WILEY

VIDEO CODEC DESIGN

Developing Image and Video Compression Systems

Iain E. G. Richardson

Video Codec Design

To
Freya and Hugh

Video Codec Design

Developing Image and Video Compression Systems

Iain E. G. Richardson

The Robert Gordon University, Aberdeen, UK

JOHN WILEY & SONS, LTD

Baffins Lane, Chichester,
West Sussex PO19 IUD, England

National 01243 779777
International (+44) 1243 779777

e-mail (for orders and customer service enquiries): cs-books@wiley.co.uk
Visit our Home Page on http://www.wileyeurope.com

Other Wiley Editorial Offices

John Wiley & Sons, Inc., 605 Third Avenue,
New York, NY 10158-0012, USA

WILEY-VCH Verlag GmbH, Pappelallee 3,
D-69469 Weinheim, Germany

John Wiley & Sons Australia Ltd, 33 Park Road, Milton,
Queensland 4064, Australia

John Wiley & Sons (Asia) Pte Ltd, 2 Clementi Loop #02-01,
Jin Xing Distripark, Singapore 129809

John Wiley & Sons (Canada) Ltd, 22 Worcester Road,
Rexdale, Ontario M9W 1L1, Canada

British Library Cataloguing in Publication Data

A catalogue record for this book is available from the British Library

ISBN 0 471 48553 5

Typeset in 10/12 Times by Thomson Press (India) Ltd., New Delhi
Printed and bound in Great Britain by Antony Rowe Ltd, Chippenham, Wiltshire
This book is printed on acid-free paper responsibly manufactured from sustainable forestry,
in which at least two trees are planted for each one used for paper production.

Contents

4

Video Coding Standards: JPEG and MPEG

4.1 INTRODUCTION

The majority of video CODECs in use today conform to one of the international standards for video coding. Two standards bodies, the International Standards Organisation (ISO) and the International Telecommunications Union (ITU), have developed a series of standards that have shaped the development of the visual communications industry. The ISO JPEG and MPEG-2 standards have perhaps had the biggest impact: JPEG has become one of the most widely used formats for still image storage and MPEG-2 forms the heart of digital television and DVD-video systems. The ITU's H.261 standard was originally developed for video conferencing over the ISDN, but H.261 and H.263 (its successor) are now widely used for real-time video communications over a range of networks including the Internet.

This chapter begins by describing the process by which these standards are proposed, developed and published. We describe the popular ISO coding standards, JPEG and JPEG-2000 for still images, MPEG-1, MPEG-2 and MPEG-4 for moving video. In Chapter 5 we introduce the ITU-T H.261, H.263 and H.26L standards.

4.2 THE INTERNATIONAL STANDARDS BODIES

It was recognised in the 1980s that video coding and transmission could become a commercially important application area. The development of video coding technology since then has been bound up with a series of international standards for image and video coding. Each of these standards supports a particular application of video coding (or a set of applications), such as video conferencing and digital television. The aim of an image or video coding standard is to support a particular class of application and to encourage interoperability between equipment and systems from different manufacturers. Each standard describes a *syntax* or method of representation for compressed images or video. The developers of each standard have attempted to incorporate the best developments in video coding technology (in terms of coding efficiency and ease of practical implementation).

Each of the international standards takes a similar approach to meeting these goals. A video coding standard describes syntax for representing compressed video data and the procedure for decoding this data as well as (possibly) a 'reference' decoder and methods of proving conformance with the standard.

In order to provide the maximum flexibility and scope for innovation, the standards do not define a video or image encoder: this is left to the designer's discretion. However, in practice the syntax elements and reference decoder limit the scope for alternative designs that still meet the requirements of the standard.

4.2.1 The Expert Groups

The most important developments in video coding standards have been due to two international standards bodies: the ITU (formerly the CCITT)[1] and the ISO.[2] The ITU has concentrated on standards to support real-time, two-way video communications. The group responsible for developing these standards is known as VCEG (Video Coding Experts Group) and has issued:

- H.261 (1990): Video telephony over constant bit-rate channels, primarily aimed at ISDN channels of $p \times 64$ kbps.
- H.263 (1995): Video telephony over circuit- and packet-switched networks, supporting a range of channels from low bit rates (20–30 kbps) to high bit rates (several Mbps).
- H.263+ (1998), H.263++ (2001): Extensions to H.263 to support a wider range of transmission scenarios and improved compression performance.
- H.26L (under development): Video communications over channels ranging from very low (under 20 kbps) to high bit rates.

The H.26x series of standards will be described in Chapter 5. In parallel with the ITU's activities, the ISO has issued standards to support storage and distribution applications. The two relevant groups are JPEG (Joint Photographic Experts Group) and MPEG (Moving Picture Experts Group) and they have been responsible for:

- JPEG (1992)[3]: Compression of still images for storage purposes.
- MPEG-1 (1993)[4]: Compression of video and audio for storage and real-time play back on CD-ROM (at a bit rate of 1.4 Mbps).
- MPEG-2 (1995)[5]: Compression and transmission of video and audio programmes for storage and broadcast applications (at typical bit rates of 3–5 Mbps and above).
- MPEG-4 (1998)[6]: Video and audio compression and transport for multimedia terminals (supporting a wide range of bit rates from around 20–30 kbps to high bit rates).
- JPEG-2000 (2000)[7]: Compression of still images (featuring better compression performance than the original JPEG standard).

