalent set of frequency components using a mathematical technique known as the **discrete Fourier transform (DFT)**. Then, using the known hearing thresholds and masking properties of each subband, the model determines the various masking effects of this set of signals. The output of the model is a set of what are known as **signal-to-mask ratios (SMRs)** and indicate those frequency components whose amplitude is below the related audible threshold. In addition, the set of scaling factors are used to determine the quantization accuracy – and hence bit allocations – to be used for each of the audible components. This is done so that those frequency components that are in regions of highest sensitivity can be quantized with more accuracy (bits) – and hence less quantization noise – than those in regions where the ear is less sensitive. In a basic encoder, all the frequency components in a sampled segment are encoded and these are carried in a frame the format of which is shown in Figure 4.8(b).



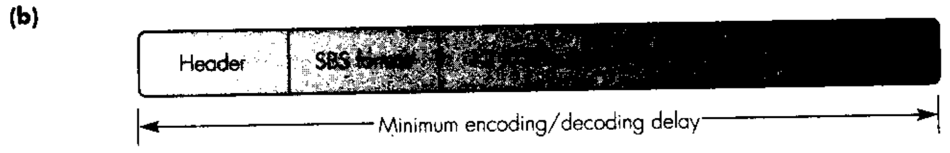


Figure 4.8 MPEG perceptual coder schematic: (a) encoder/decoder implementation schematic (b) example frame format.

The *header* contains information such as the sampling frequency that has been used. The quantization is performed in two stages using a form of companding. The peak amplitude level in each subband – the scaling factor – is first quantized using 6 bits – giving 1 of 64 levels – and a further 4 bits are then used to quantize the 12 frequency components in the subband relative to this level. Collectively this is known as the *subband sample (SBS) format* and, in this way, all the information necessary for decoding is carried within each frame. In the decoder, after the magnitude of each set of 32 subband samples have been determined by the dequantizers, these are passed to the **synthesis filter** bank. The latter then produces the corresponding set of PCM samples which are decoded to produce the time-varying analog output segment. The *ancillary data* field at the end of a frame is optional and is used, for example, to carry additional coded samples associated with, say, the surround-sound that is present with some digital video broadcasts.

The use in the encoder of different scaling factors for each subband means that the frequency components in the different subbands have varying levels of quantization noise associated with them. As we described in Section 2.2.2, this means that the frequency components in the different subbands have varying signal-to-noise ratios. The bank of synthesis filters in the decoder, however, limits the level of quantization noise in each subband to the same band of frequencies as the set of frequency components in that subband. As a result, the effect of quantization noise is reduced since the signal-to-noise ratio in each subband is increased by the larger amplitude of the signal frequency components in each subband masking the reduced level of quantization noise that is present.

As we can deduce from the figure, the psychoacoustic model is not required in the decoder and, as a consequence, it is less complex than the encoder. This is a particularly desirable feature in audio and video broadcast applications, for example, in which the cost of the decoder is an important factor. Also, it means that different psychoacoustic models can be used or, if bandwidth is plentiful, none at all.

An international standard based on this approach is defined in **ISO Recommendation 11172-3.** There are three levels of processing associated with this known as layers 1, 2, and 3. Layer 1 is the basic mode and the other two have increasing levels of processing associated with them which, in turn, produce a corresponding increase in the level of compression for the same perceived audio quality. For example, layer 1 does not include temporal masking but this is present in layers 2 and 3.

As we have already indicated, in terms of applications, MPEG audio is used primarily for the compression of general audio and, in particular, for the audio associated with various digital video applications. The performance of the three layers and examples of their corresponding application domains are summarized in Table 4.2. As we can see, the encoders associated with each of the three layers obtain increasing levels of compression and perceptual quality. The encoder and decoder delay figures are determined by the PCM sampling rate used and the corresponding frame size. For example, with layer 1, the sampling rates used are:

- 32 ksps for use with broadcast communications equipment,
- 44.1 ksps for use with CD-quality audio equipment,
- 48 ksps for use with professional sound equipment.

These produce corresponding frame durations of 12, 8.7, and 8 milliseconds with 384 samples per frame. The actual input-to-output delay, however, can be as much as two to three times these values owing to additional processing delays in the encoder and decoder. Layer 2 is identical to a standard known as **MUSICAM** and indeed was based on this. The format of each layer 2 frame is similar to that of layer 1 except that each frame contains three sets of 384 samples and hence is of a time duration equal to 1152 samples.

The bit rate figures shown in Table 4.2 are all for a single audio channel. In practice, four alternative forms of audio have been identified for multimedia applications: monophonic, dual monophonic, two-channel (also known as disjoint) stereo, and single-channel joint stereo. The latter is the digitized version of the composite stereo sound signal and hence exploits the redundancy that is present between the two channels. So the bandwidth required for audio may be the figures shown in the table – for monophonic and joint stereo – or double the values shown – for dual monophonic and two-channel stereo. Since the three layers require increasing levels of complexity (and hence cost) to achieve a particular perceived quality, the choice of layer and bit rate is often a compromise between the desired perceived quality and the available bit rate.

Table 4.2 Summary of MPEG layer 1, 2 and 3 perceptual encoders

Layer	Application	Compressed bit rate	Quality	Example input-to-output delay
1	Digital audio cassette	32 – 448 kbps	Hi-fi quality at 192 kbps per channel	20ms
2	Digital audio and digital video broadcasting	32 – 192 kbps	Near CD-quality at 128 kbps per channel	40ms
3	CD-quality audio over low bit rate channels	64 kbps	CD-quality at 64 kbps per channel	60ms

4.2.8 Dolby audio coders

The psychoacoustic models associated with the various MPEG coders control the quantization accuracy of each subband sample by computing and allocating the number of bits to be used to quantize each sample. Since the quantization accuracy that is used for each sample in a subband may vary from one set of subband samples to the next, the bit allocation information that is used to quantize the samples in each subband is sent with the actual quantized samples. This information is then used by the decoder to dequantize the set of subband samples in the frame. This mode of operation of a perceptual coder is known, therefore, as the **forward adaptive bit allocation mode** and, for comparison purposes, a simplified schematic diagram showing this operational mode is given in Figure 4.9(a). As we indicated at the end of the last section, it has the advantage that the psychoacoustic model is required only in the encoder. It has the disadvantage, however, that a significant portion of each encoded frame contains bit allocation information which, in turn, leads to a relatively inefficient use of the available bit rate.

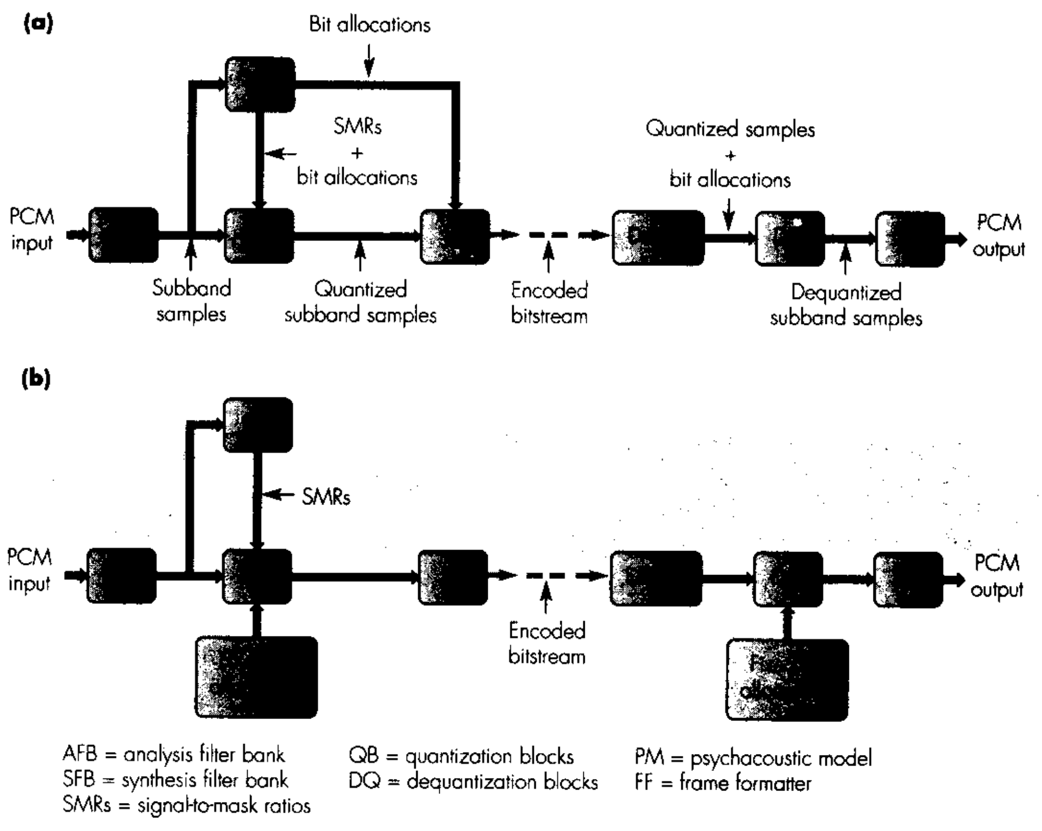

(a)
Bit allocations
SMRs + bit allocations
Quantized samples + bit allocations
PCM input
PCM output
Subband samples
Quantized subband samples
Encoded bitstream
Dequantized subband samples
(b)
SMRs
PCM input
PCM output
Encoded bitstream
AFB = analysis filter bank
SFB = synthesis filter bank
SMRs = signal-to-mask ratios
QB = quantization blocks
DQ = dequantization blocks
PM = psychoacoustic model
FF = frame formatter

Figure 4.9 Perceptual coder schematics: (a) forward adaptive bit allocation (MPEG); (b) fixed bit allocation (Dolby AC-1).

A variation of this approach is to use a fixed bit allocation strategy for each subband which is then used by both the encoder and decoder. The principle of operation of this mode is shown in Figure 4.9(b). Typically, the bit allocations that are selected for each subband are determined by the known sensitivity characteristics of the ear and the use of fixed allocations means that this information need not be sent in the frame. This approach is used in a standard known as **Dolby AC-1**, the acronym "AC" meaning **acoustic coder**. It was designed for use in satellites to relay FM radio programs and the sound associated with television programs. It uses a low-complexity psychoacoustic model with 40 subbands at a sampling rate of 32 ksps and proportionately more at 44.1 and 48 ksps. A typical compressed bit rate is 512 kbps for two-channel stereo.

A second variation, which allows the bit allocations per subband to be adaptive while at the same time minimizing the overheads in the encoder bit-stream, is for the decoder also to contain a copy of the psychoacoustic model. This is then used by the decoder to compute the same – or very similar – bit allocations that the psychoacoustic model in the encoder has used to quantize each set of subband samples. Clearly, however, in order for the psychoacoustic model in the decoder to carry out its own computation of the bit allocations, it is necessary for it to have a copy of the subband samples. Hence with this operational mode, instead of each frame containing bit allocation information – in addition to the set of quantized samples – it contains the encoded frequency coefficients that are present in the sampled waveform segment. This is known as the **encoded spectral envelope** and this mode of operation, the **backward adaptive bit allocation mode**. Figure 4.10(a) illustrates the principle of operation of both the encoder and decoder.

This approach is used in the **Dolby AC-2** standard which is utilized in many applications including the compression associated with the audio of a number of PC sound cards. Typically, these produce audio of hi-fi quality at a bit rate of 256 kbps. For broadcast applications, however, it has the disadvantage that, since the same psychoacoustic model is required in the decoder, the model in the encoder cannot be modified without changing all decoders.

To meet this requirement, a third variation has been developed that uses both backward and forward bit allocation principles. This is known as the **hybrid backward/forward adaptive bit allocation mode** and is illustrated in Figure 4.10(b).

As we can deduce from part (a) of the figure, with the backward bit allocation method on its own, since the psychoacoustic model uses the encoded spectral envelope, the quantization accuracy of the subband samples is affected by the quantization noise introduced by the spectral encoder. Hence in the hybrid scheme, although a backward adaptive bit allocation scheme is used as in AC-2 – using PM_B – an additional psychoacoustic model – PM_F – is used to compute the difference between the bit allocations computed by PM_B and those that are computed by PM_F using the forward-adaptive bit allocation scheme. This information is then used by PM_B to improve the quantization

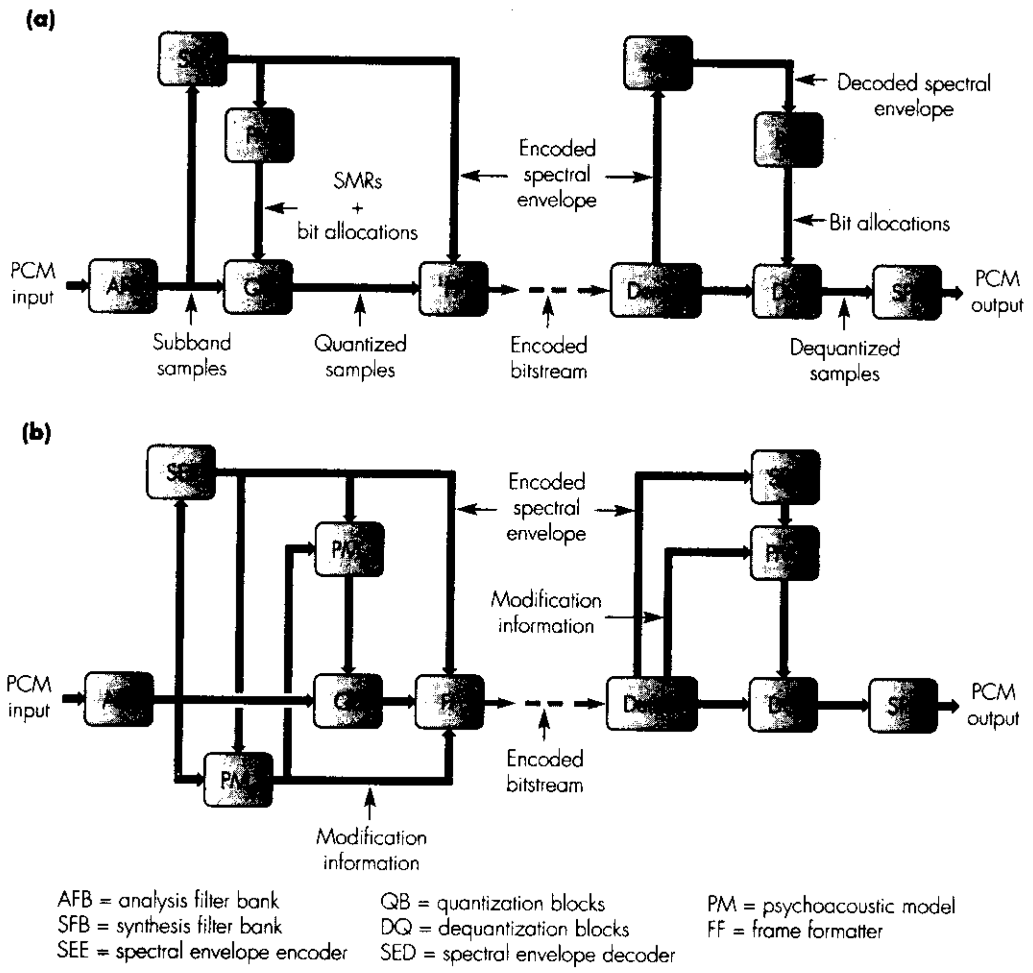


Figure 4.10 Perceptual coder schematic: (a) backward adaptive bit allocation (Dolby AC-2); (b) hybrid backward/forward adaptive bit allocation (Dolby AC-s).

accuracy of the set of subband samples. The modification information is also sent in the encoded frame and is used by the PMB in the decoder to improve the dequantization accuracy. In addition, should it be required to modify the operational parameters of the PMB in the encoder and decoder(s), then this information can be sent also with the computed difference information. As we can see from the figure, the PMF must compute two sets of quantization information for each set of subband samples and hence is relatively complex. However, since this is not required in the decoder, this is not an issue.

The hybrid approach is used in the **Dolby AC-3** standard which has been defined for use in a similar range of applications as the MPEG audio standards including the audio associated with **advanced television** (ATV). This is the HDTV standard in North America and, in this application, the acoustic quality of both the MPEG and Dolby audio coders were found to be comparable. The sampling rate can be 32, 44.1, or 48 ksps depending on the bandwidth of the source audio signal. Each encoded block contains 512 subband samples. However, in order to obtain continuity from one block to the next, the last 256 subband samples in the previous block are repeated to become the first 256 samples in the next block and hence each block contains only 256 new samples. Assuming a PCM sampling rate of 32 ksps, although each block of samples is of 8 ms duration – 256/32 – the duration of each encoder block is 16 ms. The audio signal bandwidth at this sampling rate is 15 kHz and hence each subband has a bandwidth of 62.5 Hz, that is, 15 k/256. The (stereo) bit rate is, typically, 192 kbps.

4.3 Video compression

As we described in Chapter 1, video (with sound) features in a number of multimedia applications:

- interpersonal: video telephony and videoconferencing;
- interactive: access to stored video in various forms;
- entertainment: digital television and movie/video-on-demand.

However, as we described in Section 2.6.2, the quality of the video used in these applications varies and is determined by the digitization format and frame refresh rate used. As you may recall, the digitization format defines the sampling rate that is used for the luminance, Y, and two chrominance, C_b and C_r, signals and their relative position in each frame. We derived a number of (worst-case) bit rates that are generated with different digitization formats and these ranged from in the order of 10 Mbps for the sub-quarter common intermediate format (SQCIF), as used for video telephony, to 162 Mbps for the 4:2:0 format as used for digital television broadcasts.

In practice, therefore, as we described in Chapter 1, the bit rate requirements resulting from all the digitization formats defined in Section 2.6.2 are substantially larger than the bit rates of the transmission channels that are available with the networks associated with these applications. It is for this reason that compression is used in all applications that involve video. However, there is not just a single standard associated with video but rather a range of standards, each targeted at a particular application domain. The majority are all now international standards and we shall describe a number of these in this section. Prior to doing this, however, we shall first describe the basic principles on which the standards are based.