Since releasing Version 1 of MPEG-4, the MPEG committee has concentrated on 'framework' standards that are not primarily concerned with video coding:

- MPEG-7[8]: Multimedia Content Description Interface. This is a standard for describing multimedia content data, with the aim of providing a standardised system for content-based

indexing and retrieval of multimedia information. MPEG-7 is concerned with access to multimedia data rather than the mechanisms for coding and compression. MPEG-7 is scheduled to become an international standard in late 2001.

- MPEG-21[9]: Multimedia Framework. The MPEG-21 initiative looks beyond coding and indexing to the complete multimedia content 'delivery chain', from creation through production and delivery to 'consumption' (e.g. viewing the content). MPEG-21 will define key elements of this delivery framework, including content description and identification, content handling, intellectual property management, terminal and network interoperation and content representation. The motivation behind MPEG-21 is to encourage integration and interoperation between the diverse technologies that are required to create, deliver and decode multimedia data. Work on the proposed standard started in June 2000.

Figure 4.1 shows the relationship between the standards bodies, the expert groups and the video coding standards. The expert groups have addressed different application areas (still images, video conferencing, entertainment and multimedia), but in practice there are many overlaps between the applications of the standards. For example, a version of JPEG, Motion JPEG, is widely used for video conferencing and video surveillance; MPEG-1 and MPEG-2 have been used for video conferencing applications; and the core algorithms of MPEG-4 and H.263 are identical.

In recognition of these natural overlaps, the expert groups have cooperated at several stages and the result of this cooperation has led to outcomes such as the ratification of MPEG-2 (Video) as ITU standard H.262 and the incorporation of 'baseline' H.263 into MPEG-4 (Video). There is also interworking between the VCEG and MPEG committees and

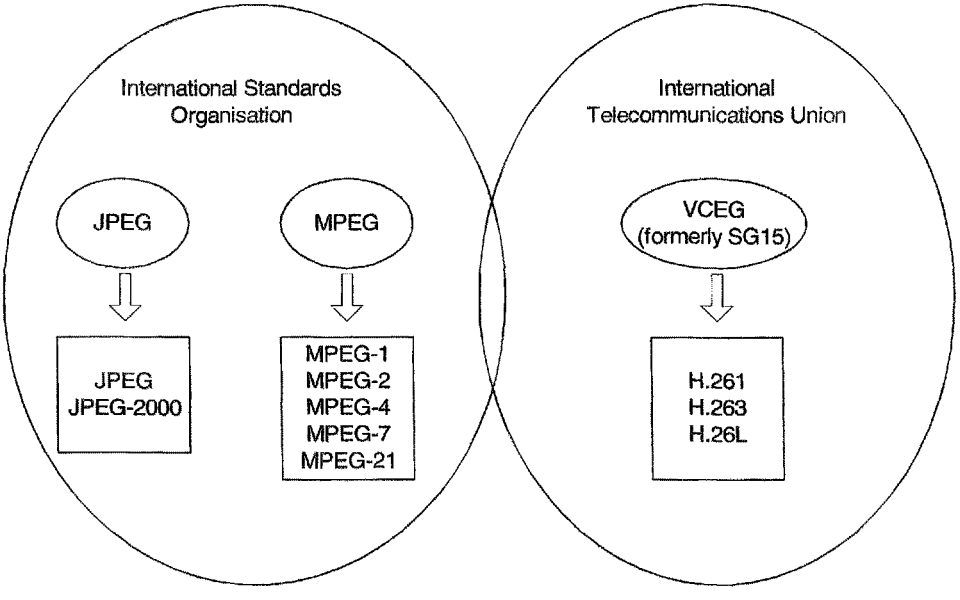


Figure 4.1 International standards bodies

other related bodies such as the Internet Engineering Task Force (IETF), industry groups (such as the Digital Audio Visual Interoperability Council, DAVIC) and other groups within ITU and ISO.

4.2.2 The Standardisation Process

The development of an international standard for image or video coding is typically an involved process:

1. The scope and aims of the standard are defined. For example, the emerging H.26L standard is designed with real-time video communications applications in mind and aims to improve performance over the preceding H.263 standard.
2. Potential technologies for meeting these aims are evaluated, typically by competitive testing. The test scenario and criteria are defined and interested parties are encouraged to participate and demonstrate the performance of their proposed solutions. The 'best' technology is chosen based on criteria such as coding performance and implementation complexity.
3. The chosen technology is implemented as a *test model*. This is usually a software implementation that is made available to members of the expert group for experimentation, together with a *test model document* that describes its operation.
4. The test model is developed further: improvements and features are proposed and demonstrated by members of the expert group and the best of these developments are integrated into the test model.
5. At a certain point (depending on the timescales of the standardisation effort and on whether the aims of the standard have been sufficiently met by the test model), the model is 'frozen' and the test model document forms the basis of a *draft standard*.
6. The draft standard is reviewed and after approval becomes a published *international standard*.

Officially, the standard is not available in the public domain until the final stage of approval and publication. However, because of the fast-moving nature of the video communications industry, draft documents and test models can be very useful for developers and manufacturers. Many of the ITU VCEG documents and models are available via public FTP.[10] Most of the MPEG working documents are restricted to members of MPEG itself, but a number of overview documents are available at the MPEG website.[11] Information and links about JPEG and MPEG are available.[12,13] Keeping in touch with the latest developments and gaining access to draft standards are powerful reasons for companies and organisations to become involved with the MPEG, JPEG and VCEG committees.

4.2.3 Understanding and Using the Standards

Published ITU and ISO standards may be purchased from the relevant standards body.[1,2] For developers of standards-compliant video coding systems, the published standard is an

essential point of reference as it defines the syntax and capabilities that a video CODEC must conform to in order to successfully interwork with other systems. However, the standards themselves are not an ideal introduction to the concepts and techniques of video coding: the aim of the standard is to define the syntax as explicitly and unambiguously as possible and this does not make for easy reading.

Furthermore, the standards do not necessarily indicate practical constraints that a designer must take into account. Practical issues and good design techniques are deliberately left to the discretion of manufacturers in order to encourage innovation and competition, and so other sources are a much better guide to practical design issues. This book aims to collect together information and guidelines for designers and integrators; other texts that may be useful for developers are listed in the bibliography.

The test models produced by the expert groups are designed to facilitate experimentation and comparison of alternative techniques, and the test model (a software model with an accompanying document) can provide a valuable insight into the implementation of the standard. Further documents such as implementation guides (e.g. H.263 Appendix III[14]) are produced by the expert groups to assist with the interpretation of the standards for practical applications.

In recent years the standards bodies have recognised the need to direct developers towards certain subsets of the tools and options available within the standard. For example, H.263 now has a total of 19 optional modes and it is unlikely that any particular application would need to implement all of these modes. This has led to the concept of *profiles* and *levels*. A 'profile' describes a subset of functionalities that may be suitable for a particular application and a 'level' describes a subset of operating resolutions (such as frame resolution and frame rates) for certain applications.

4.3 JPEG (JOINT PHOTOGRAPHIC EXPERTS GROUP)

4.3.1 JPEG

International standard ISO 10918[3] is popularly known by the acronym of the group that developed it, the Joint Photographic Experts Group. Released in 1992, it provides a method and syntax for compressing continuous-tone still images (such as photographs). Its main application is storage and transmission of still images in a compressed form, and it is widely used in digital imaging, digital cameras, embedding images in web pages, and many more applications. Whilst aimed at still image compression, JPEG has found some popularity as a simple and effective method of compressing moving images (in the form of Motion JPEG).

The JPEG standard defines a syntax and decoding process for a *baseline CODEC* and this includes a set of features that are designed to suit a wide range of applications. Further optional modes are defined that extend the capabilities of the baseline CODEC.

The baseline CODEC

A baseline JPEG CODEC is shown in block diagram form in Figure 4.2. Image data is processed one 8 × 8 block at a time. Colour components or planes (e.g. R, G, B or Y, Cr, Cb)

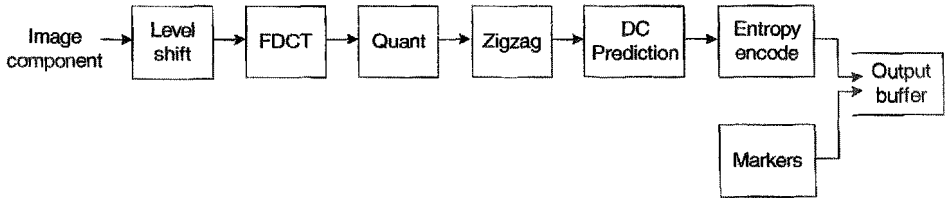


Figure 4.2 JPEG baseline CODEC block diagram

may be processed separately (one complete component at a time) or in interleaved order (e.g. a block from each of three colour components in succession). Each block is coded using the following steps.

Level shift Input data is shifted so that it is distributed about zero: e.g. an 8-bit input sample in the range 0 : 255 is shifted to the range −128 : 127 by subtracting 128.

Forward DCT An 8 × 8 block transform, described in Chapter 7.

Quantiser Each of the 64 DCT coefficients C_{ij} is quantised by integer division:

$$Cq_{ij} = \text{round}\left(\frac{C_{ij}}{Q_{ij}}\right)$$

Q_{ij} is a quantisation parameter and Cq_{ij} is the quantised coefficient. A larger value of Q_{ij} gives higher compression (because more coefficients are set to zero after quantisation) at the expense of increased distortion in the decoded image. The 64 parameters Q_{ij} (one for each coefficient position ij) are stored in a quantisation 'map'. The map is not specified by the standard but can be perceptually weighted so that lower-frequency coefficients (DC and low-frequency AC coefficients) are quantised less than higher-frequency coefficients. Figure 4.3

Low frequencies

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

High frequencies

Figure 4.3 JPEG quantisation map

gives an example of a quantisation map: the weighting means that the visually important lower frequencies (to the top left of the map) are preserved and the less important higher frequencies (to the bottom right) are more highly compressed.

Zigzag reordering The 8×8 block of quantised coefficients is rearranged in a zigzag order so that the low frequencies are grouped together at the start of the rearranged array.

DC differential prediction Because there is often a high correlation between the DC coefficients of neighbouring image blocks, a prediction of the DC coefficient is formed from the DC coefficient of the preceding block:

$$DC_{pred} = DC_{cur} - DC_{prev}$$

The prediction DC_{pred} is coded and transmitted, rather than the actual coefficient DC_{cur}.