4.3.1 Video compression principles

In the context of compression, since video is simply a sequence of digitized pictures, video is also referred to as **moving pictures** and the terms "frame" and "picture" are used interchangeably. In general, we shall use the term frame except where a particular standard uses the term picture.

In principle, one approach to compressing a video source is to apply the JPEG algorithm described earlier in Section 3.4.5 to each frame independently. This approach is known as **moving JPEG** or **MJPEG**. As we concluded at the end of that section, however, typical compression ratios obtainable with JPEG are between 10:1 and 20:1, neither of which is large enough on its own to produce the compression ratios needed.

In practice, in addition to the spatial redundancy present in each frame, considerable redundancy is often present between a set of frames since, in general, only a small portion of each frame is involved with any motion that is taking place. Examples include the movement of a person's lips or eyes in a video telephony application and a person or vehicle moving across the screen in a movie. In the case of the latter, since a typical scene in a movie has a minimum duration of about 3 seconds, assuming a frame refresh rate of 60 frames per second, each scene is composed of a minimum of 180 frames. Hence by sending only information relating to those segments of each frame that have movement associated with them, considerable additional savings in bandwidth can be made by exploiting the temporal differences that exist between many of the frames.

The technique that is used to exploit the high correlation between successive frames is to predict the content of many of the frames. As we shall describe, this is based on a combination of a preceding – and in some instances a succeeding – frame. Instead of sending the source video as a set of individually-compressed frames, just a selection is sent in this form and, for the remaining frames, only the differences between the actual frame contents and the predicted frame contents are sent. The accuracy of the prediction operation is determined by how well any movement between successive frames is estimated. This operation is known as **motion estimation** and, since the estimation process is not exact, additional information must also be sent to indicate any small differences between the predicted and actual positions of the moving segments involved. The latter is known as **motion compensation** and we shall discuss each issue separately.

Frame types

As we can see from the above, there are two basic types of compressed frame: those that are encoded independently and those that are predicted. The first are known as **intracoded frames** or **I-frames**. In practice, there are two types of predicted frames: **predictive** or **P-frames** and **bidirectional** or **B-frames** and because of the way they are derived, the latter are also known as **intercoded** or **interpolation frames**. A typical sequence of frames involving just I- and P-frames is shown in Figure 4.11(a) and a sequence involving all three frame types is shown in part (b) of the figure.

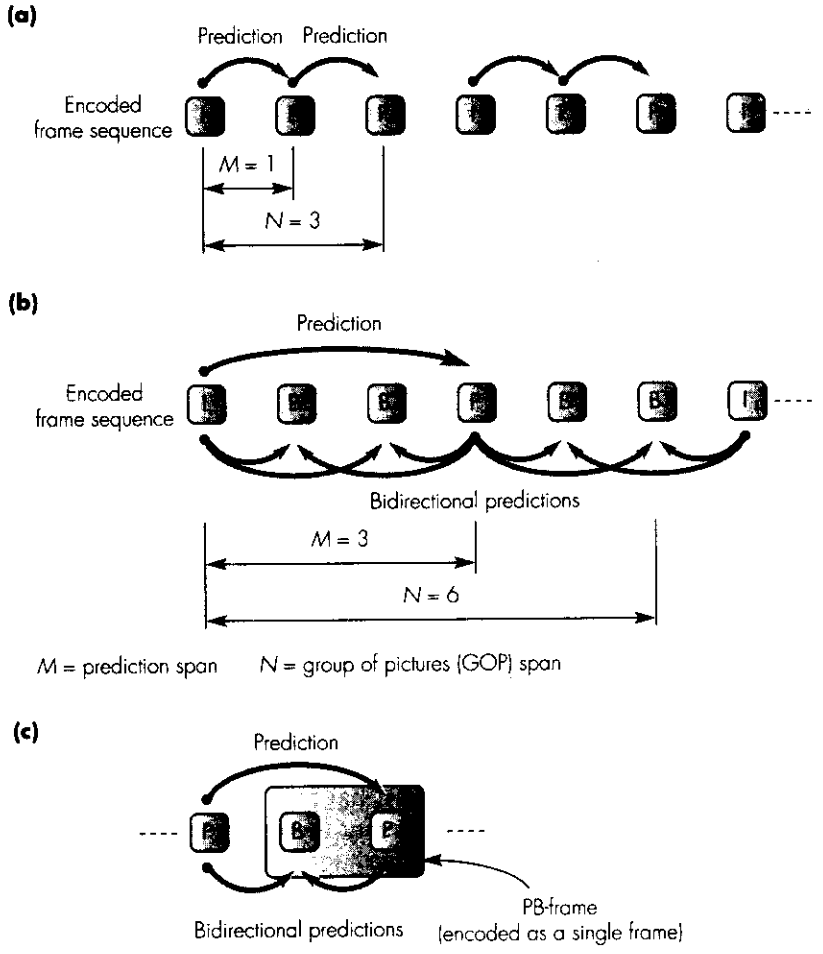


Figure 4.11 Example frame sequences with: (a) I- and P-frames only; (b) I-, P- and B-frames; (c) PB-frames.

I-frames are encoded without reference to any other frames. Each frame is treated as a separate (digitized) picture and the Y, C_b and C_r matrices are encoded independently using the JPEG algorithm (DCT, quantization, entropy encoding – described earlier in Section 3.4.5) except that the quantization threshold values that are used are the same for all DCT coefficients. Hence the level of compression obtained with I-frames is relatively small. In principle, therefore, it would appear to be best if these were limited to, say, the first frame relating to a new scene in a movie. In practice, however, the compression algorithm is independent of the contents of frames and hence has no knowledge of the start and end of scenes. Also, I-frames must be present in the output stream at regular intervals in order to allow for the possibility of the contents of an encoded I-frame being corrupted during transmission. Clearly, if an I-frame was corrupted then, in the case of a movie, since the predicted frames are based on the contents of an I-frame, a complete scene would be lost which, of course, would be totally unacceptable. Normally, therefore, I-frames are inserted into the output stream relatively frequently. The number of frames/pictures between successive I-frames is known as a **group of pictures** or **GOP**. It is given the symbol *N* and typical values for *N* are from 3 through to 12.

As we can deduce from Figure 4.11(a), the encoding of a P-frame is relative to the contents of either a preceding I-frame or a preceding P-frame. As indicated, P-frames are encoded using a combination of motion estimation and motion compensation and hence significantly higher levels of compression can be obtained with them. In practice, however, the number of P-frames between each successive pair of I-frames is limited since any errors present in the first P-frame will be propagated to the next. The number of frames between a P-frame and the immediately preceding I- or P-frame is called the **prediction span**. It is given the symbol *M* and typical values range from 1, as in Figure 4.11(a), through to 3, as in Figure 4.11(b).

As we shall describe shortly, motion estimation involves comparing small segments of two consecutive frames for differences and, should a difference be detected, a search is carried out to determine to which neighbouring segment the original segment has moved. In order to minimize the time for each search, the search region is limited to just a few neighbouring segments. In applications such as video telephony, the amount of movement between consecutive frames is relatively small and hence this approach works well. In those applications that may involve very fast moving objects, however, it is possible for a segment to have moved outside of the search region. To allow for this possibility, in applications such as movies, in addition to P-frames, a second type of prediction frame is used. These are the B-frames and, as we can see in Figure 4.11(b), their contents are predicted using search regions in both past and future frames. In addition to allowing for occasional fast moving objects, this also provides better motion estimation when, for example, an object moves in front of or behind another object.

As we can deduce from the two example frame sequences shown in Figure 4.11, for P-frames their contents are encoded by considering the contents of the current (uncoded) frame relative to the contents of the immediately preceding (uncoded) frame. In the case of B-frames, however, three (uncoded) frame contents are involved: the immediately preceding I- or P-frame, the current frame being encoded, and the immediately succeeding I- or P-frame. This results in an increase in the encoding (and decoding) delay which is equal to the time to wait for the next I- or P-frame in the sequence. In practice, however, B-frames provide the highest level of compression and, because they are not involved in the coding of other frames, they do not propagate errors.

To perform the decoding operation, the received (compressed) information relating to I-frames can be decoded immediately it is received in order to recreate the original frame. With P-frames, the received information is first decoded and the resulting information is then used, together with the decoded contents of the preceding I- or P-frame, to derive the decoded frame contents. In the case of B-frames, the received information is first decoded and the resulting information is then used, together with both the immediately preceding I- or P-frame contents and the immediately succeeding P- or I-frame contents, to derive the decoded frame contents. Hence in order to minimize the time required to decode each B-frame, the order of encoding (and transmission) of the (encoded) frames is changed so that both the preceding and succeeding I- or P-frames are available when the B-frame is received. For example, if the uncoded frame sequence is:

IBBPBBPBBI...

then the reordered sequence would be:

IPBBPBBIBBPBB.

Hence, with B-frames, the decoded contents of both the immediately preceding I- or P-frame and the immediately succeeding P- or I-frame are available when the B-frame is received.

A fourth type of frame known as a PB-frame has also been defined. This does not refer to a new frame type as such but rather the way two neighbouring P- and B-frames are encoded as if they were a single frame. The general principle is as shown in Figure 4.11(c) and, as we shall expand upon in Section 4.3.3, this is done in order to increase the frame rate without significantly increasing the resulting bit rate required.

Finally, although only used in a specific type of application, a fifth type of frame known as a **D-frame** has been defined for use in movie/video-on-demand applications. As we described in Section 1.4.3, with this type of application, a user (at home) can select and watch a particular movie/video which is stored in a remote server connected to a network. The selection operation is performed by means of a set-top box and, as with a VCR, the user may wish to rewind or fast-forward through the movie. Clearly, this requires the compressed video to be decompressed at much higher speeds. Hence to support this function, the encoded video stream also contains D-frames which are inserted at regular intervals throughout the stream. These are highly compressed frames and are ignored during the decoding of P- and B-frames. As you may recall from our earlier discussion of the DCT compression algorithm in Section 3.4.5, the DC coefficient associated with each 8 × 8 block of pixels – both for the luminance and the two chrominance signals – is the mean of all the values in the related block. Hence by using only the encoded DC coefficients of each block of pixels in the periodically inserted D-frames, a low-resolution sequence of frames is provided each of which can be decoded at the higher speeds that are expected with the rewind and fast-forward operations.

Motion estimation and compensation

As we showed earlier in Figure 4.11, the encoded contents of both P- and B-frames are predicted by estimating any motion that has taken place between the frame being encoded and the preceding I- or P-frame and, in the case of B-frames, the succeeding P- or I-frame. The various steps that are involved in encoding each P-frame are shown in Figure 4.12.

As we show in Figure 4.12(a), the digitized contents of the *Y* matrix associated with each frame are first divided into a two-dimensional matrix of

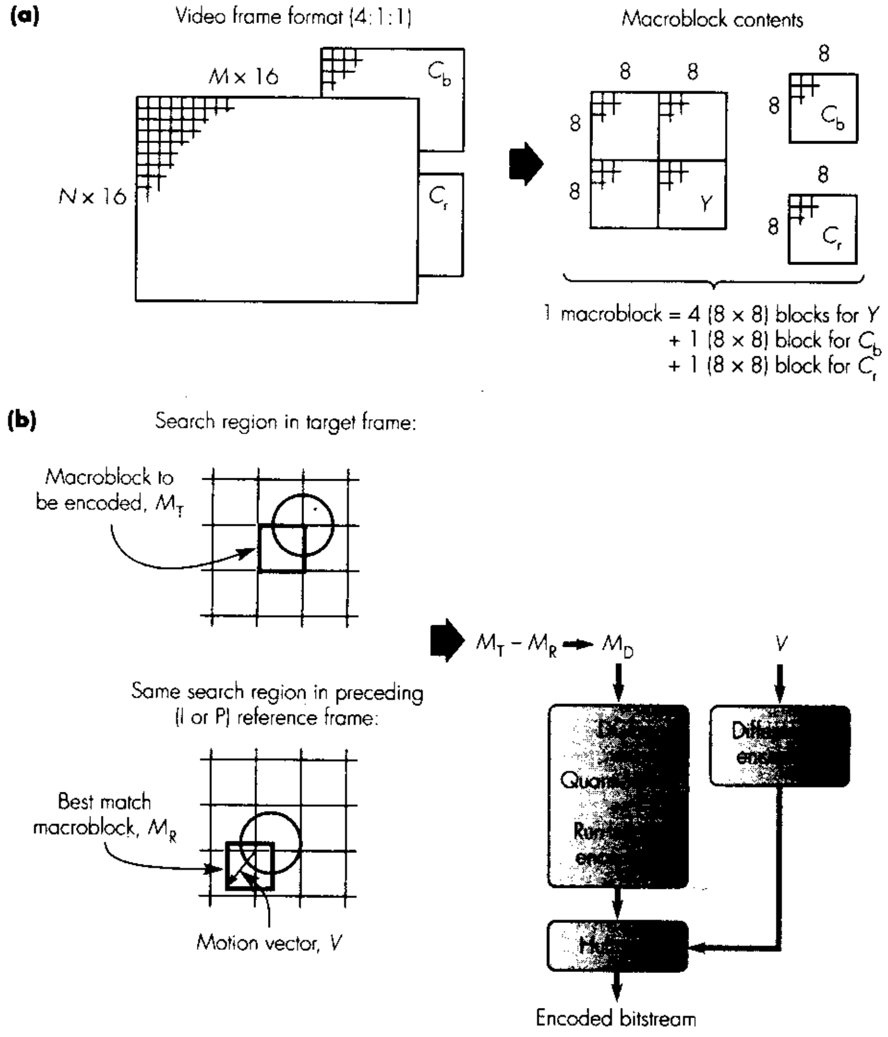

(a)
Video frame format (4:1:1)
$M \times 16$
$N \times 16$
C_b
C_r
Macroblock contents
8
8
8
8
8
8
8
8
Y
C_b
C_r
1 macroblock = 4 (8 × 8) blocks for Y
+ 1 (8 × 8) block for C_b
+ 1 (8 × 8) block for C_r
(b)
Search region in target frame:
Macroblock to be encoded, M_T
Same search region in preceding (I or P) reference frame:
Best match macroblock, M_R
Motion vector, V
$M_T - M_R \rightarrow M_D$
V
Encoded bitstream

Figure 4.12 P-frame encoding: (a) macroblock structure; (b) encoding procedure.

16 ×16 pixels known as a **macroblock**. In the example, the 4:1:1 digitization format is assumed and hence the related C_b and C_r matrices in the macroblock are both 8 × 8 pixels. For identification purposes, each macroblock has an **address** associated with it and, since the block size used for the DCT operation is also 8 × 8 pixels, a macroblock comprises four DCT blocks for luminance and one each for the two chrominance signals.

To encode a P-frame, the contents of each macroblock in the frame – known as the **target frame** – are compared on a pixel-by-pixel basis with the contents of the corresponding macroblock in the preceding -I or P-frame. The latter is known as the **reference frame**. If a close match is found, then only the address of the macroblock is encoded. If a match is not found, the search is extended to cover an area around the macroblock in the reference frame. Typically, this comprises a number of macroblocks as shown in Figure 4.12(b).

In practice, the various standards do not specify either the extent of the search area or a specific search strategy and instead specify only how the results of the search are to be encoded. Normally, only the contents of the Y matrix are used in the search and a match is said to be found if the mean of the absolute errors in all the pixel positions in the difference macroblock is less than a given threshold. Hence, using a particular strategy, all the possible macroblocks in the selected search area in the reference frame are searched for a match and, if a close match is found, two parameters are encoded. The first is known as the **motion vector** and indicates the (x,y) offset of the macroblock being encoded and the location of the block of pixels in the reference frame which produces the (close) match. The search – and hence offset – can be either on macroblock boundaries or, as in the figure, on pixel boundaries. The motion vector is then said to be **single-pixel resolution**. The second parameter is known as the **prediction error** and comprises three matrices (one each for Y, C_b and C_r) each of which contains the difference values (in all the pixel locations) between those in the target macroblock and the set of pixels in the search area that produced the close match.

Since the physical area of coverage of a macroblock is small, the motion vectors can be relatively large values. Also, most moving objects are normally much larger than a single macroblock. Hence, when an object moves, multiple macroblocks are affected in a similar way. The motion vectors, therefore, are encoded using differential encoding (DE) and the resulting codewords are then Huffman encoded. The three difference matrices, however, are encoded using the same steps as for I-frames: DCT, quantization, entropy encoding. Finally, if a match cannot be found – for example if the moving object has moved out of the extended search area – the macroblock is encoded independently in the same way as the macroblocks in an I-frame.

To encode a B-frame, any motion is estimated with reference to both the immediately preceding I- or P-frame and the immediately succeeding P- or I-frame. The general scheme is shown in Figure 4.13. The motion vector and difference matrices are computed using first the preceding frame as the

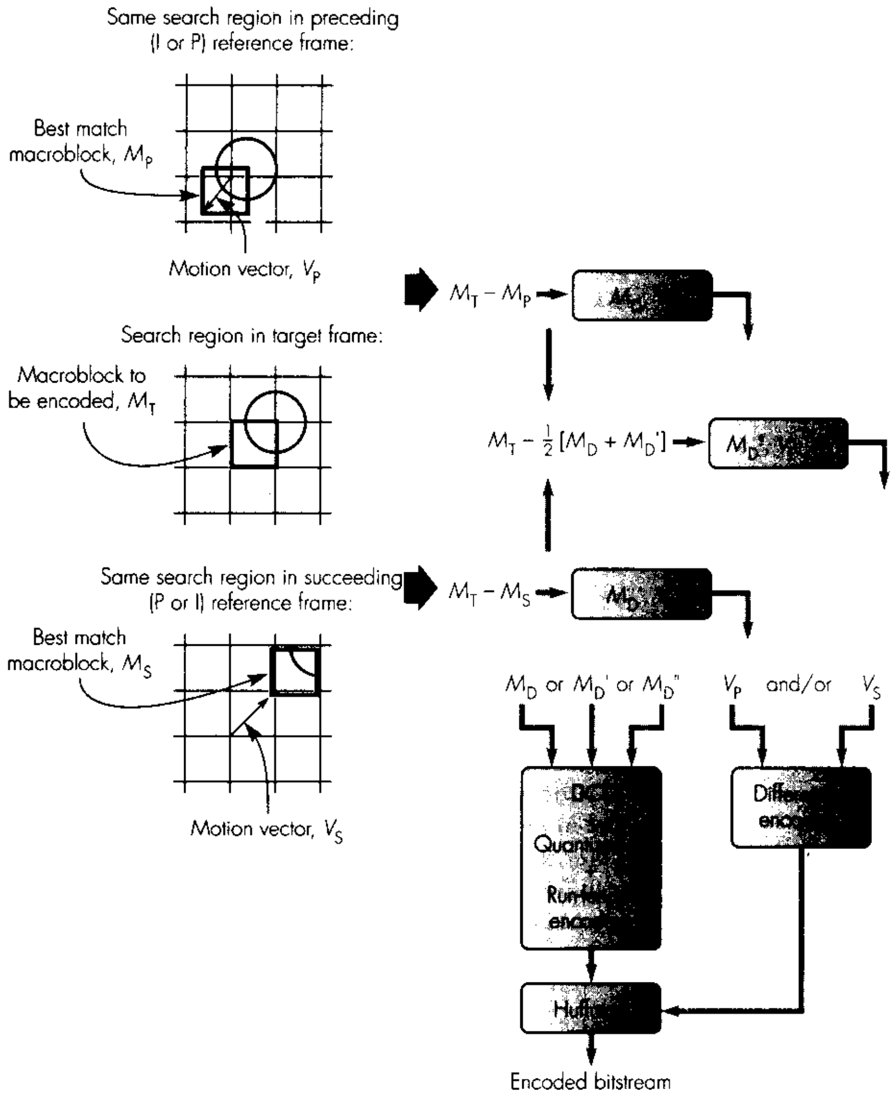

Same search region in preceding (I or P) reference frame:
Best match macroblock, M_P
Motion vector, V_P
Search region in target frame:
Macroblock to be encoded, M_T
Same search region in succeeding (P or I) reference frame:
Best match macroblock, M_S
Motion vector, V_S
$M_T - M_P$
M_D
$M_T - \frac{1}{2}[M_D + M_D']$
M_D''
$M_T - M_S$
M_D'
M_D or M_D' or M_D''
V_P and/or V_S
Encoded bitstream

Figure 4.13 B-frame encoding procedure.

reference and then the succeeding frame as the reference. A third motion vector and set of difference matrices are then computed using the target and the mean of the two other predicted sets of values. The set with the lowest set of difference matrices is then chosen and these are encoded in the same way as for P-frames. The motion vector is then said to be to a resolution of a fraction of a pixel; for example, **half-pixel resolution.**

Implementation issues

A schematic diagram showing the essential units associated with the encoding of I-, P-, and B-frames is given in parts (a), (b) and (c) of Figure 4.14 respectively and an example format of the encoded bitstream output by the encoder is given in part (d) of the figure.

As we can deduce from part (a) of the figure, the encoding procedure used for the macroblocks that make up an I-frame is the same as that used in the JPEG standard to encode each 8 × 8 block of pixels. This was described earlier in Section 3.4.5 and, as we can see, the procedure involves each macroblock being encoded using the three steps: forward DCT, quantization, and entropy encoding. Hence assuming four blocks for luminance and two for chrominance, each macroblock would require six 8 × 8 pixel blocks to be encoded.

In the case of P-frames, the encoding of each macroblock is dependent on the output of the motion estimation unit which, in turn, depends on the contents of the macroblock being encoded and the contents of the macroblock in the search area of the reference frame that produces the closest match to that being encoded. There are three possibilities:

(1) If the two contents are the same, only the address of the macroblock in the reference frame is encoded.

(2) If the two contents are very close, both the motion vector and the difference matrices associated with the macroblock in the reference frame are encoded.

(3) If no close match is found, then the target macroblock is encoded in the same way as a macroblock in an I-frame.

As we can see in Figure 4.14(b), in order to carry out its role, the motion estimation unit containing the search logic, utilizes a copy of the (uncoded) reference frame. This is obtained by taking the computed difference values – between the frame currently being compressed (the target frame) and the current reference frame – and decompressing them using the dequantize (DQ) plus inverse DCT (IDCT) blocks. After the complete target frame has been compressed, the related set of difference values are used to update the current reference frame contents ready to encode the next (target) frame. The same procedure is followed for encoding B-frames except both the preceding (reference) frame and the succeeding frame to the target frame are involved.

As we can see from the above, for each macroblock, it is necessary to identify the type of encoding that has been used. This is the role of the **formatter** and a typical format that is used to encode the macroblocks in each frame is shown in part (d) of the figure. The *type* field indicates the type of frame being encoded – I-, P-, or B- – and the *address* identifies the location of the macroblock in the frame. The *quantization value* is the threshold value

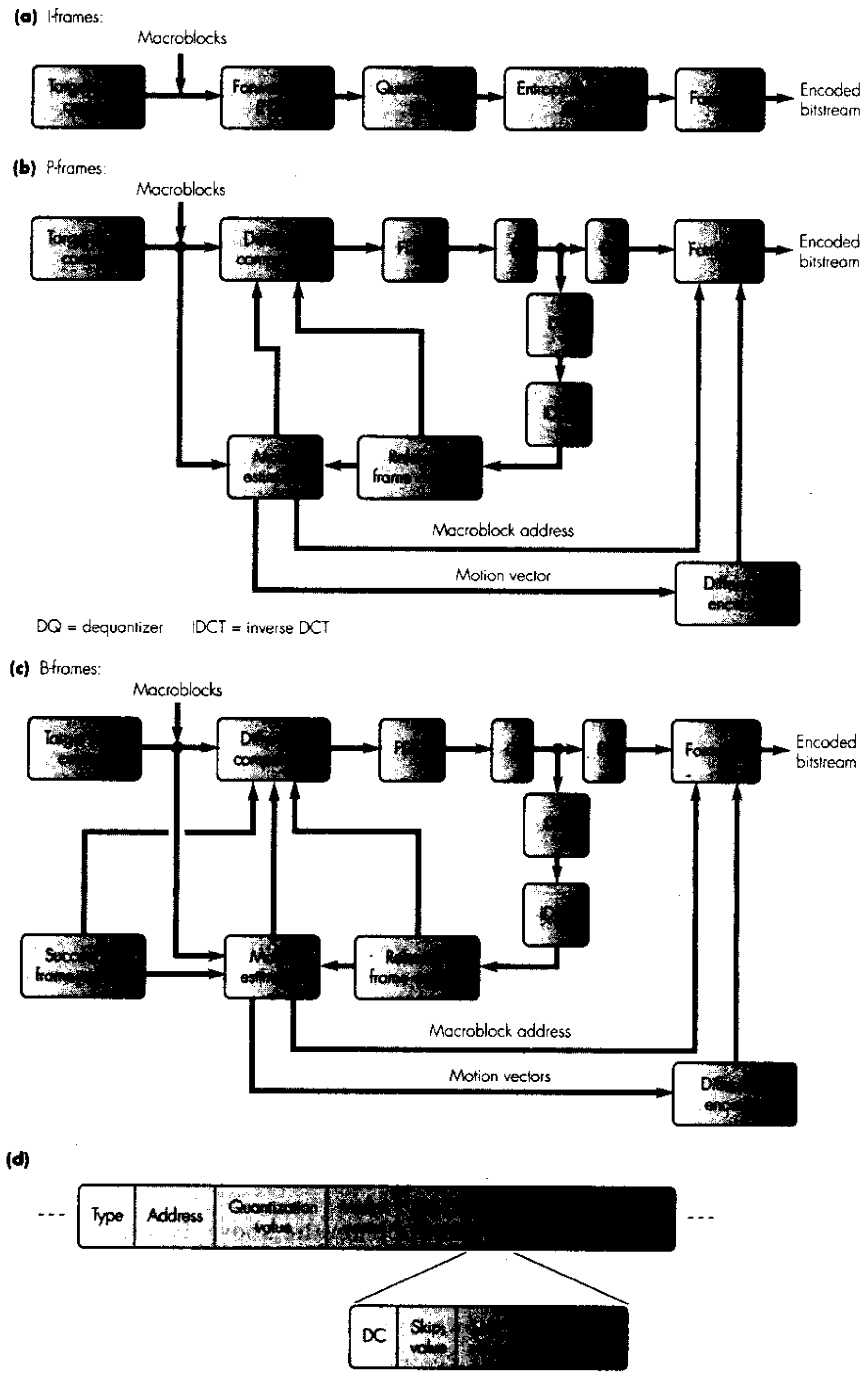


Figure 4.14 Implementation schematics: (a) I-frames; (b) P-frames; (c) B-frames; (d) example macroblock encoded bitstream format.

that has been used to quantize all the DCT coefficients in the macroblock and *motion vector* is the encoded vector if one is present. The ***blocks present*** indicates which of the six 8 × 8 pixel blocks that make up the macroblock are present – if any – and, for those present, the JPEG-encoded DCT coefficients for each block. As we can deduce from this, the amount of information output by the encoder varies and depends on the complexity of the source video. Hence a basic video encoder of the type shown in Figure 4.14 generates an encoded bitstream that has a variable bit rate.

The decoding of the received bitstream is simpler (and hence faster) than the encoding operation since the time-consuming motion estimation processing is not required. As the encoded bitstream is received and decoded, each new frame is assembled a macroblock at a time. Decoding the macroblocks of an I-frame is the same as decoding the blocks in a JPEG-encoded image. To decode a P-frame, the decoder retains a copy of the immediately preceding (decoded) I- or P- frame in a buffer and uses this, together with the received encoded information relating to each macroblock, to build the Y, C_b, and C_r matrices for the new frame in a second buffer. With uncoded macroblocks, the macroblock's address is used to locate the corresponding macroblock in the previous frame and its contents are then transferred into the second buffer unchanged. With fully-encoded macroblocks, these are decoded directly and the contents transferred to the buffer. With macroblocks containing a motion vector and a set of difference matrices, then these are used, together with the related set of matrices in the first buffer, to determine the new values for the macroblock in the second buffer. The decoding of B-frames is similar except three buffers are used: one containing the decoded contents of the preceding I- or P-frame, the second the decoded contents of the succeeding P- or I-frame, and the third to hold the frame being assembled.

Performance

The compression ratio for I-frames – and hence all intracoded frames – is similar to that obtained with JPEG and, for video frames, typically is in the region of 10:1 through 20:1 depending on the complexity of the frame contents. The compression ratio of both P- and B- frames is higher and depends on the search method used. However, typical figures are in the region of 20:1 through 30:1 for P-frames and 30:1 through 50:1 for B-frames.

4.3.2 H.261

The H.261 video compression standard has been defined by the ITU-T for the provision of video telephony and videoconferencing services over an integrated services digital network (ISDN). Hence, as we described earlier in Section 1.3.4, it is assumed that the network offers transmission channels of multiples of 64 kbps. The standard is also known, therefore, as ***p* × 64** where *p* can be 1 through 30. The digitization format used is either the common intermediate format (CIF) or the quarter CIF (QCIF). Normally, the CIF is used for videoconferencing and the QCIF for video telephony, both of which we described in Section 2.6.2. However, because each frame is divided into macroblocks of 16 × 16 pixels for compression, the horizontal resolution is reduced from 360 to 352 pixels to produce an integral number of 22 macroblocks. Hence, since both formats use subsampling of the two chrominance signals at half the rate used for the luminance signal, the spatial resolution of each format is:

$$\begin{aligned} \text{CIF: } Y &= 352 \times 288, \quad C_b = C_r = 176 \times 144 \\ \text{QCIF: } Y &= 176 \times 144, \quad C_b = C_r = 88 \times 72 \end{aligned}$$

Progressive (non-interlaced) scanning is used with a frame refresh rate of 30 fps for the CIF and either 15 or 7.5 fps for the QCIF.

Just I- and P- frames are used in H.261 with three P-frames between each pair of I-frames. The encoding of each of the six 8 × 8 pixel blocks that make up each macroblock in both I- and P-frames – 4 blocks for Y and one each for C_b and C_r – is carried out using the procedures described in the last section. The format of each encoded macroblock is shown in outline in Figure 4.15(a) and the format of each complete frame is shown in part (b) of the figure.

As we described in the last section, each macroblock has an *address* associated with it for identification purposes and the *type* field indicates whether the macroblock has been encoded independently – intracoded – or with reference to a macroblock in a preceding frame – intercoded. The *quantization value* is the threshold value that has been used to quantize all the DCT coefficients in the macroblock and *motion vector* is the encoded vector if one is present. The *coded block pattern* indicates which of the six 8 × 8 pixel blocks that make up the macroblock are present – if any – and, for those present, the JPEG-encoded DCT coefficients are given in each block.

The start of each new (encoded) video frame/picture is indicated by the *picture start code*. This is followed by a *temporal reference* field which is a timestamp to enable the decoder to synchronize each video block with an associated audio block containing the same time stamp. The *picture type* field indicates whether the frame is an I- or P-frame. Although the encoding operation is carried out on individual macroblocks, a larger data structure known as a *group of (macro) blocks (GOB)* is also defined. As we show in Figure 4.15(c), this is a matrix of 11 × 3 macroblocks, the size of which has been chosen so that both the CIF and QCIF comprise an integral number of GOBs – 12 in the case of the CIF (2 × 6) and 3 in the case of the QCIF (1 × 3) which allows interworking between the two formats. At the head of each GOB is a unique *start code* which is chosen so that no valid sequence of variable-length codewords from the table of codewords used in the entropy encoding stage can produce the same code. In the event of a transmission error affecting a GOB, the decoder simply searches the received bitstream for this code which signals the start of the next GOB. For this reason the *start code* is also known as a

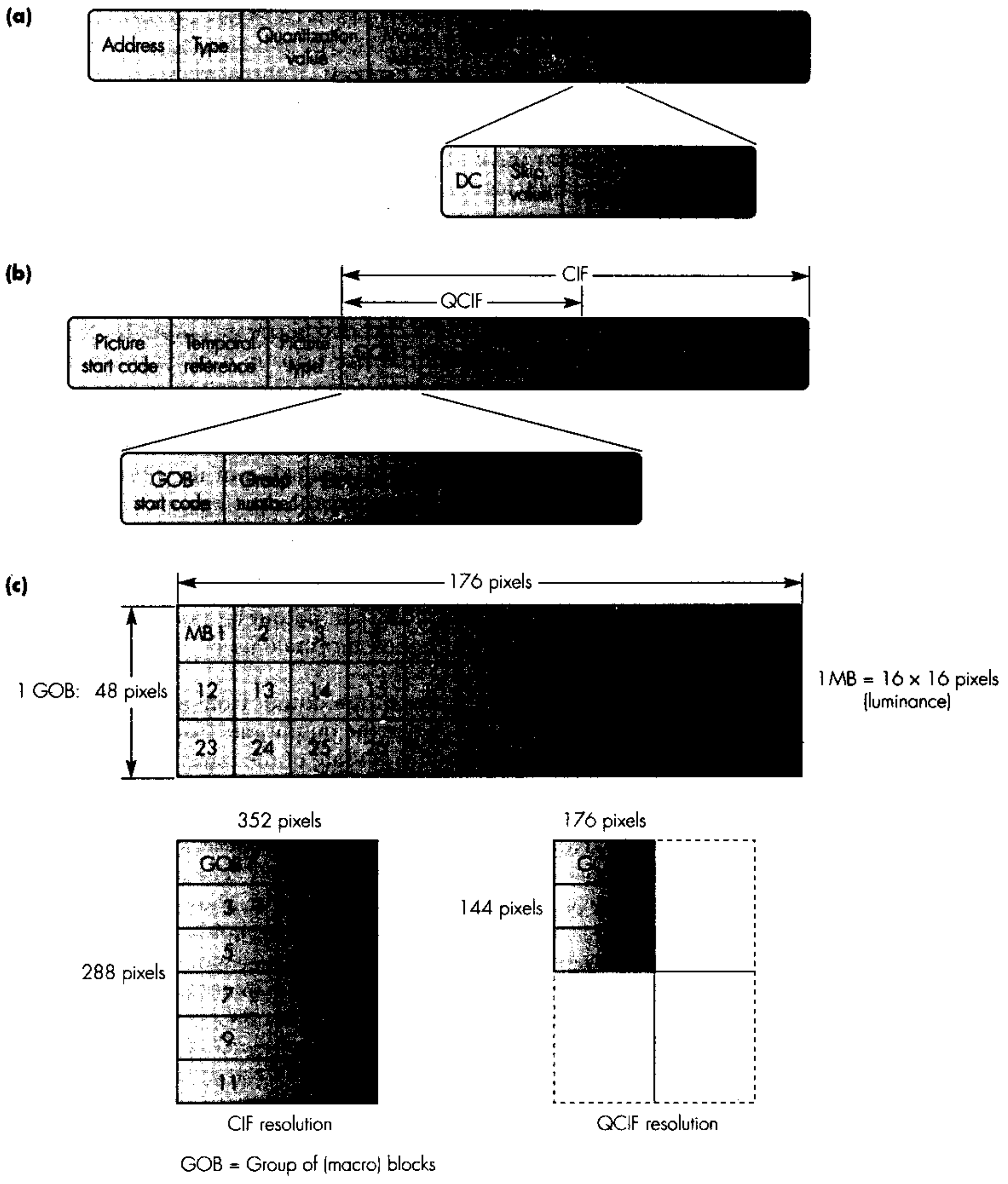


Figure 4.15 H.261 encoding formats: (a) macroblock format; (b) frame/picture format; (c) GOB structure.

resynchronization marker. In addition, each GOB has a *group number* associated with it which allows for a string of GOBs to be missing from a particular frame. This may be necessary, for example, if the amount of (compressed) information to be transmitted is temporarily greater than the bandwidth of the transmission channel.

As we indicated at the end of our earlier discussion on motion estimation, the amount of information produced during the compression operation varies. However, since the transmission bandwidth that is available with the target applications of the H.261 standard is fixed – 64 kbps or multiples of this – in order to optimize the use of this bandwidth, it is necessary to convert the variable bit rate produced by the basic encoder into a constant bit rate. This is achieved by first passing the encoded bitstream output by the encoder through a **first-in, first-out (FIFO) buffer** prior to it being transmitted and then providing a feedback path from this to the quantizer unit within the encoder. The general scheme is shown in Figure 4.16(a) and the role of the FIFO buffer is shown in Figure 4.16(b).