Entropy encoding The differential DC coefficients and AC coefficients are encoded as follows. The number of bits required to represent the DC coefficient, SSSS, is encoded using a variable-length code. For example, SSSS=0 indicates that the DC coefficient is zero; SSSS=1 indicates that the DC coefficient is +/−1 (i.e. it can be represented with 1 bit); SSSS=2 indicates that the coefficient is +3, +2, −2 or −3 (which can be represented with 2 bits). The actual value of the coefficient, an SSSS-bit number, is appended to the variable-length code (except when SSSS=0).

Each AC coefficient is coded as a variable-length code RRRRSSSS, where RRRR indicates the number of preceding zero coefficients and SSSS indicates the number of bits required to represent the coefficient (SSSS=0 is not required). The actual value is appended to the variable-length code as described above.

Example

A run of six zeros followed by the value +5 would be coded as:

[RRRR=6] [SSSS=3] [Value=+5]

Marker insertion Marker codes are inserted into the entropy-coded data sequence. Examples of markers include the frame header (describing the parameters of the frame such as width, height and number of colour components), scan headers (see below) and restart interval markers (enabling a decoder to resynchronise with the coded sequence if an error occurs).

The result of the encoding process is a compressed sequence of bits, representing the image data, that may be transmitted or stored. In order to view the image, it must be decoded by reversing the above steps, starting with marker detection and entropy decoding and ending with an inverse DCT. Because quantisation is not a reversible process (as discussed in Chapter 3), the decoded image is not identical to the original image.

Lossless JPEG

JPEG also defines a lossless encoding/decoding algorithm that uses DPCM (described in Chapter 3). Each pixel is predicted from up to three neighbouring pixels and the predicted value is entropy coded and transmitted. Lossless JPEG guarantees image fidelity at the expense of relatively poor compression performance.

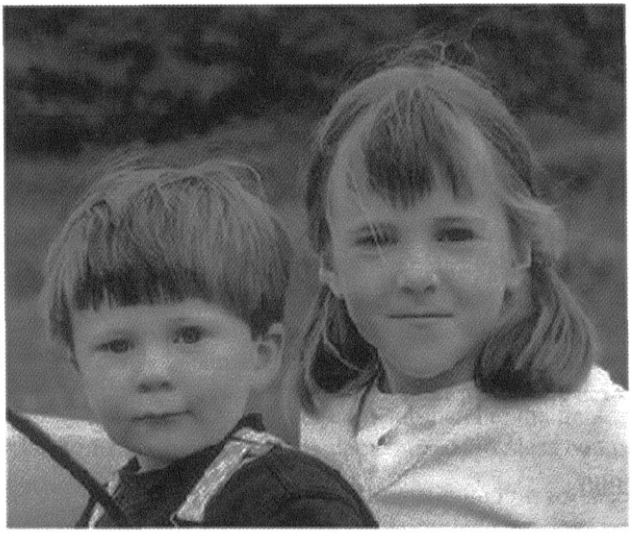
Optional modes

Progressive encoding involves encoding the image in a series of progressive 'scans'. The first scan may be decoded to provide a 'coarse' representation of the image; decoding each subsequent scan progressively improves the quality of the image until the final quality is reached. This can be useful when, for example, a compressed image takes a long time to transmit: the decoder can quickly recreate an approximate image which is then further refined in a series of passes. Two versions of progressive encoding are supported: *spectral selection*, where each scan consists of a subset of the DCT coefficients of every block (e.g. (a) DC only; (b) low-frequency AC; (c) high-frequency AC coefficients) and *successive approximation*, where the first scan contains N most significant bits of each coefficient and later scans contain the less significant bits. Figure 4.4 shows an image encoded and decoded using progressive spectral selection. The first image contains the DC coefficients of each block, the second image contains the DC and two lowest AC coefficients and the third contains all 64 coefficients in each block.

(a)

Figure 4.4 Progressive encoding example (spectral selection): (a) DC only; (b) DC + two AC; (c) all coefficients

(b)



(c)

Figure 4.4 (*Contined*)

Hierarchical encoding compresses an image as a series of components at different spatial resolutions. For example, the first component may be a subsampled image at a low spatial resolution, followed by further components at successively higher resolutions. Each successive component is encoded *differentially* from previous components, i.e. only the differences are encoded. A decoder may choose to decode only a subset of the full resolution image; alternatively, the successive components may be used to progressively refine the resolution in a similar way to progressive encoding.

The two progressive encoding modes and the hierarchical encoding mode can be thought of as *scalable coding* modes. Scalable coding will be discussed further in the section on MPEG-2.

4.3.2 Motion JPEG

A 'Motion JPEG' or MJPEG CODEC codes a video sequence as a series of JPEG images, each corresponding to one frame of video (i.e. a series of intra-coded frames). Originally, the JPEG standard was not intended to be used in this way: however, MJPEG has become popular and is used in a number of video communications and storage applications. No attempt is made to exploit the inherent temporal redundancy in a moving video sequence and so compression performance is poor compared with inter-frame CODECs (see Chapter 5, 'Performance Comparison'). However, MJPEG has a number of practical advantages:

- *Low complexity*: algorithmic complexity, and requirements for hardware, processing and storage are very low compared with even a basic inter-frame CODEC (e.g. H.261).
- *Error tolerance*: intra-frame coding limits the effect of an error to a single decoded frame and so is inherently resilient to transmission errors. Until recent developments in error resilience (see Chapter 11), MJPEG outperformed inter-frame CODECs in noisy environments.
- *Market awareness*: JPEG is perhaps the most widely known and used of the compression standards and so potential users are already familiar with the technology of Motion JPEG.

Because of its poor compression performance, MJPEG is only suitable for high-bandwidth communications (e.g. over dedicated networks). Perversely, this means that users generally have a good experience of MJPEG because installations do not tend to suffer from the bandwidth and delay problems encountered by inter-frame CODECs used over 'best effort' networks (such as the Internet) or low bit-rate channels. An MJPEG coding integrated circuit(IC), the Zoran ZR36060, is described in Chapter 12.

4.3.3 JPEG-2000

The original JPEG standard has gained widespread acceptance and is now ubiquitous throughout computing applications: it is the main format for photographic images on the world wide web and it is widely used for image storage. However, the block-based DCT algorithm has a number of disadvantages, perhaps the most important of which is the 'blockiness' of highly compressed JPEG images (see Chapter 9). Since its release, many alternative coding schemes have been shown to outperform baseline JPEG. The need for better performance at high compression ratios led to the development of the JPEG-2000 standard.[7,15]

The features that JPEG-2000 aims to support are as follows:

- Good compression performance, particularly at high compression ratios.

- Efficient compression of continuous-tone, bi-level and compound images (e.g. photographic images with overlaid text: the original JPEG does not handle this type of image well).
- Lossless and lossy compression (within the same compression framework).
- Progressive transmission (JPEG-2000 supports SNR scalability, a similar concept to JPEG's successive approximation mode, and spatial scalability, similar to JPEG's hierarchical mode).
- Region-of-interest (ROI) coding. This feature allows an encoder to specify an arbitrary region within the image that should be treated differently during encoding: e.g. by encoding the region with a higher quality or by allowing independent decoding of the ROI.
- Error resilience tools including data partitioning (see the description of MPEG-2 below), error detection and concealment (see Chapter 11 for more details).
- Open architecture. The JPEG-2000 standard provides an open 'framework' which should make it relatively easy to add further coding features either as part of the standard or as a proprietary 'add-on' to the standard.

The architecture of a JPEG-2000 encoder is shown in Figure 4.5. This is superficially similar to the JPEG architecture but one important difference is that the same architecture may be used for lossy or lossless coding.

The basic coding unit of JPEG-2000 is a 'tile'. This is normally a $2^n \times 2^n$ region of the image, and the image is 'covered' by non-overlapping identically sized tiles. Each tile is encoded as follows:

- *Transform*: A wavelet transform is carried out on each tile to decompose it into a series of sub-bands (see Sections 3.3.1 and 7.3). The transform may be reversible (for lossless coding applications) or irreversible (suitable for lossy coding applications).
- *Quantisation*: The coefficients of the wavelet transform are quantised (as described in Chapter 3) according to the 'importance' of each sub-band to the final image appearance. There is an option to leave the coefficients unquantised (lossless coding).
- *Entropy coding*: JPEG-2000 uses a form of arithmetic coding to encode the quantised coefficients prior to storage or transmission. Arithmetic coding can provide better compression efficiency than variable-length coding and is described in Chapter 8.

The result is a compression standard that can give significantly better image compression performance than JPEG. For the same image quality, JPEG-2000 can usually compress images by at least twice as much as JPEG. At high compression ratios, the quality of images

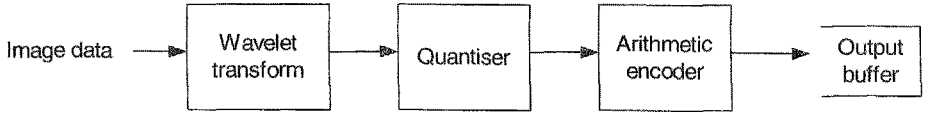


Figure 4.5 Architecture of JPEG-2000 encoder

degrades gracefully, with the decoded image showing a gradual 'blurring' effect rather than the more obvious blocking effect associated with the DCT. These performance gains are achieved at the expense of increased complexity and storage requirements during encoding and decoding. One effect of this is that images take longer to store and display using JPEG-2000 (though this should be less of an issue as processors continue to get faster).

4.4 MPEG (MOVING PICTURE EXPERTS GROUP)

4.4.1 MPEG-1

The first standard produced by the Moving Picture Experts Group, popularly known as MPEG-1, was designed to provide video and audio compression for storage and playback on CD-ROMs. A CD-ROM played at 'single speed' has a transfer rate of 1.4 Mbps. MPEG-1 aims to compress video and audio to a bit rate of 1.4 Mbps with a quality that is comparable to VHS videotape. The target market was the 'video CD', a standard CD containing up to 70 minutes of stored video and audio. The video CD was never a commercial success: the quality improvement over VHS tape was not sufficient to tempt consumers to replace their video cassette recorders and the maximum length of 70 minutes created an irritating break in a feature-length movie. However, MPEG-1 is important for two reasons: it has gained widespread use in other video storage and transmission applications (including CD-ROM storage as part of interactive applications and video playback over the Internet), and its functionality is used and extended in the popular MPEG-2 standard.

The MPEG-1 standard consists of three parts. Part 1[16] deals with system issues (including the multiplexing of coded video and audio), Part 2[4] deals with compressed video and Part 3[17] with compressed audio. Part 2 (video) was developed with aim of supporting efficient coding of video for CD playback applications and achieving video quality comparable to, or better than, VHS videotape at CD bit rates (around 1.2 Mbps for video). There was a requirement to minimise decoding complexity since most consumer applications were envisaged to involve decoding and playback only, not encoding. Hence MPEG-1 decoding is considerably simpler than encoding (unlike JPEG, where the encoder and decoder have similar levels of complexity).