As you may recall from our earlier discussion of the JPEG standard in Section 3.4.5, the output bit rate produced by the encoder is determined by the quantization threshold values that are used; the higher the threshold, the lower the accuracy and hence the lower is the output bit rate. Hence, since the same compression technique is used for macroblocks in video encoders, it is possible to obtain a constant output bit rate from the encoder by dynamically varying the quantization threshold used. This is the role of the FIFO buffer.

As the name implies, the order of the output from a FIFO buffer is the same as that on input. However, since the output rate from the buffer is constant – determined by the (constant) bit rate of the transmission channel – if the input rate temporarily exceeds the output rate then the buffer will start to fill. Conversely, if the input rate falls below the output rate then the buffer contents will decrease. In order to exploit this property, two threshold levels are defined: the *low threshold* and the *high threshold*. The amount of information in the buffer is continuously monitored and, should the contents fall below the *low threshold*, then the quantization threshold is reduced – thereby increasing the output rate from the encoder. Conversely, should the contents increase beyond the *high threshold*, then the quantization threshold is increased in order to reduce the output rate from the encoder.

Normally, the control procedure operates at the GOB level rather than at the macroblock level. Hence, should the high threshold be reached, first the quantization value associated with the GOB is increased and, if this is not sufficient, GOBs are dropped until the overload subsides. Of course, any adjustments to the quantization threshold values that are made must be made also to those used in the matching dequantizer. In addition, the standard also allows complete frames to be missing in order to match the frame rate to the level of transmission bandwidth that is available.

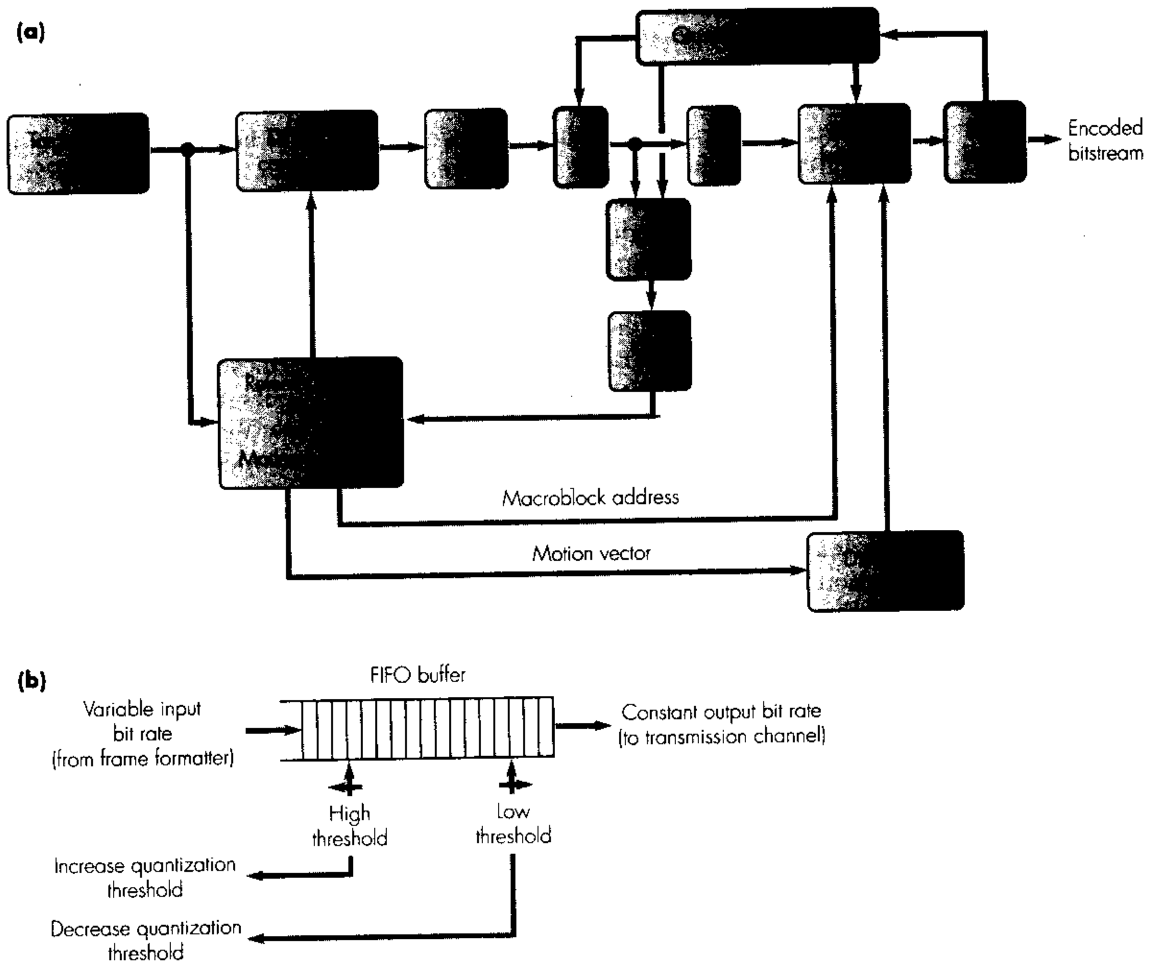

(a)
Encoded bitstream
Macroblock address
Motion vector
(b)
FIFO buffer
Variable input bit rate (from frame formatter)
Constant output bit rate (to transmission channel)
High threshold
Low threshold
Increase quantization threshold
Decrease quantization threshold

Figure 4.16 H.261 video encoder principles: (a) implementation schematic; (b) FIFO buffer operation.

4.3.3 H.263

The H.263 video compression standard has been defined by the ITU-T for use in a range of video applications over wireless and public switched telephone networks (PSTNs). The applications include video telephony, videoconferencing, security surveillance, interactive games playing, and so on, all of which require the output of the video encoder to be transmitted across the network connection in real time as it is output by the encoder. As we described in Section 1.3.1, the access circuit to the PSTN operates in an analog mode and to transmit a digital signal over these circuits, requires a modem. Typical maximum bit rates over switched connections range from 28.8 kbps through to 56 kbps and hence the requirement of the video encoder is to compress the video associated with these applications down to these very low bit rates.

The basic structure of the H.263 video encoder is based on that used in the H.261 standard. At bit rates lower than 64 kbps, however, the H.261 encoder gives a relatively poor picture quality. Since it uses only I- and P-frames, at low bit rates it has to revert to using a high quantization threshold and a relatively low frame rate. The high quantization threshold leads to what are known as **blocking artifacts** which are caused by the macroblocks encoded using high thresholds differing from those quantized using lower thresholds. The use of a low frame rate can also lead to very jerky movements. In order to minimize these effects, the H.263 standard has a number of advanced coding options compared with those used in an H.261 encoder. We shall limit our discussion of H.263 to an overview of a collection of these options.

Digitization formats

We described the various digitization formats associated with digital video in Section 2.6.2. In the case of the H.263 standard, the two mandatory formats are the QCIF and the sub-QCIF (S-QCIF). As with H.261, however, because each frame is divided into macroblocks of 16 × 16 pixels for compression, the horizontal resolution is reduced from 180 to 176 pixels to produce an integral number of (11) macroblocks. Hence, since subsampling of the two chrominance signals is used, the two alternative spatial resolutions are:

QCIF: $Y = 176 \times 144, \quad C_b = C_r = 88 \times 72$

S-QCIF: $Y = 128 \times 96, \quad C_b = C_r = 64 \times 68$

Progressive scanning is used with a frame refresh rate of either 15 or 7.5 fps.

In practice, the support of both formats is mandatory only for the decoder, and the encoder need support only one of them. As you may recall from Section 4.3.1, the motion estimation unit is not required in the decoder and hence is less expensive to implement than the encoder. The additional cost of having two alternative decoders is, therefore only small. However, by having a choice for the encoder, this means that either a simple – and hence low-cost – encoder design based on the S-QCIF can be used for applications such as games playing, or a more sophisticated design based on the QCIF can be used for applications such as videoconferencing. The decoder can be the same in both cases as it supports both formats.

Frame types

In order to obtain the higher levels of compression that are needed, the H.263 standard uses I-, P-, and B-frames. Also, in order to use as high a frame rate as possible, optionally, neighboring pairs of P- and B-frames can be encoded as a single entity. As we described in Section 4.3.1, the resulting encoded frame is known as a PB-frame and, because of the much reduced encoding overheads that are required, its use enables a higher frame rate to be used with a given transmission channel.

As we showed earlier in Figure 4.11(c), a PB-frame comprises a B-frame and the immediately succeeding P-frame. The encoded information for the corresponding macroblock in both these frames is interleaved, with the information for the P-frame preceding that of the B-frame. Hence at the decoder, as the encoded information is received, the macroblock for the P-frame is reconstructed first using the received information relating to the P-macroblock and the retained contents of the preceding P-frame. The contents of the reconstructed P-macroblock are then used, together with the received encoded information relating to the macroblock in the corresponding B-frame and the retained contents of the preceding P-frame, to bidirectionally predict the decoded contents of the B-macroblock. Then, when the decoding of both frames is complete, the B-frame is decoded first followed by the P-frame.

Unrestricted motion vectors

As we showed earlier in Figures 4.12 and 4.13, the motion vectors associated with predicted macroblocks are normally restricted to a defined area in the reference frame around the location in the target frame of the macroblock being encoded. In addition, the search area is restricted to the edge of the frame. This means that should a small portion of a potential close-match macroblock fall outside of the frame boundary, then the target macroblock is automatically encoded as for an I-frame. This occurs even though the portion of the macroblock within the frame area is a close match.

To overcome this limitation, in the unrestricted motion vector mode, for those pixels of a potential close-match macroblock that fall outside of the frame boundary, the edge pixels themselves are used instead and, should the resulting macroblock produce a close match, then the motion vector, if necessary, is allowed to point outside of the frame area. In practice, with the small digitized frame formats that are used with the H.263 standard, this has been found to give a significant improvement in the level of compression obtained.

Error resilience

As we indicated earlier, the target network for the H.263 standard is a wireless network or a PSTN. With this type of network there is a relatively high probability that transmission (bit) errors will be present in the bitstream received by the decoder. Normally, such errors are characterized by periods when a string of error-free frames is received followed by a short burst of errors which, typically, corrupt a string of macroblocks within a frame. In practice, it is not possible to identify the specific macroblocks that are corrupted but rather that the related group of (macro)blocks (GOB) contains one or more macroblocks that are in error. Also, as we can deduce from the frame sequences we showed earlier in Figure 4.11, because the contents of many frames are predicted from information in other frames, it is highly probable that the same GOB in each of the following frames that are derived from the GOB in error will contain errors. This means that when an error in a GOB occurs, the error will persist for a number of frames, hence making the error more apparent to the viewers.

As we explained in Section 4.3.2, when an error in a GOB is detected, the decoder skips the remaining macroblocks in the affected GOB and searches for the unique resynchronization marker (start code) at the head of the next GOB. It then recommences decoding from the start of this GOB. In order to mask the error from the viewer, an **error concealment scheme** is incorporated into the decoder. For example, a common approach is to use the contents of the corresponding GOB from the preceding (decoded) frame.

In addition, since a PSTN provides only a relatively low bit rate transmission channel, to conserve bandwidth, intracoded (I) frames are inserted at relatively infrequent intervals. Hence in applications such as video telephony in which the video and audio are being transmitted in real time, the lack of I-frames has the effect that errors within a GOB may propagate to other regions of the frame due to the resulting errors in the motion estimation vectors and motion compensation information. With digitization formats such as the QCIF (which has only three GOBs per frame) the resulting effect can be very annoying to the viewer. A typical effect is shown in diagrammatic form in Figure 4.17(a) and, as we can see, although the initial error occurs in one GOB position, it rapidly spreads to other neighbouring GOBs. It is for this reason that schemes are included in the standard that aim at minimizing the effects of errors on neighbouring GOBs. The schemes include **error tracking**, **independent segment decoding**, and **reference picture selection.** We shall discuss the principle of operation of each of these schemes separately.

Error tracking With real-time applications such as video telephony, a two-way communications channel is required for the exchange of the compressed audio and video information generated by the codec in each terminal. This means that there is always a return channel from the receiving terminal back to the sending terminal and this is used in all three schemes by the decoder in order to inform the related encoder that an error in a GOB has been detected. Typically, errors are detected in a number of ways including:

- one or more out-of-range motion vectors,
- one or more invalid variable-length codewords,
- one or more out-of-range DCT coefficients,
- an excessive number of coefficients within a macroblock.

In the error tracking scheme, the encoder retains what is known as error prediction information for all the GOBs in each of the most recently transmitted frames; that is, the likely spatial and temporal effects on the macroblocks in

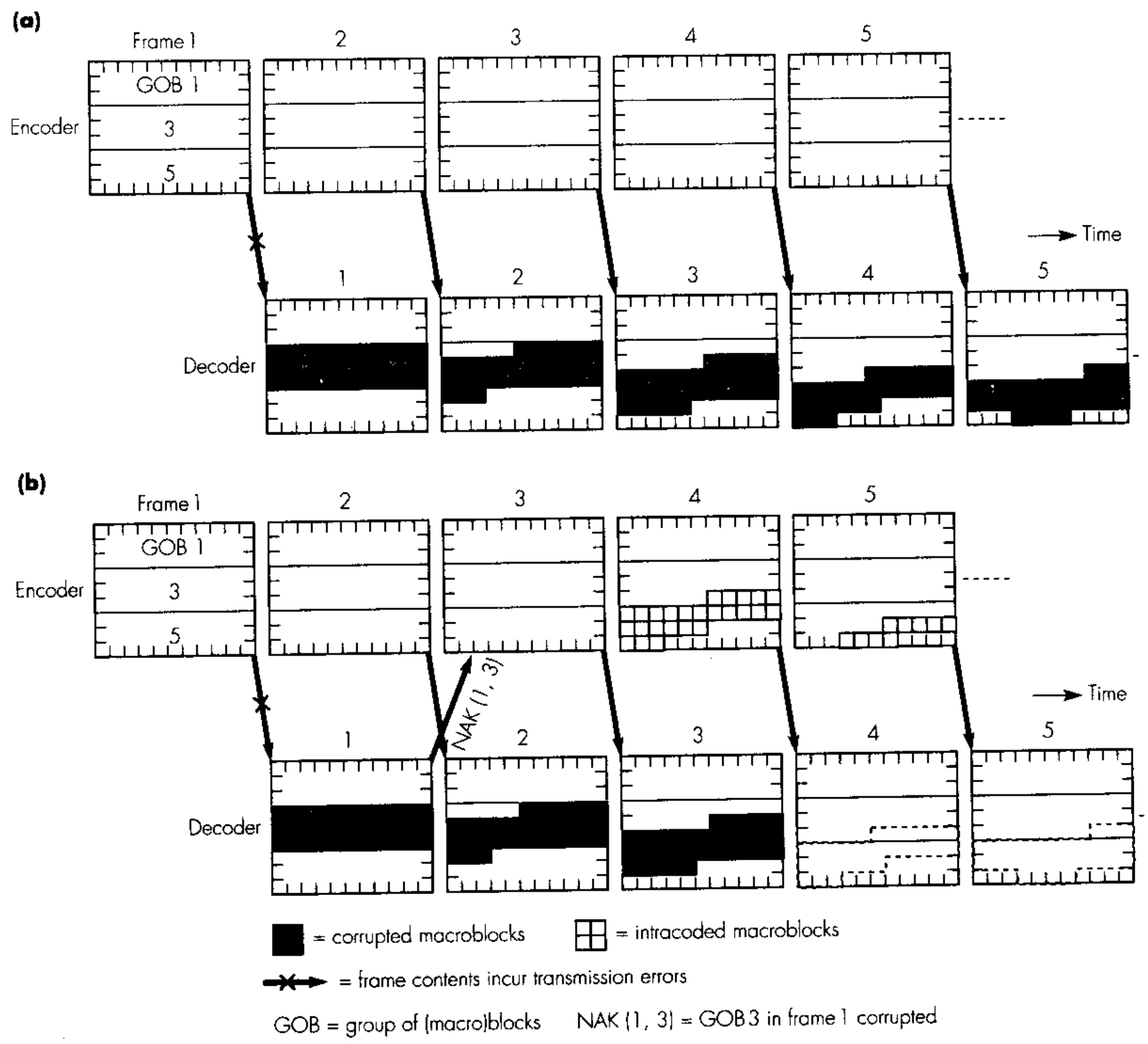

(a)
Frame 1
2
3
4
5
GOB 1
3
5
Encoder
Decoder
Time
(b)
NAK (1, 3)
= corrupted macroblocks
= intracoded macroblocks
= frame contents incur transmission errors
GOB = group of (macro)blocks
NAK (1, 3) = GOB 3 in frame 1 corrupted

Figure 4.17 H.263 error tracking scheme: (a) example error propagation; (b) same example with error tracking applied.

the following frames that will result if a specific GOB in a frame is corrupted. When an error is detected, the return channel is used by the decoder to send a negative acknowledgment (NAK) message back to the encoder in the source codec containing both the frame number and the location of the GOB in the frame that is in error. The encoder then uses the error prediction information relating to this GOB to identify the macroblocks in those GOBs in later frames that are likely to be affected. It then proceeds to transmit the macroblocks in these frames in their intracoded form. The principle of the scheme is shown in Figure 4.17(b).

The example frame sequence shows in diagrammatic form how the corrupted macroblocks shown in part(a) of the figure are predicted and, for each affected frame, the predicted macroblocks are sent in their intracoded form. Hence in the example, on receipt of NAK(1,3), it is assumed that the encoder has predicted and retained error prediction information relating to frame 1. Since the next frame to be encoded is frame 4, the affected macroblocks are intracoded rather than predicted. Similarly for the affected macroblocks in frame 5.

Independent segment decoding The aim of this scheme is not to overcome errors that occur within a GOB but rather to prevent these errors from affecting neighboring GOBs in succeeding frames. To achieve this, each GOB is treated as a separate subvideo which is independent of the other GOBs in the frame. This means, therefore, that motion estimation and compensation is limited to the boundary pixels of a GOB rather than a frame. The operation of the scheme is shown in parts (a) and (b) of Figure 4.18.