MPEG-1 features

The input video signal to an MPEG-1 video encoder is 4 : 2 : 0 Y : Cr : Cb format (see Chapter 2) with a typical spatial resolution of 352 × 288 or 352 × 240 pixels. Each frame of video is processed in units of a *macroblock*, corresponding to a 16 × 16 pixel area in the displayed frame. This area is made up of 16 × 16 luminance samples, 8 × 8 Cr samples and 8 × 8 Cb samples (because Cr and Cb have half the horizontal and vertical resolution of the luminance component). A macroblock consists of six 8 × 8 blocks: four luminance (Y) blocks, one Cr block and one Cb block (Figure 4.6).

Each frame of video is encoded to produce a coded *picture*. There are three main types: I-pictures, P-pictures and B-pictures. (The standard specifies a fourth picture type, D-pictures, but these are seldom used in practical applications.)

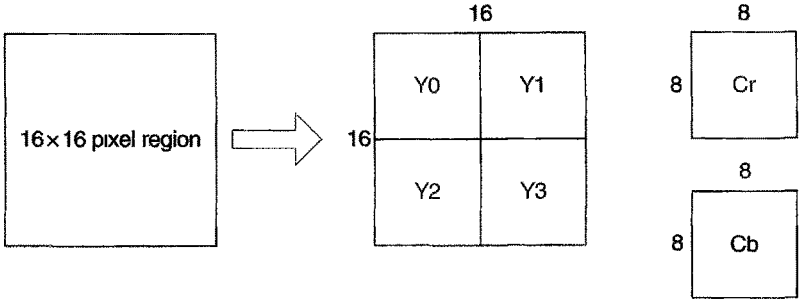


Figure 4.6 Structure of a macroblock

I-pictures are intra-coded without any motion-compensated prediction (in a similar way to a baseline JPEG image). An I-picture is used as a reference for further predicted pictures (P- and B-pictures, described below).

P-pictures are inter-coded using motion-compensated prediction from a *reference picture* (the P-picture or I-picture preceding the current P-picture). Hence a P-picture is predicted using *forward prediction* and a P-picture may itself be used as a reference for further predicted pictures (P- and B-pictures).

B-pictures are inter-coded using motion-compensated prediction from *two* reference pictures, the P- and/or I-pictures before and after the current B-picture. Two motion vectors are generated for each macroblock in a B-picture (Figure 4.7): one pointing to a matching area in the previous reference picture (a *forward* vector) and one pointing to a matching area

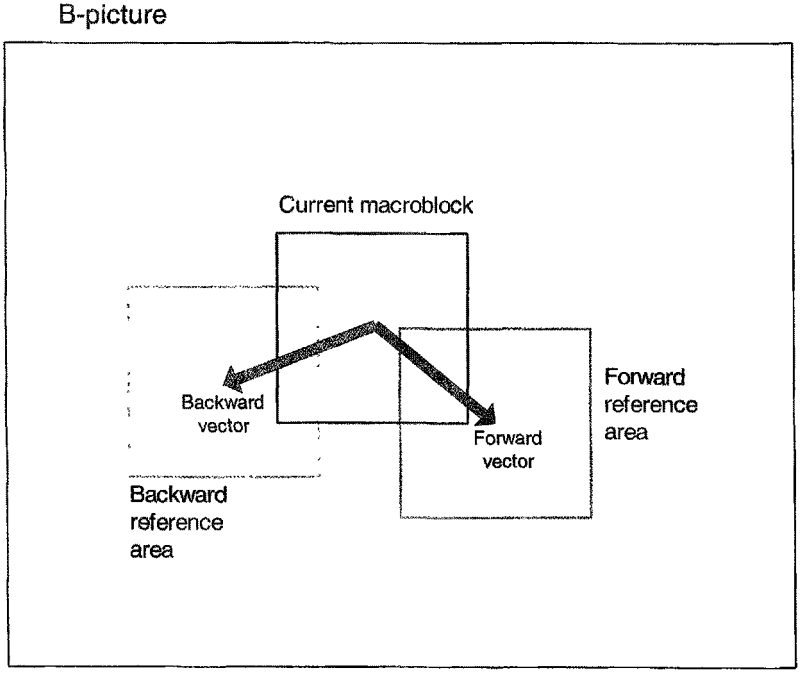


Figure 4.7 Prediction of B-picture macroblock using forward and backward vectors

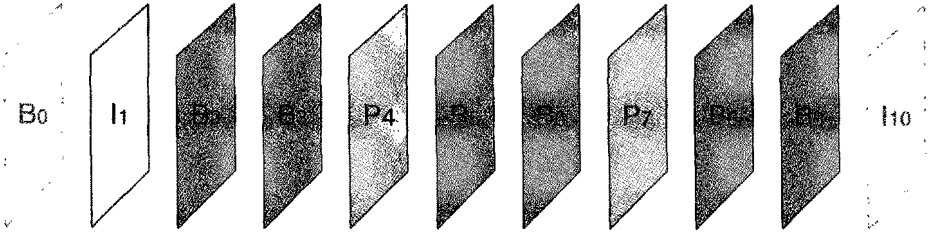


Figure 4.8 MPEG-1 group of pictures (IBBPBBPBB): display order

in the future reference picture (a *backward* vector). A motion-compensated prediction macroblock can be formed in three ways: forward prediction using the forward vector, backwards prediction using the backward vector or bidirectional prediction (where the prediction reference is formed by averaging the forward and backward prediction references). Typically, an encoder chooses the prediction mode (forward, backward or bidirectional) that gives the lowest energy in the difference macroblock. B-pictures are not themselves used as prediction references for any further predicted frames.

Figure 4.8 shows a typical series of I-, B- and P-pictures. In order to encode a B-picture, two neighbouring I- or P-pictures ('anchor' pictures or 'key' pictures) must be processed and stored in the prediction memory, introducing a delay of several frames into the encoding procedure. Before frame B_2 in Figure 4.8 can be encoded, its two 'anchor' frames I_1 and P_4 must be processed and stored, i.e. frames 1–4 must be processed before frames 2 and 3 can be coded. In this example, there is a delay of at least three frames during encoding (frames 2, 3 and 4 must be stored before B_2 can be coded) and this delay will be larger if more B-pictures are used.

In order to limit the delay at the decoder, encoded pictures are *reordered* before transmission, such that all the anchor pictures required to decode a B-picture are placed *before* the B-picture. Figure 4.9 shows the same series of frames, reordered prior to transmission. P_4 is now placed *before* B_2 and B_3. Decoding proceeds as shown in Table 4.1: P_4 is decoded immediately after I_1 and is stored by the decoder. B_2 and B_3 can now be decoded and displayed (because their prediction references, I_1 and P_4, are both available), after which P_4 is displayed. There is at most one frame delay between decoding and display and the decoder only needs to store two decoded frames. This is one example of 'asymmetry' between encoder and decoder: the delay and storage in the decoder are significantly lower than in the encoder.

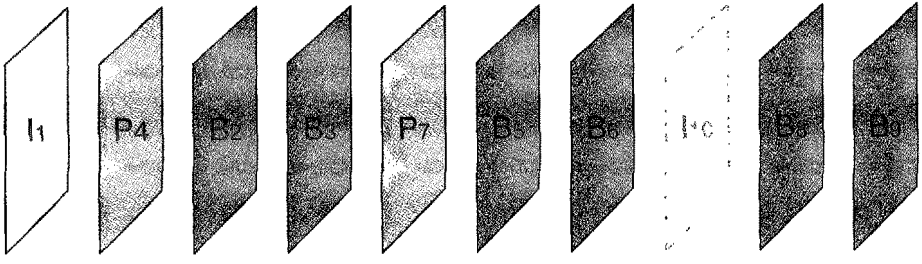


Figure 4.9 MPEG-1 group of pictures: transmission order

Table 4.1 MPEG-1 decoding and display order

Decode	Display
I_1	I_1
P_4	—
B_2	B_2
B_3	B_3
—	P_4
P_7	—
B_5	B_5
... etc.	... etc.

I-pictures are useful resynchronisation points in the coded bit stream: because it is coded without prediction, an I-picture may be decoded independently of any other coded pictures. This supports random access by a decoder (a decoder may start decoding the bit stream at any I-picture position) and error resilience (discussed in Chapter 11). However, an I-picture has poor compression efficiency because no temporal prediction is used. P-pictures provide better compression efficiency due to motion-compensated prediction and can be used as prediction references. B-pictures have the highest compression efficiency of each of the three picture types.

The MPEG-1 standard does not actually define the design of an encoder: instead, the standard describes the coded syntax and a hypothetical 'reference' decoder. In practice, the syntax and functionality described by the standard mean that a compliant encoder has to contain certain functions. The basic CODEC is similar to Figure 3.18. A 'front end' carries out motion estimation and compensation based on one reference frame (P-pictures) or two reference frames (B-pictures). The motion-compensated residual (or the original picture data in the case of an I-picture) is encoded using DCT, quantisation, run-level coding and variable-length coding. In an I- or P-picture, quantised transform coefficients are rescaled and transformed with the inverse DCT to produce a stored reference frame for further predicted P- or B-pictures. In the decoder, the coded data is entropy decoded, rescaled, inverse transformed and motion compensated. The most complex part of the CODEC is often the motion estimator because bidirectional motion estimation is computationally intensive. Motion estimation is only required in the encoder and this is another example of asymmetry between the encoder and decoder.

MPEG-1 syntax

The syntax of an MPEG-1 coded video sequence forms a hierarchy as shown in Figure 4.10. The levels or *layers* of the hierarchy are as follows.

Sequence layer This may correspond to a complete encoded video programme. The sequence starts with a *sequence header* that describes certain key information about the coded sequence including picture resolution and frame rate. The sequence consists of a series of *groups of pictures* (GOPs), the next layer of the hierarchy.