As we can see from part (a), although when an error in a GOB occurs the same GOB in each successive frame is affected – until a new intracoded GOB is sent by the encoder – neighbouring GOBs are not affected. Clearly, however, a limitation of this scheme is that the efficiency of the motion estimation and compensation in the vertical direction is reduced significantly owing to the search area being limited to a single GOB. Thus, the scheme is not normally used on its own but in conjunction with either the error tracking scheme – as we show in Figure 4.18(b) – or, more usually, with the reference picture selection scheme described below.

Reference picture selection This scheme is similar to the error tracking scheme inasmuch as it endeavors to stop errors propagating by the decoder returning acknowledgment messages when an error in a GOB is detected. The scheme can be operated in two different modes as we show in parts(a) and (b) of Figure 4.19.

The mode of operation we show in part (a) is known as the *NAK mode* and, in this mode, only GOBs in error are signaled by the decoder returning a NAK message. As we showed earlier in Figure 4.11, normally intercoded (P or B) frames are encoded using an intracoded (I) frame as the initial reference frame. However, as we showed in Figure 4.14, during the encoding of intercoded frames a copy of the (decoded) preceding frame is retained by the encoder. With this scheme, therefore, the encoder can select any of these previously decoded frames as the reference. Hence in the example shown in Figure 4.19(a), when the NAK relating to frame 2 is received, the encoder selects (the decoded) GOB 3 of frame 1 as the reference to encode GOB 3 of the next frame – frame 5 in this example.

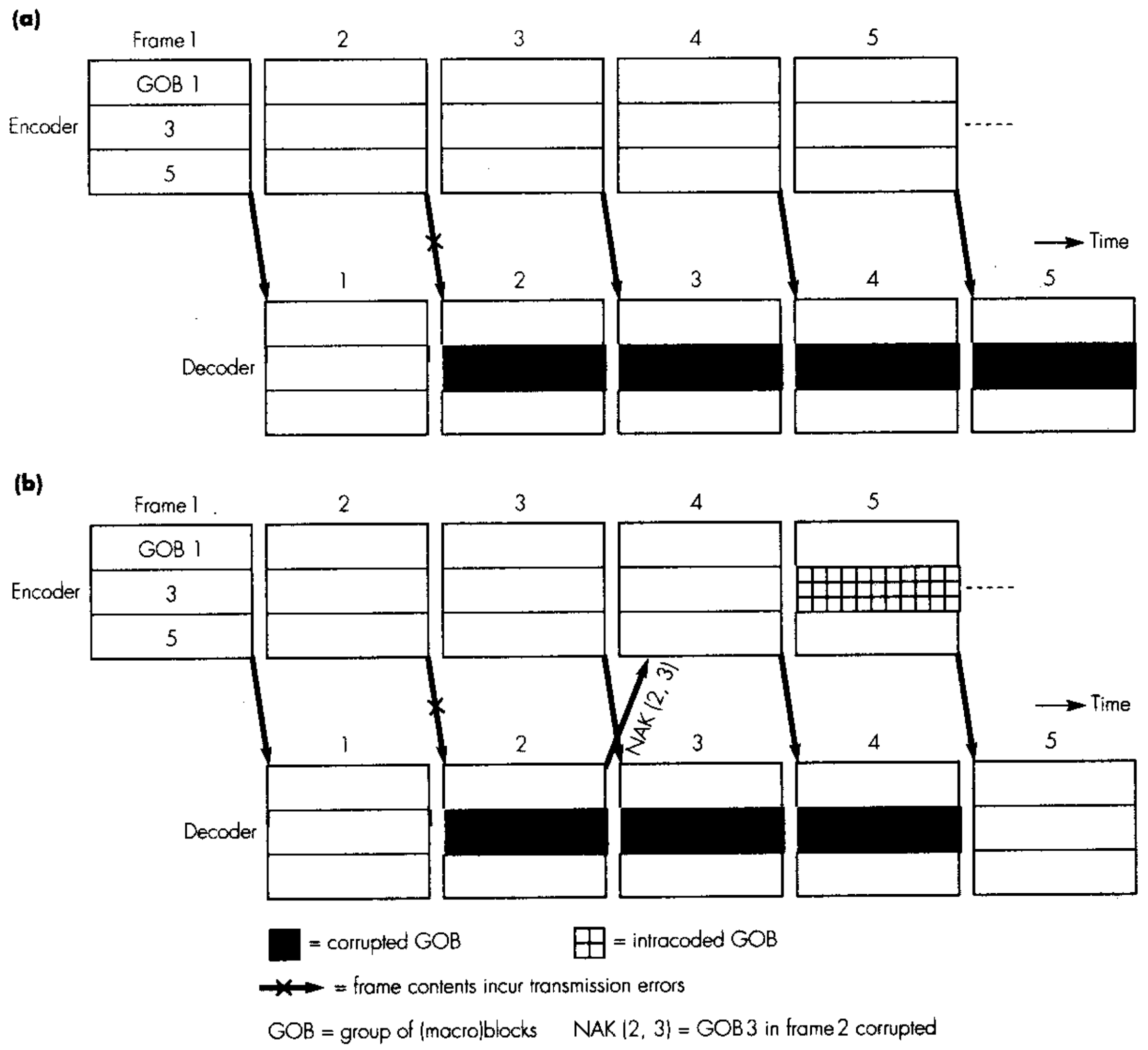


Figure 4.18 Independent segment decoding: (a) effect of a GOB being corrupted; (b) when used with error tracking.

As we can see in the figure, with this scheme the GOB in error will propagate for a number of frames, the number being determined by the round-trip delay of the communications channel; that is, the time delay between the NAK being sent by the decoder and an intercoded frame derived from the initial I-frame being received.

The alternative mode of operation is known as the *ACK mode* since, as we show in Figure 4.19(b), with this mode all frames received without errors are

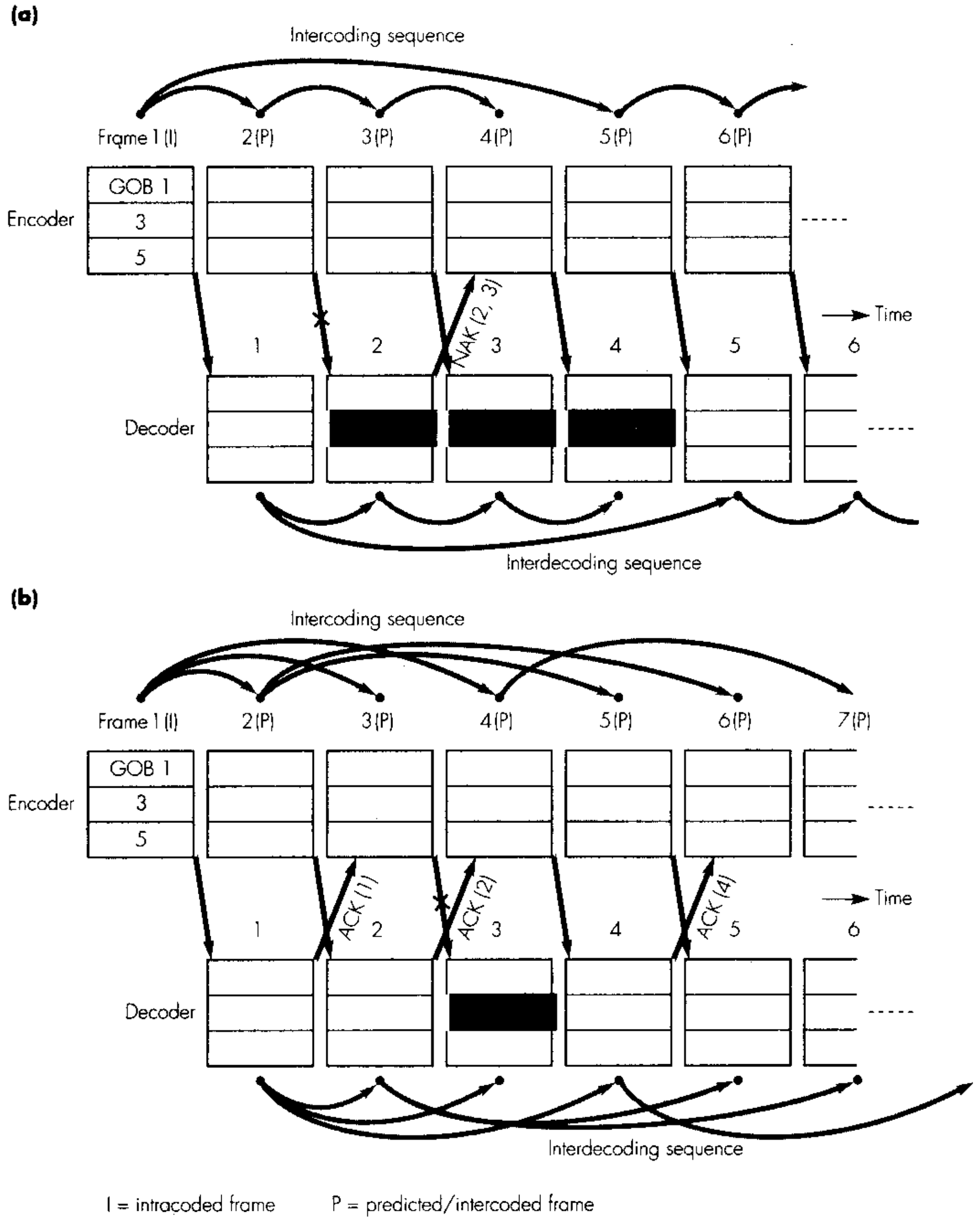


Figure 4.19 Reference picture selection with independent segment decoding: (a) NAK mode; (b) ACK mode.

acknowledged by the decoder returning an ACK message. Only frames that have been acknowledged are used as reference frames. Hence, in the example, the lack of an ACK for frame 3 means that frame 2 must be used to encode frame 6 in addition to frame 5. At this point the ACK for frame 4 is received and hence the encoder then uses this to encode frame 7. The effect of using a reference frame which is distant (in time) from the frame being encoded is to reduce the encoding efficiency for the frame. Thus the ACK mode performs best when the round-trip delay of the communications channel is short and, ideally, less than the time the encoder takes to encode each frame.

4.3.4 MPEG

The **Motion Pictures Expert Group (MPEG)** was formed by the ISO to formulate a set of standards relating to a range of multimedia applications that involve the use of video with sound. The outcome is a set of three standards which relate to either the recording or the transmission of integrated audio and video streams. Each is targeted at a particular application domain and describes how the audio and video are compressed and integrated together. The three standards, which use different video resolutions, are:

- **MPEG-1:** This is defined in a series of documents which are all subsets of **ISO Recommendation 11172**. The video resolution is based on the source intermediate digitization format (SIF) with a resolution of up to 352 × 288 pixels. The standard is intended for the storage of VHS-quality audio and video on CD-ROM at bit rates up to 1.5 Mbps. Normally, however, higher bit rates of multiples of this are more common in order to provide faster access to the stored material.
- **MPEG-2:** This is defined in a series of documents which are all subsets of **ISO Recommendation 13818**. It is intended for the recording and transmission of studio-quality audio and video. The standard covers four levels of video resolution:
  - Low: based on the SIF digitization format with a resolution of up to 352 × 288 pixels. It is compatible with the MPEG-1 standard and produces VHS-quality video. The audio is of CD quality and the target bit rate is up to 4 Mbps;
  - Main: based on the 4:2:0 digitization format with a resolution of up to 720 × 576 pixels. This produces studio-quality digital video and the audio allows for multiple CD-quality audio channels. The target bit rate is up to 15 Mbps or 20 Mbps with the 4:2:2 digitization format;
  - High 1440: based on the 4:2:0 digitization format with a resolution of 1440 × 1152 pixels. It is intended for high-definition television (HDTV) at bit rates up to 60 Mbps or 80 Mbps with the 4:2:2 format.
  - High: based on the 4:2:0 digitization format with a resolution of 1920 × 1152 pixels. It is intended for wide-screen HDTV at a bit rate of up to 80 Mbps or 100 Mbps with the 4:2:2 format.

- **MPEG-4:** Initially, this standard was concerned with a similar range of applications to those of H.263, each running over very low bit rate channels ranging from 4.8 to 64 kbps. Later its scope was expanded to embrace a wide range of interactive multimedia applications over the Internet and the various types of entertainment networks.

Unlike these three standards which are concerned with the compression of the multimedia information associated with an application, the **MPEG-7** standard is concerned with describing the structure and features of the content of the (compressed) multimedia information produced by the different standards. The resulting descriptions are then used in **search engines** to locate particular items of material that have a defined feature.

The MPEG-3 standard was to be focussed on HDTV but was not developed separately as the work was subsequently incorporated into MPEG-2. Hence we shall restrict our discussion to MPEG-1, the main and high levels of MPEG-2, and selected features of the MPEG-4 standard. MPEG-7 is outside of the scope of the book.

The first three MPEG standards are in three parts: video, audio, and system. The video and audio parts are concerned with the way each is compressed and how the resulting bitstreams are formatted. The system part is concerned with how the two streams are integrated together to produce a synchronized output stream. We shall discuss just the video compression part of the MPEG-1/2 standards in this chapter and the system parts in the next chapter.

4.3.5 MPEG-1

The MPEG-1 (and MPEG-2) video standard uses a similar video compression technique as H.261. As indicated, the digitization format used with MPEG-1 is the source intermediate format (SIF) which we described in Section 2.6.2. However, as with the H.261 standard, because each frame is divided into macroblocks of 16 × 16 pixels for compression, the horizontal resolution is reduced from 360 to 352 pixels to produce an integral number of (22) macroblocks. Hence, since the two chrominance signals are subsampled at half the rate of the luminance signal, the spatial resolutions for the two types of video source are:

NTSC:	$Y = 352 \times 240,\quad C_b = C_r = 176 \times 120$
PAL:	$Y = 352 \times 288,\quad C_b = C_r = 176 \times 144$

Also, since the application domain of MPEG-1 is for the storage of video (with audio), progressive scanning is used with a refresh rate of 30 Hz (for NTSC) and 25 Hz (for PAL).

The standard allows the use of I-frames only, I- and P-frames only, or I-, P- and B-frames, the latter being the most common. No D-frames are supported in any of the MPEG standards and hence, in the case of MPEG-1, I-frames must be used for the various random-access functions associated with VCRs. The accepted maximum random-access time is 0.5 seconds and so this is the main factor – along with video quality – that influences the maximum separation of I-frames in the frame sequence used. Two example sequences are:

IBBPBBPBBI... and IBBPBBPBBPBBI...

the first being the original sequence proposed for use with PAL (which has a slower frame refresh rate) and the second for use with NTSC. The second standard is now used with both systems and is illustrated in Figure 4.20.

Example 4.1

An MPEG-1 system uses the frame sequence shown in Figure 4.20.

(i) Define the terms *M* and *N* and hence determine their values for the sequence shown in the figure.

(ii) Derive a suitable reordered sequence that ensures firstly, only two frames must be stored in the decoder, and secondly, the required I- and/or P-frames are available to decode each P- and B-frame as they are received.

Answer:

(i) As we described earlier in Section 4.3.1 under the subheading of "Frame types", *M* is the distance (in frames) between a P-frame and the immediately preceding I- or P- frame, and *N* is the number of frames between two successive I-frames. The latter is known as a group of pictures or GOP. Hence for the frame sequence shown in Figure 4.20, *M* = 3 and *N* = 12.

(ii) A suitable reordered frame sequence that meets the defined requirements is:

IPBBPBBPBBIBBPBB ...

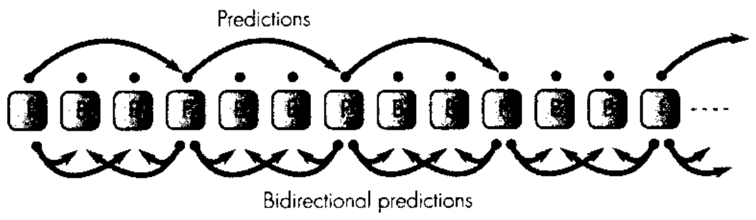


Figure 4.20 MPEG-1 example frame sequence.

The compression algorithm used in MPEG-1 is based on the H.261 standard. Hence each macroblock is made up of 16×16 pixels in the Y plane and 8×8 pixels in the C_b and C_r planes. However, there are two main differences. The first is that time-stamps (temporal references) can be inserted within a frame to enable the decoder to resynchronize more quickly in the event of one or more corrupted or missing macroblocks. The number of macroblocks between two time-stamps is known as a *slice* and a slice can comprise from 1 through to the maximum number of macroblocks in a frame. Typically, a slice is made equal to 22 which is the number of macroblocks in a line.

The second difference arises because of the introduction of B-frames which increases the time interval between I and P frames. To allow for the resulting increase in the separation of moving objects with P-frames, the search window in the reference frame is increased. Also, to improve the accuracy of the motion vectors, a finer resolution is used. Typical compression ratios vary from about 10:1 for I-frames, 20:1 for P-frames and 50:1 for B-frames.

As with the H.261 standard, the compressed bitstream produced by the video encoder is hierarchical and is shown in Figure 4.21(a). At the top level, the complete compressed video is known as a **sequence** which, in turn, con-

Example 4.2

A digitized video is to be compressed using the MPEG-1 standard. Assuming a frame sequence of:

IBBPBBPBBPBBI...

and average compression ratios of 10:1 (I), 20:1 (P) and 50:1 (B), derive the average bit rate that is generated by the encoder for both the NTSC and PAL digitization formats.

Answer:

Frame sequence = IBBPBBPBBPBBI...

Hence: 1/12 of frames are I-frames, 3/12 are P-frames, and 8/12 are B-frames.

and Average compression ratio $= (1 \times 0.1 + 3 \times 0.05 + 8 \times 0.02)/12$
$= 0.0342$ or 29.24:1

NTSC frame size:

Without compression $= 352 \times 240 \times 8 + 2\ (176 \times 120 \times 8)$
$= 1.013760$ Mbits per frame

With compression $= 1.01376 \times 1/29.24$
$= 34.670$ kbits per frame

Hence bit rate generated at 30 fps = 1.040 Mbps

4.2 Continued

PAL frame size:

Without compression = 352 × 288 × 8 + 2 (176 × 144 × 8)
= 1.216512 Mbits per frame

With compression = 1.216512 × 1/29.24
= 41.604 kbits per frame

Hence bit rate generated at 25 fps = 1.040 Mbps

Normally, allowing for packetization and multiplexing overheads, a bandwidth of 1.2 Mbps is allocated for the video. Hence, assuming a maximum bit rate of 1.5 Mbps, this leaves 300 kbps for the compressed audio stream.

sists of a string of **groups of pictures (GOPs)** each comprising a string of I, P or B pictures/frames in the defined sequence. Each picture/frame is made up of N **slices**, each of which comprises multiple macroblocks, and so on down to an 8 × 8 pixel block. Hence in order for the decoder to decompress the received bitstream, each data structure must be clearly identified within the bitstream. The format of the bitstream is shown in Figure 4.21(b).

The start of a sequence is indicated by a *sequence start code*. This is followed by three parameters, each of which applies to the complete video sequence. The *video parameters* specify the screen size and aspect ratio, the *bitstream parameters* indicate the bit rate and the size of the memory/frame buffers that are required, and the *quantization parameters* contain the contents of the quantization tables that are to be used for the various frame/picture types. These are followed by the encoded video stream which, as we can see, is in the form of a string of GOPs.

Each GOP (IBBP...) is separated by a *(GOP) start code* which is followed by a *time-stamp* for synchronization purposes and a *parameters* field which defines the particular sequence of frame types that are used in each GOP. This is then followed by the string of encoded pictures/frames in each GOP. Each is separated by a *picture start code* and is followed by a *type* field (I, P or B), *buffer parameters*, which indicate how full the memory buffer should be before the decoding operation should start, and *encode parameters* which indicate the resolution used for the motion vectors. This is followed by a string of slices, each comprising a string of macroblocks. Each slice is separated by a *slice start code* which is followed by a *vertical position* field, which defines the scan line the slice relates to, and a *quantization parameter* that indicates the scaling factor that applies to this slice. This is then followed by a string of macroblocks each of which is encoded in the same way as for H.261, the basic principles of which we described earlier in Section 4.3.1. As we can deduce from this, a slice is similar to the GOB used in the H.261 standard.

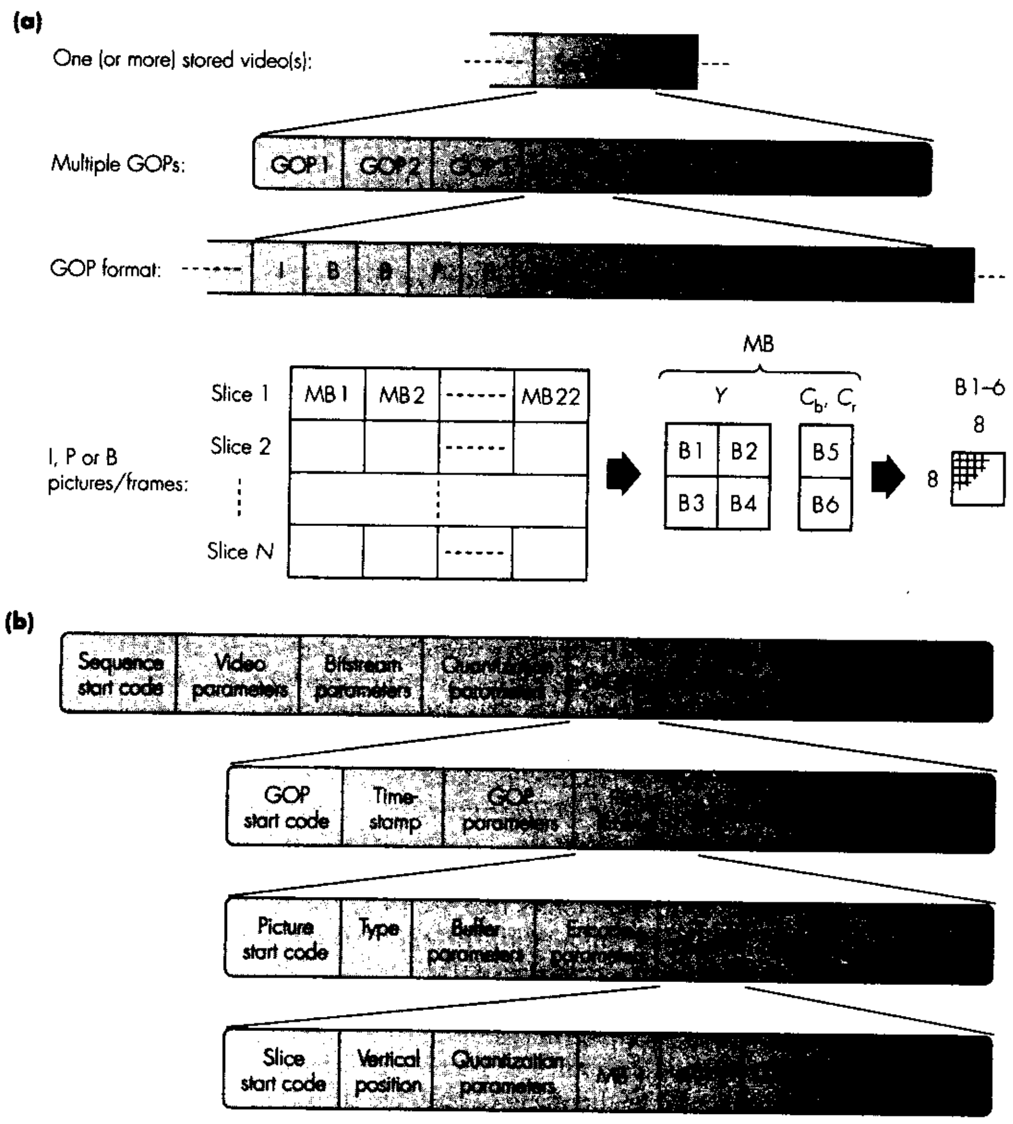


Figure 4.21 MPEG-1 video bitstream structure: (a) composition; (b) format.

4.3.6 MPEG-2

As we indicated in Section 4.3.4, MPEG-2 supports four levels – low, main, high 1440 and high – each targeted at a particular application domain. In addition, there are five **profiles** associated with each level: simple, main, spatial resolution, quantization accuracy, and high. These have been defined so that the four levels and five profiles collectively form a two-dimensional table which acts as a framework for all standards activities associated with MPEG-2. In this way the development of both the existing standards and any new standards can take place relative to one another. For example, at a particular level, the decoders used with a given profile will be able to decode all the lower profiles that have been defined for that level, hence making interworking between older and newer equipments possible. As indicated earlier, since the low level of MPEG-2 is compatible with MPEG-1, we shall restrict our discussion of MPEG-2 to the **main profile at the main level (MP@ML)** and the two high levels relating to HDTV.

MP@ML

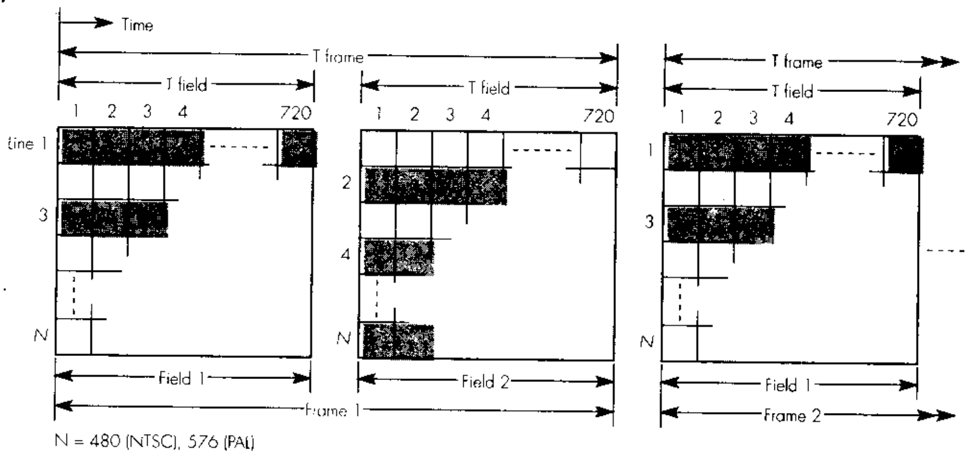
The target application of the MP@ML standard is for digital television broadcasting. Hence interlaced scanning is used – which we described in Section 2.6.1 – with a resulting frame refresh rate of either 30 Hz (NTSC) or 25 Hz (PAL). The 4:2:0 digitization format is used with a resolution of either 720 × 480 pixels at 30 Hz or 720 × 576 pixels at 25 Hz. The output bit rate from the system multiplexer can range from 4 Mbps through to 15 Mbps, the actual rate used being determined by the bandwidth available with the broadcast channel.

The video coding scheme is similar to that used in MPEG-1 the main difference resulting from the use of interlaced – instead of progressive – scanning. The use of interlaced scanning means that each frame is made up of two (interlaced) fields and, as we show in Figure 4.22(a), alternative lines are present in each field. Hence for I-frames, the question arises as to how the DCT blocks are derived from each macroblock. As we show in parts (b) and (c) of the figure, two alternatives are possible depending on whether the DCT blocks are derived from the lines in a field – **field mode** – or the lines in a frame – **frame mode**.

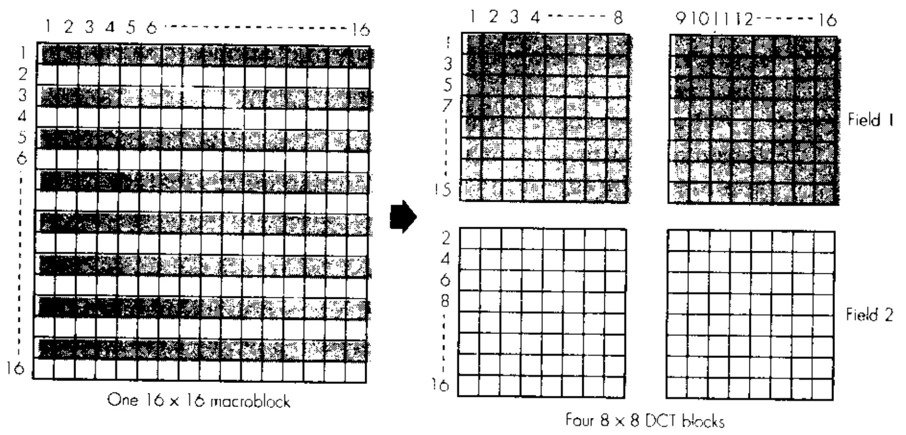
As described in Section 2.6.1, for a frame refresh rate – temporal resolution – of 30/25 Hz, the corresponding field refresh rate is 60/50 Hz. The choice of field or frame mode is thus determined by the amount of motion present in the video. If a large amount of motion is present, then it is better to perform the DCT encoding operation on the lines in a field – part (b) – since this will produce a higher compression ratio owing to the shorter time interval between successive fields. Alternatively, if there is little movement, the frame mode can be used since the longer time interval between successive (complete) frames is less important. Hence in this case, the macroblocks/DCT blocks are derived from the lines in each complete frame – part (c). In fact, the standard allows either mode to be used, the choice being determined by the type of video; for example, a live sporting event is likely to be encoded using the field mode and a studio-based program the frame mode.

In relation to the motion estimation associated with the encoding of macroblocks in P- and B-frames, three different modes are possible: field, frame, and mixed. In the field mode, the motion vector for each macroblock

(a)



(b)



(c)

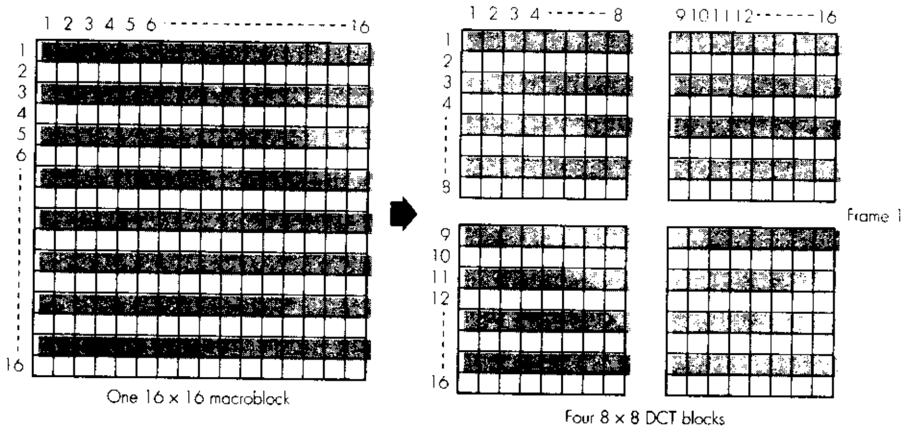


Figure 4.22 MPEG-2 DCT block derivation with I-frames: (a) effect of interlaced scanning; (b) field mode; (c) frame mode.

is computed using the search window around the corresponding macroblock in the immediately preceding (I or P) field for both P- and B-frames and, for B-frames, the immediately succeeding (P or I) field. The motion vectors, therefore, relate to the amount of movement that has taken place in the time to scan one field. In the frame mode, a macroblock in an odd field is encoded relative to that in the preceding/succeeding odd field(s) and similarly for the macroblocks in even fields. In this case, therefore, the motion vectors relate to the amount of movement that has taken place in the time to scan two fields; that is, the time to scan a complete frame. In the mixed mode, the motion vectors for both field and frame modes are computed and the one with the smallest (mean) values is selected.

HDTV

There are three standards associated with HDTV: **advanced television (ATV)** in North America, **digital video broadcast (DVB)** in Europe, and **multiple sub-Nyquist sampling encoding (MUSE)** in Japan and the rest of Asia. All three standards, in addition to defining the digitization format and audio and video compression schemes used, also define how the resulting digital bitstreams are transmitted over the different types of broadcast network.

In addition to these standards, as with normal digital television, there is an **ITU-R HDTV specification** concerned with the standards used in television studios for the production of HDTV programs and also for the international exchange of programs. This defines a 16/9 aspect ratio with 1920 samples per line and 1152 (1080 visible) lines per frame. Currently, interlaced scanning is used with the 4:2:0 digitization format. In the future, however, it is expected that progressive scanning will be introduced with the 4:2:2 format.

The ATV standard, which was formulated by an alliance of a large number of television manufacturers, is also known as the **Grand Alliance** (or simply **GA**) **standard**. It includes the ITU-R HDTV specification and a second, lower-resolution format. This also uses a 16/9 aspect ratio but with a resolution of 1280 × 720. The video compression algorithm in both cases is based on the main profile at the high level (MP@HL) of MPEG-2 and the audio compression standard is based on Dolby AC-3.

The DVB HDTV standard is based on the 4/3 aspect ratio and defines a resolution of 1440 samples per line and 1152 (1080 visible) lines per frame. As we can see, this is exactly twice the resolution of the low-definition PAL digitization format of 720 × 576. The video compression algorithm is based on what is known as the **SSP@H1440 – spatially-scaleable profile at high 1440** – of MPEG-2 which is very similar to that used with MP@HL. The audio compression standard is MPEG Audio layer 2.

The MUSE standard is based on a 16/9 aspect ratio with a digitization format of 1920 samples per line and 1035 lines per frame. The video compression algorithm is similar to that used in MP@HL.

4.3.7 MPEG-4

The main application domain of the MPEG-4 standard is in relation to the audio and video associated with interactive multimedia applications over the Internet and the various types of entertainment networks. The standard contains features to enable a user not only to passively access a video sequence (or complete video) – using, for example, start/stop/pause commands – but also to access and manipulate the individual elements that make up each scene within the sequence/video. If the accessed video is a computer-generated cartoon, for example, the user may be given the capability – by the creator of the video – to reposition, delete, or alter the movement of the individual characters within a scene. In addition, because of its high coding efficiency with scenes such as those associated with video telephony, the standard is also used for this type of application running over low bit rate networks such as wireless and PSTNs. For such applications, therefore, it is an alternative to the H.263 standard.

Scene composition

The main difference between MPEG-4 and the other standards we have considered is that MPEG-4 has a number of what are called **content-based functionalities**. Before being compressed each scene is defined in the form of a background and one or more foreground **audio-visual objects (AVOs)**. Each AVO is in turn defined in the form of one or more **video objects** and/or **audio objects**; for example, a stationary car in a scene may be defined using just a single video object while a person who is talking may be defined using both an audio and a video object. In a similar way, each video and audio object may itself be defined as being made up of a number of subobjects. For example, if a person's eyes and mouth are the only things that move, in order to exploit this, the person's face may be defined in the form of three subobjects: one for the head and the other two for the eyes and mouth. Once this has been done, the encoding of the background and each AVO is carried out separately and, in the case of AVOs consisting of both audio and video objects, each has additional timing information relating to it to enable the receiver to synchronize the various objects and subobjects together before they are decoded.

Each audio and video object has a separate **object descriptor** associated with it which allows the object – providing the creator of the audio and/or video has provided the facility – to be manipulated by the viewer prior to it being decoded and played out. The language used to describe and modify objects is called the **binary format for scenes (BIFS)**. This has commands to delete an object and, in the case of a video object, change its shape, appearance – color for example – and, if appropriate, animate the object in real time. Audio objects have a similar set of commands, to change its volume for example. It is also possible to have multiple versions of an AVO, the first containing the base-level compressed audio and video streams and the others various levels of enhancement detail. This type of compression is called scaling and allows the encoded contents of an AVO to be played out at a rate and resolution that matches those of the interactive terminal being used.

At a higher level, the composition of a scene in terms of the AVOs it contains is defined in a separate **scene descriptor**. This defines the way the various AVOs are related to each other in the context of the complete scene; for example, the position of each AVO on the display screen and the composition of each AVO in terms of the objects and subobjects it comprises.

A simple example illustrating how a frame/scene is defined in the form of a number of AVOs using content-based image coding is shown in Figure 4.23. As we can see, each video frame is segmented into a number of **video object planes (VOPs)** each of which corresponds to an AVO of interest. Hence in our simple example, the frame is shown as consisting of three VOPs: VOP 0 to represent the person approaching the car, VOP 1 the remainder of the car parked outside the house, and VOP 2 the remainder of the background. Each VOP is encoded separately based on its shape, motion, and texture.

As we show in the figure, each VOP is encapsulated within a rectangle which, in practice, is chosen so that it completely covers the related AVO using the minimum number of macroblocks. The spatial coordinates of the VOP are then relative to the coordinates of the top-left macroblock. The motion and texture of each VOP are encoded using a procedure similar, for example, to one of those we described earlier in the chapter. The resulting

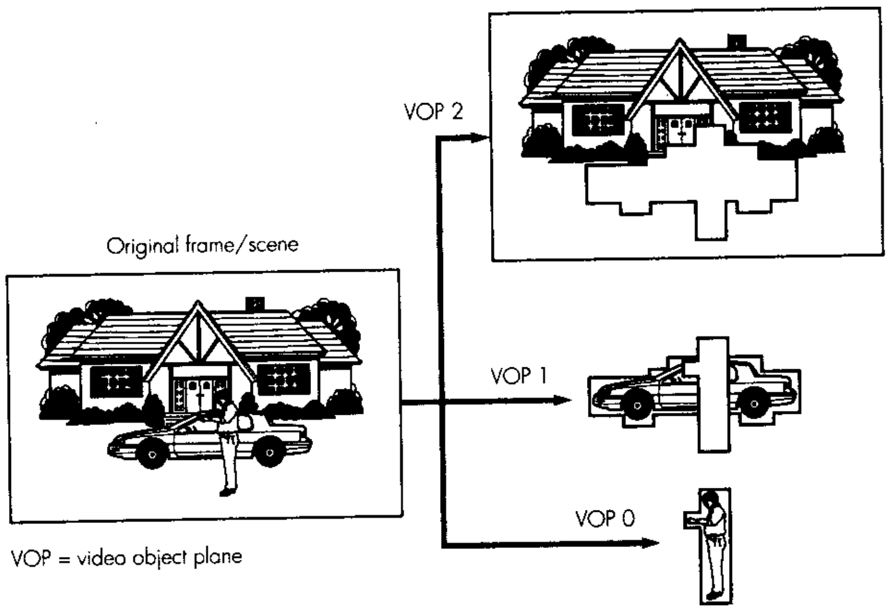


Figure 4.23 Content-based video coding principles showing how a frame/scene is defined in the form of multiple video object planes.

bitstreams are then multiplexed together for transmission together with the related object and scene descriptor information. Similarly, at the receiver, the received bitstream is first demultiplexed and the individual streams decoded. The decompressed information, together with the object and scene descriptions, are then used to create the video frame that is played out on the terminal. Typically, the latter is a multimedia PC or TV set and the audio relating to each scene is played out in time synchronism with the video.

Audio and video compression

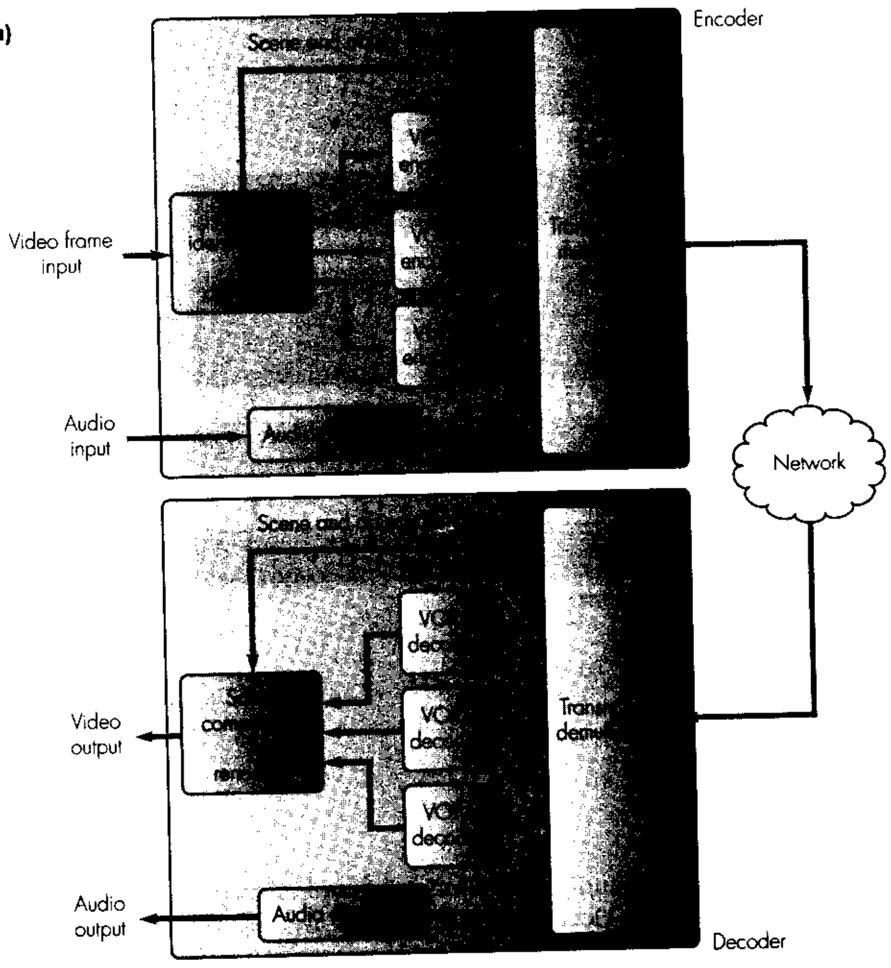
The audio associated with an AVO is compressed using one of the algorithms we described earlier in the chapter and depends on the available bit rate of the transmission channel and the sound quality required. Examples range from G.723.1 (CELP) for interactive multimedia applications over the Internet and video telephony over wireless networks and PSTNs, through to Dolby AC-3 or MPEG Layer 2 for interactive TV applications over entertainment networks.

An overview of the structure of the encoder associated with the audio and video of a frame/scene is shown in part (a) of Figure 4.24 and the essential features of each VOP encoder are shown in part (b). As we can see, first each VOP is identified and defined and is then encoded separately. In practice, the segmentation of a video frame into various objects – from which each VOP is derived – is a difficult image-processing task. In principle, it involves identifying regions of a frame that have similar properties such as color, texture, or brightness. Each of the resulting object shapes is then bounded by a rectangle (which contains the minimum number of macroblocks) to form the related VOP. Each is then encoded based on its shape, motion, and texture. Thus any VOP which has no motion associated with it will produce a minimum of compressed information. Also, since the VOP(s) which move often occupy only a small portion of the scene/frame, the bit rate of the multiplexed videostream is much lower than that obtained with the other standards. For applications that support interactions with a particular VOP, a number of advanced coding algorithms are available to perform the shape coding function.

Transmission format

As we show in Figure 4.25, all the information relating to a frame/scene encoded in MPEG-4 is transmitted over a network in the form of a **transport stream (TS)**, consisting of a multiplexed stream of **packetized elementary streams (PESs)**. As indicated earlier, it is intended that MPEG-4 should be used with applications relating to many different types of network including the Internet, a PSTN, a wireless network, or an entertainment network – cable/satellite/terrestrial. However, as we shall expand upon in Section 5.5.2 in the next chapter, a standard 188-byte packet format has been defined for use in the various types of digital TV broadcast networks. In most cases, the MPEG-4 transport stream uses this same packet format since this helps interworking with the encoded bitstreams associated with the MPEG-1/2 standards.

(a)



(b)

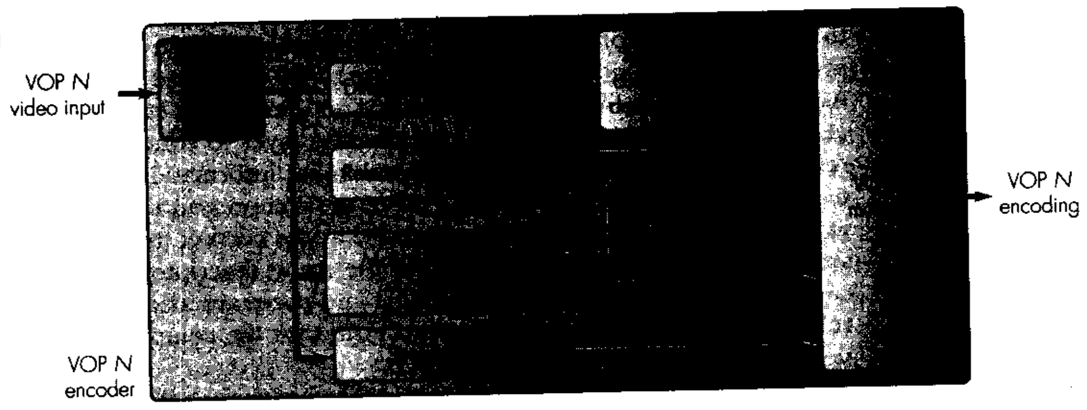


Figure 4.24 MPEG-4 coding principles: (a) encoder/decoder schematics; (b) VOP encoder schematic.

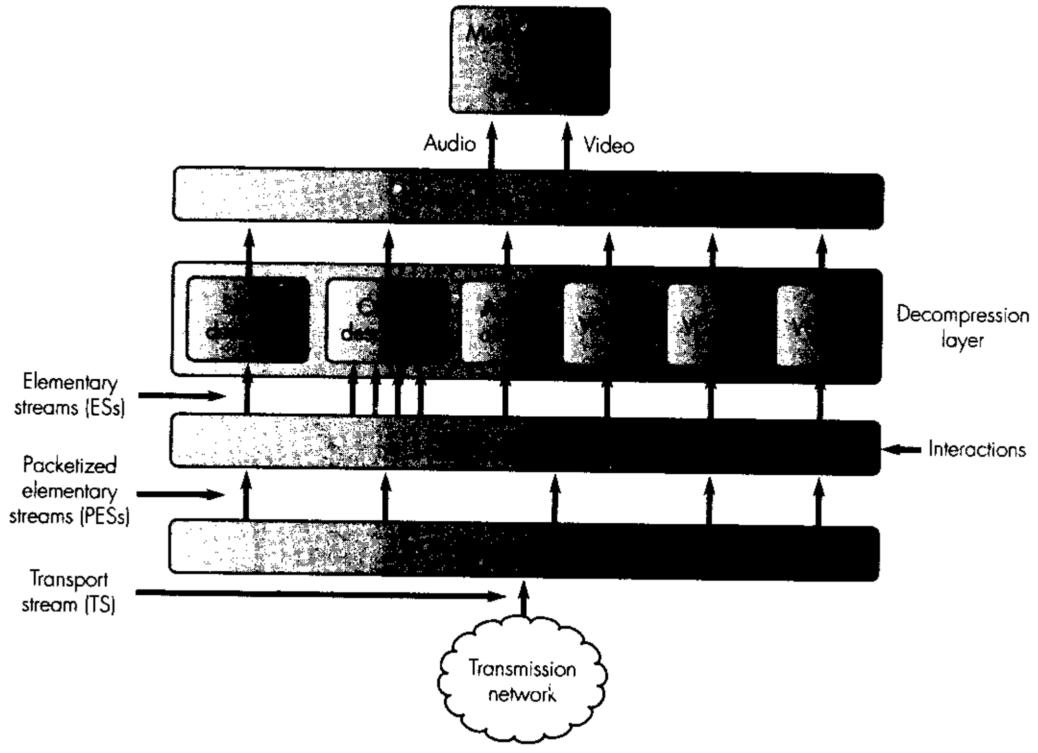


Figure 4.25 MPEG-4 decoder schematic.

The compressed audio and video information relating to each AVO in the scene is called an **elementary stream (ES)** and this is carried in the payload field of a PES packet. Each PES packet contains a *type* field in the packet header and this is used by the **FlexMux layer** to identify and route the PES to the related synchronization block in the **synchronization layer**. The compressed audio and video associated with each AVO is carried in a separate ES. Associated with each object descriptor is an **elementary stream descriptor (ESD)** which is used by the synchronization layer to route each ES to its related decoder. Each PES also contains timing information in the form of time-stamps and these are used to ensure the compressed information relating to each AVO is passed to the decoder at the correct time instants. The stream of object descriptor blocks and the scene descriptor block are both carried in separate elementary streams and these are also passed to their related decoder blocks in time synchronism with the compression information associated with the individual AVOs.

The output from the decoders associated with each AVO making up a frame, together with the related object scene descriptor information, is then passed to the **composition and rendering block**. As the name implies, the latter takes the decompressed data relating to each AVO and, together with the scene descriptor information, uses this to compose each frame ready for output to the display screen. The audio relating to each frame/scene is then played out in time synchronism with the video.

Error resilience techniques

As with the H.263 standard, a number of techniques are included in the MPEG-4 standard which make the encoding scheme more resilient to transmission errors. As we indicated earlier, these are important with transmission channels such as those associated with wireless networks and PSTNs. They include:

- the use of fixed-length video packets as the base-level data structure instead of GOBs;
- a new variable-length coding (VLC) scheme based on reversible VLCs.

We shall discuss the principles of each technique separately.

Video packets As we explained earlier in Sections 4.3.2 and 4.3.3 in the discussions relating to the error resilience techniques used in the H.261 and H.263 standards, resynchronization markers are inserted by the encoder at the start of each GOB to enable the decoder, on detecting an error in a GOB, to locate the end of the affected GOB and the start of the next GOB from where decoding recommences. A similar technique is used in the MPEG-1 and MPEG-2 standards at the slice level. In each case, however, when an error in a GOB/slice occurs, the complete set of macroblocks in the GOB/slice is discarded as there is no way of identifying the specific macroblocks that are corrupted.

In practice, the amount of compressed information in each GOB varies and depends on the level of activity/motion taking place in the related part of the frame. Thus, as we show in Figure 4.26(a), the number of bits in each GOB varies. In the figure, for example, it is assumed that the QCIF format is being used and that most of the activity is occurring in GOB 3 which, therefore, is much longer than the other two GOBs. Should a transmission error in a frame occur, it is highly probable that GOB 3 will be affected. This, of course, is the one we would prefer not to be affected. To overcome this effect, in MPEG-4 the compressed bitstream is divided into groups each comprising, not an equal number of macroblocks, but rather an equal number of bits. The resulting group of (compressed) bits is known as a **video packet** and, as with GOBs, these are separated by resynchronization markers.

The effect of using fixed-length video packets for the transmission of the compressed bitstream shown in Figure 4.26(a) is shown in part(b) of the figure and, as we can see, the effect is to have multiple resynchronization markers in those areas of the frame where most activity is taking place. With

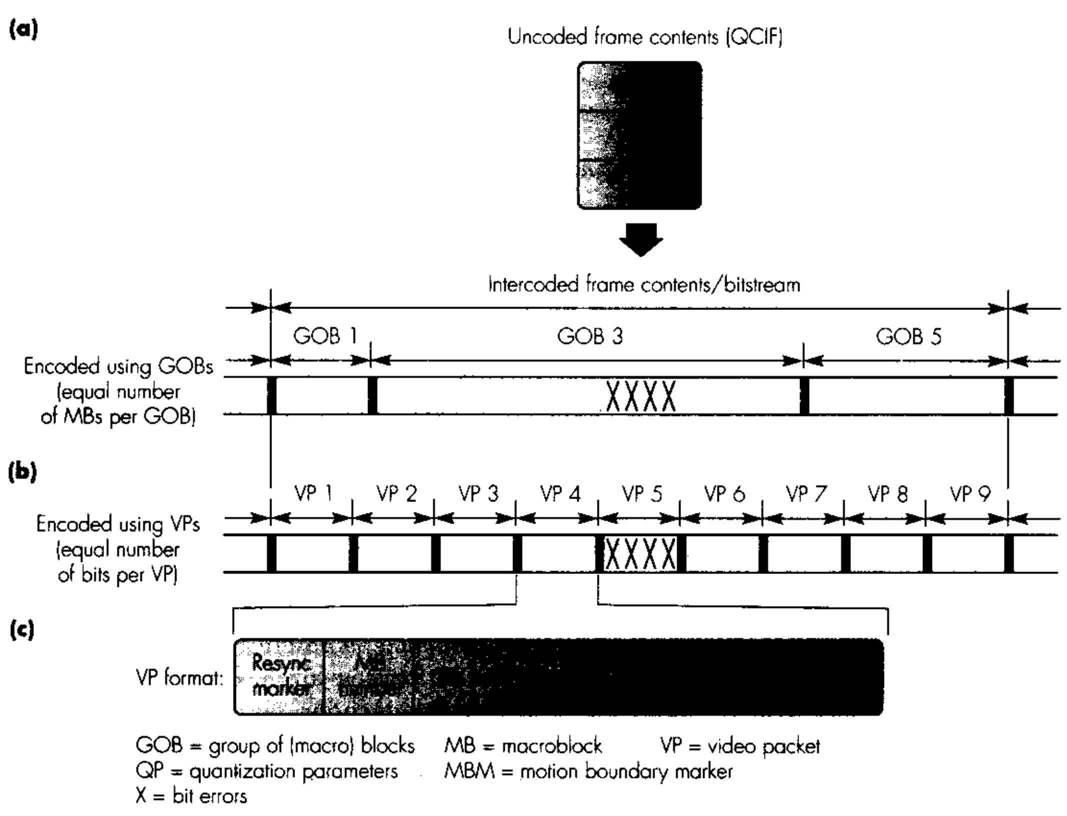

(a)
Uncoded frame contents (QCIF)
Intercoded frame contents/bitstream
GOB 1
GOB 3
GOB 5
Encoded using GOBs (equal number of MBs per GOB)
XXXX
(b)
VP 1
VP 2
VP 3
VP 4
VP 5
VP 6
VP 7
VP 8
VP 9
Encoded using VPs (equal number of bits per VP)
XXXX
(c)
VP format:
Resync marker
GOB = group of (macro) blocks
MB = macroblock
VP = video packet
QP = quantization parameters
MBM = motion boundary marker
X = bit errors

Figure 4.26 MPEG-4 encoding: (a) conventional GOB approach; (b) using fixed-length video packets; (c) video packet format.

this scheme, therefore, when an error occurs, only a reduced set of macroblocks are affected leaving the macroblocks in the remaining packets unaffected. In the standard, the suggested spacing of the resynchronization markers is determined by the link bit rate of the transmission channel; for example, for a bit rate of 24 kbps, the suggested spacing – and hence length of each video packet – is 512 bits.

The format of each video packet is shown in Figure 4.26(c) and, as we can see, it is similar to that of a GOB – which we showed earlier in Figure 4.15 (b) – except that after the resynchronization marker (GOB start code), the GOB number is replaced by the number of the first macroblock – relative to the complete frame – in the packet. In addition, in order to make the contents of each packet independent of the contents of other packets, the motion vectors are limited to the boundaries of the macroblocks that are in the packet.

A further refinement, as we show in the figure, is to separate the motion vectors for each macroblock in the packet from the DCT information. Normally, as we saw earlier in Figure 4.14(d), for each macroblock the encoded motion vector and DCT information are located together. As a result, the motion vectors and DCT information in a GOB are distributed throughout the complete GOB with the effect that, when an error in a GOB occurs, the complete GOB must be discarded. However, by separating these two fields by a unique bit pattern – known as the **motion boundary marker (MBM)** – it is then possible to identify separately each set of values and, if only one set is corrupted, the other set can be used.

An additional level of resilience is achieved by (optionally) including in the header of each packet a copy of the picture/frame-level parameters. These include the spatial dimensions of the picture/frame, the type of coding used (intra or inter), and temporal reference information (time-stamps). Clearly, if this is corrupted, then the decoder must discard the total frame contents. Hence by replicating this information in the header of each packet – or a selected number of packets – the decoder can use a voting system to determine if errors in the frame header were present and, if errors are detected, utilize the alternative information from the head of a packet instead. In order to indicate to the decoder that frame-level information is present in the header of a packet, the encoder sets a bit in the header field known as the **header extension code (HEC) bit.**

Reversible VLCs

As we showed earlier in Figure 4.14 and described in the associated text, after compression, the resulting motion vectors and DCT information are entropy encoded prior to transmission using a table of variable-length codewords (VLCs). When using conventional Huffman VLCs of the type we described in Section 3.4.5, any bit errors that occur will cause the decoder to lose synchronization and, typically, discard the remaining codewords until it finds the next motion boundary or resynchronization marker. In order to minimize this effect, MPEG-4 uses what are known as **reversible VLCs (RVLCs)**. Unlike conventional VLCs which have the prefix property, RVLCs can be decoded when read either from left-to-right or right-to-left. This means that the list of codewords making up the compressed bitstream can be identified whether the bitstream is decoded in either the forward or the reverse direction.

A simple way of producing a set of RVLCs is to first choose a set of VLCs each of which has a constant **Hamming weight**; that is, each codeword has the same number of binary 1s. The associated set of RVLCs is then produced by adding a fixed-length prefix and suffix to each of the corresponding VLCs. To illustrate this, a list of RVLCs and their derivation is shown in Figure 4.27(a). In this (simple) example, each VLC in the set has a Hamming weight of 1 and the fixed-length prefix and suffix is a single binary 1. Hence each RVLC in the resulting set of RVLCs has three binary 1s in it. As we can

(a)

VLC	RVLC
1	111
01	1011
001	10011
0001	100011

Maximum codeword length = 6 bits

(b)

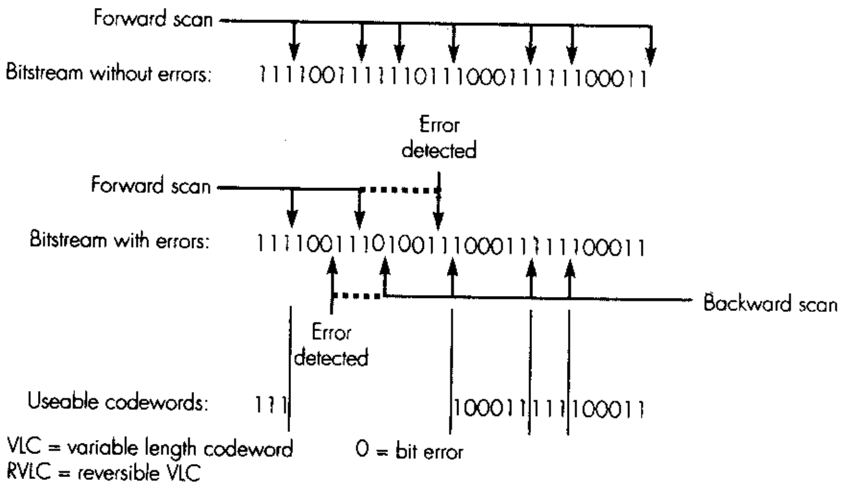

Forward scan
Bitstream without errors: 1111001111111011100011111100011
Error detected
Forward scan
Bitstream with errors: 111100111010011100011111100011
Backward scan
Error detected
Useable codewords: 111 100011 111 100011
VLC = variable length codeword
RVLC = reversible VLC
0 = bit error

Figure 4.27 Reversible VLCs: (a) example codeword set; (b) effect of transmission errors on decoding procedure.

deduce from the set of codewords, they do not have the prefix property but, when decoding a string of codewords, each can be readily identified by counting the number of binary 1s and, on receipt of each third 1, the end of a codeword is signaled. Moreover, this applies whether the bitstream is scanned in the forward or the reverse direction.

To exploit this property, the decoder first scans the received bitstream in the forward direction in the normal way identifying valid codewords as it proceeds. If an invalid (corrupted) codeword is detected, it terminates the search but, instead of discarding the remaining bits, it simply skips to the next motion boundary or resynchronization marker and proceeds to scan the bitstream in the reverse direction until it encounters an invalid codeword in this direction. The general principle is shown in Figure 4.27(b) and, as we can see, when an error is present, the two scans locate the error at different points in the bitstream resulting in an overlap region. The usable codewords are then determined by ensuring that, for each direction, the position pointer of the last correctly decoded codeword is less than the corresponding error detection pointer in the opposite direction.

Summary

In this chapter we have described the underlying principles associated with both audio and video compression. In addition, we have presented an overview of a number of international standards that have been defined for use in a range of multimedia applications.

The standards relating to audio compression are divided into algorithms for the compression of speech – as used in a range of interpersonal communication applications – and algorithms for the compression of general audio. The latter are used primarily in entertainment applications.

The standards relating to speech compression are summarized in Table 4.3. They are all ITU-T Recommendations except for LPC-10 which is a military standard. They all relate to interpersonal applications (including telephony, video telephony, and videoconferencing) and each has been defined for use with a particular networking infrastructure.

Table 4.3 Summary of Speech compression standards and their applications.

Standard	*Compression Technique*	*Compressed bit rate (kbps)*	*Speech Quality*	*Example applications*
[illegible]	[illegible] companding	64	Good	[illegible]
[illegible]	Adaptive differential PCM (ADPCM)	32 16	Good Fair	[illegible]
[illegible]	[illegible] coding	64 56/48	Excellent Good	[illegible]
[illegible]	[illegible]	40/32 24/16	Good Fair	[illegible]
LPC-10	linear predictive coding (LPC)	2.4/1.2	Poor	Telephony in military Networks
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	CELP	8	Good	[illegible]
[illegible]	CELP	8	Good	[illegible]
[illegible]	CELP	6.3 5.3	Good Fair	Video and Internet telephony

The general audio compression standards are summarized in Table 4.4. As we can see, the majority are intended for use in various entertainment applications.

A summary of the various video compression standards that were described is given in Table 4.5 and, as we can see, these have been defined for use in a range of multimedia applications. The two ITU-T standards – H.261 and H.263 – are intended for use with various types of network including PSTNs, ISDNs, LANs, intranets, and the Internet. Example applications include video telephony, videoconferencing, and security surveillance. The three MPEG standards – MPEG-1, 2, and 4 – are intended for use in various entertainment applications including the storage of video on CD-ROMs, digital video broadcasting, and more general interactive multimedia applications. The MPEG-7 standard is concerned with a means of defining the structure and content of the (compressed) information produced by the other standards.

Table 4.4 Summary of compression standards for general audio.

Standard	Compressed bit rate (kbps)	Quality	Example applications
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

Table 4.5 Summary of video compression standards.

Standard	Digitization format	Compressed bit rate	Example applications
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

Table 4.5 Continued

Standard	*Digitization format*	*Compressed bit rate*	*Example applications*
[illegible]	SIF	<1.5Mbps	Storage of VHS-quality video on CD-ROMs
[illegible]	SIF	<4Mbps	Recording of VHS-quality video
	4:2:0	<15Mbps	Digital video broadcasting
	4:2:2	<20Mbps	
[illegible]	4:2:0	<60Mbps	High definition television (4/3 aspect ratio)
	4:2:2	<80Mbps	
[illegible]	4:2:0	<80Mbps	High definition television (16/9 aspect ratio)
	4:2:2	<100Mbps	
MPEG-4	Various	5kbps-tens Mbps	Versatile multimedia coding standard
MPEG-7 [illegible]	[illegible]	–	Structure and content descriptions of compressed multimedia information [illegible]

Exercises

Section 4.2

4.1 With the aid of a schematic diagram, explain the operation of a basic DPCM signal encoder and decoder. Include in your explanation the source of errors that can arise.

4.2 Explain how the performance of a basic DPCM scheme can be improved by utilizing a more accurate version of the previous signal. Hence with the aid of a schematic diagram of the signal encoder and decoder, explain the principle of operation of a third-order DPCM scheme.

4.3 Explain how a basic ADPCM scheme obtains improved performance over a DPCM scheme. Give examples of the performance of a G.721-complaint codec.

4.4 With the aid of a schematic diagram of the signal encoder and decoder, explain how better sound quality – for the same bit rate – can be obtained using a subband coding ADPCM. Give examples of the bit rates used for the lower and higher subbands and state an application of this type of codec.

4.5 State the principle of operation of a G.726-compliant codec and give some example operating bit rates that are used.

4.6 Explain briefly how higher levels of compression can be obtained by making the predictor coefficients associated with ADPCM adaptive.

4.7 Explain the principles on which LPC codes are based. Hence, with the aid of a schematic diagram of an LPC encoder and decoder, identify the perception parameters and associated vocal tract excitation parameters that are used. Explain how these are used by the decoder to generate the output speech signal. State the meaning of LPC-10.

4.8 Explain the main difference between an LPC codec and a CELP codec. Include in your explanation the meaning of the terms "waveform template" and "template codebook".

4.9 Explain the meaning of the following terms relating to speech coders:
(i) processing delay,
(ii) algorithmic delay,
(iii) lookahead,
(iv) coder delay.

State the latter for a G.711 (PCM) and a G.723.1 (CELP) coder and hence explain how the coder delay often determines the suitability of a coder for a specific application.

4.10 Explain the principles on which perceptual coders are based and how they differ from an LPC and CELP coder.

4.11 With the aid of a graph, show how the sensitivity of the human ear varies with frequency. Explain clearly the dimensions of the y-axis of your graph and, by showing two equal-amplitude signals on your graph, explain how one may be heard and the other not.

4.12 With the aid of a graph showing the sensitivity of the human ear, explain the meaning of the term "frequency masking". Illustrate on your graph the masking effect of a loud signal on neighboring signals.

4.13 With the aid of a diagram showing the sensitivity of the human ear, illustrate how the effect of frequency masking varies with frequency. Hence explain the term "critical bandwidth" and identify how this also varies with frequency. Deduce how this effect can be exploited.

4.14 With the aid of a graph, explain the meaning of the term "temporal masking". What are the implications of exploiting this effect?

4.15 In relation to the schematic diagram shown in Figure 4.8 of an MPEG perceptual decoder, explain the operation of/meaning of the following:
(i) PCM encoder,
(ii) frequency subbands,
(iii) subband simple,
(iv) sampled segment duration,
(v) subband scaling factor,
(vi) psychoacoustic model,
(vii) quantizer,
(viii) frame formatter.

4.16 In relation to the schematic diagram shown in Figure 4.8 of an MPEG perceptual decoder, explain the operation/meaning of the following:
(i) frame demultiplexer,
(ii) dequantizer,
(iii) synthesis filter bank,
(iv) PCM decoder.

What are the implications of not requiring a psychoacoustic model in the decoder?

4.17 In relation to MPEG perceptual coders, explain the three levels of processing used and an application and typical bit rate of each.

4.18 With the aid of schematic diagrams, explain the difference between the forward adaptive bit allocation mode as used with an MPEG perceptual coder and the fixed bit allocation mode as used with a Dolby AC-1 coder.

4.19 With the aid of schematic diagrams, explain the operation of the following two types of perceptual coder:
(i) backward adaptive bit allocation (Dolby AC-2)
(ii) hybrid backward/forward adaptive bit allocation (Dolby AC-3).

State typical operational parameters with each coder type and an example application of each.

Section 4.3

4.20 Explain the meaning of the following terms relating to video compression:
(i) moving pictures,
(ii) MJPEG,
(iii) motion estimation and compensation.
Include in your explanation why the latter is used.

4.21 With the aid of example frame sequences, explain the meaning of the following types of compressed frame and the reasons for their use:
(i) I-frames,
(ii) P-frames,
(iii) B-frames.

4.22 Explain why I-frames are inserted into the compressed output stream relatively frequently. Hence explain the terms "group of pictures" (GOP) and "prediction span".

4.23 Assuming the uncoded frame sequence is:

...IBBPBBPBBBBI...

explain the implications of transmitting the compressed frames in this sequence. Hence derive a recorded sequence that overcomes any deficiencies.

4.24 With the aid of a frame sequence diagram, explain the meaning of the term "PB-frame" and why such frames are sometimes used.

4.25 Explain the application of D-frames and the compression technique that is used with them.

4.26 In relation to the encoding procedure of I-, P-, and B-frames, explain the meaning and use of the following terms:
(i) macroblock,
(ii) macroblock address,
(iii) target frame,
(iv) preceding reference frame,
(v) succeeding reference frame,
(vi) motion vector,
(vii) prediction error.

4.27 With the aid of a diagram, explain how the motion vector and prediction error are computed for a P-frame. Use for example purposes single-pixel resolution.

4.28 With the aid of diagrams, explain how the motion vectors(s) and prediction error are computed for a B-frame. Include in your explanation the meaning of the term "half-pixel resolution".

4.29 State and explain the encoding procedure used with
(i) the motion vector and
(ii) the prediction error.

4.30 In relation to the block schematic diagrams shown in Figure 4.14, explain
(i) the encoding of an I-frame,
(ii) the derivation of the contents of the reference frame used when encoding P- ,and B-frames,
(iii) the role of the frame formatter including the meaning of each of the fields shown in part (d) of the figure.

4.31 State the following relating to the CIF and QCIF formats of the H.261 encoding standard:
(i) the horizontal and vertical resolution in pixels
(ii) the number of macroblocks per frame
(iii) the number of GOBs per frame and their identity.

4.32 In relation to the macroblock and frame/picture formats shown in Figure 4.15, explain the role of the following fields:
(i) quantization value,
(ii) coded block pattern,
(iii) temporal reference,
(iv) start code/resyncronization marker.

4.33 With the aid of Figure 4.16, explain the role and operation of the quantization control block. Include in your explanation the use of the FIFO buffer and the associated high and low threshold levels.

4.34 Describe the following relating to the H.263 video compression standard assuming the QCIF:
(i) the number of pixels per frame,
(ii) the number of GOBs per frame,
(iii) the encoding of a PB frame,
(iv) unrestricted motion vectors.

4.35 With reference to the frame sequence diagram shown in Figure 4.17(a), explain how
(i) the corruption of one or more macroblocks within a GOB may corrupt the whole GOB

(ii) errors within a GOB may propagate to other neighbouring GOBs.

4.36 With the aid of the frame sequence diagram shown in Figure 4.17(b), explain the operation of the error tracking scheme. Include in your explanation:
(i) how errors are detected,
(ii) error prediction information and its use,
(iii) NAK messages and their content.

4.37 With the aid of the frame sequence diagram shown in Figure 4.18(b), explain the operation of the independent segment decoding scheme. Include in your explanation:
(i) how errors in a GOB do not propagate to neighbouring GOBs,
(ii) how error tracking can be used to improve error concealment.

4.38 With the aid of the sequence diagrams in Figures 4.19(a) and (b), explain the operation of the reference picture selection scheme. Include:
(i) how the propagation of errors in a GOB is overcome in the NAK mode
(ii) how the choice of reference frame is made in the ACK mode.

4.39 For the MPEG-1 frame sequence shown in Figure 4.20, derive a suitable recorded sequence of frames that results in minimum delays in the decoder.

4.40 State how the compression algorithm used with MPEG-1 differs from that used in the H.261 standard.

4.41 Assuming the frame sequence shown in Figure 4.21(a) and average ratios of 12:1 (I), 20:1 (P), and 40:1 (B), derive the average bit rate generated by an MPEG-1 encoder for both the NTSC and PAL digitization formats.

4.42 Explain the meaning of the following terms relating to the video bitstream structure shown in Figure 4.21:
(i) sequence,
(ii) GOP,
(iii) slice,
(iv) time-stamp,
(v) buffer and encode parameters.

4.43 With the aid of a diagram, explain the difference between interlaced and progressive scanning.

4.44 With the aid of the examples shown in Figure 4.22, explain how the DCT blocks are derived from each macroblock in an I-frame
(i) in the field mode and
(ii) in the frame mode.
State an application for each mode.

4.45 With the aid of a diagram, explain the meaning of the following terms relating to MPEG-4 video compression standard:
(i) content-based functionality,
(ii) audio and video objects,
(iii) video object plain.

4.46 With the aid of the diagrams shown in Figure 4.24, explain the meaning of the terms:
(i) scene and object descriptions,
(ii) scene composition and rendering,
(iii) texture, motion, and shape encoding.

4.47 In relation to the schematic diagram of an MPEG-4 decoder shown in Figure 4.25, explain the role of the following:
(i) transport stream,
(ii) flexmux layer,
(iii) PES,
(iv) synchronization layer,
(v) ES,
(vi) decomposition layer,
(vii) composition and rendering layer.

4.48 With the aid of the diagrams shown in Figure 4.26, explain:
(i) the advantages of using fixed-length video packets instead of GOBs with low bit rate wireless and PSTNs
(ii) the separation of the motion vectors and DCT information in each video packet by motion boundary marker
(iii) the replication of picture/frame-level parameters in the header of each video packet.

4.49 With the aid of the diagrams shown in Figure 4.27, explain:
(i) how reversible variable-length codewords (RVLCs) reduce the effects of transmission errors
(ii) the derivation of RVLCs
(iii) forward and reverse scans and their use.