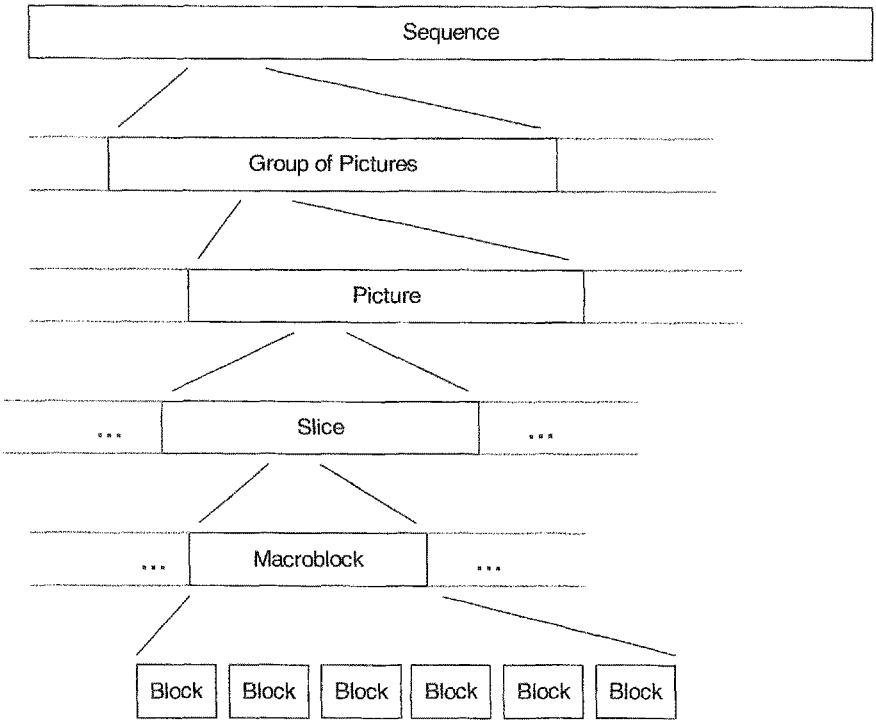


Figure 4.10 MPEG-1 synatx hierarchy

GOP layer A GOP is one I-picture followed by a series of P- and B-pictures (e.g. Figure 4.8). In Figure 4.8, the GOP contains nine pictures (one I, two P and six B) but many other GOP structures are possible, for example:

(a) All GOPs contain just one I-picture, i.e. no motion compensated prediction is used: this is similar to Motion JPEG.

(b) GOPs contain only I- and P-pictures, i.e. no bidirectional prediction is used: compression efficiency is relatively poor but complexity is low (since B-pictures are more complex to generate).

(c) Large GOPs: the proportion of I-pictures in the coded stream is low and hence compression efficiency is high. However, there are few synchronisation points which may not be ideal for random access and for error resilience.

(d) Small GOPs: there is a high proportion of I-pictures and so compression efficiency is low, however there are frequent opportunities for resynchronisation.

An encoder need not keep a consistent GOP structure within a sequence. It may be useful to vary the structure occasionally, for example by starting a new GOP when a scene change or cut occurs in the video sequence.

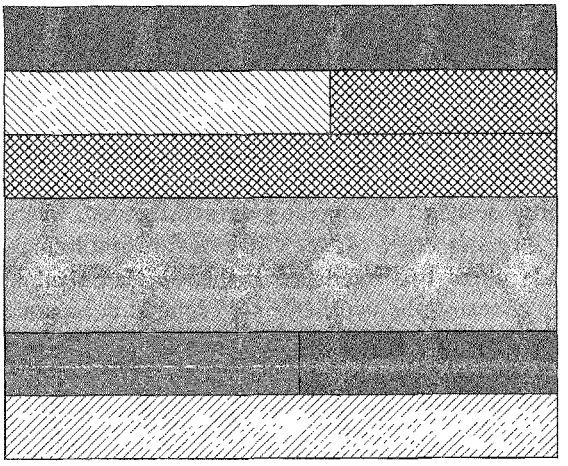


Figure 4.11 Example of MPEG-1 slices

Picture layer A picture defines a single coded frame. The picture header describes the type of coded picture (I, P, B) and a temporal reference that defines when the picture should be displayed in relation to the other pictures in the sequence.

Slice layer A picture is made up of a number of slices, each of which contains an integral number of macroblocks. In MPEG-1 there is no restriction on the size or arrangement of slices in a picture, except that slices should cover the picture in raster order. Figure 4.11 shows one possible arrangement: each shaded region in this figure is a single slice.

A slice starts with a slice header that defines its position. Each slice may be decoded independently of other slices within the picture and this helps the decoder to recover from transmission errors: if an error occurs within a slice, the decoder can always restart decoding from the next slice header.

Macroblock layer A slice is made up of an integral number of macroblocks, each of which consists of six blocks (Figure 4.6). The macroblock header describes the type of macroblock, motion vector(s) and defines which 8×8 blocks actually contain coded transform data. The picture type (I, P or B) defines the 'default' prediction mode for each macroblock, but individual macroblocks within P- or B-pictures may be intra-coded if required (i.e. coded without any motion-compensated prediction). This can be useful if no good match can be found within the search area in the reference frames since it may be more efficient to code the macroblock without any prediction.

Block layer A block contains variable-length code(s) that represent the quantised transform coefficients in an 8×8 block. Each DC coefficient (DCT coefficient [0, 0]) is coded differentially from the DC coefficient of the previous coded block, to exploit the fact that neighbouring blocks tend to have very similar DC (average) values. AC coefficients (all other coefficients) are coded as a (run, level) pair, where 'run' indicates the number of preceding zero coefficients and 'level' the value of a non-zero coefficient.

4.4.2 MPEG-2

The next important entertainment application for coded video (after CD-ROM storage) was digital television. In order to provide an improved alternative to analogue television, several key features were required of the video coding algorithm. It had to efficiently support larger frame sizes (typically 720 × 576 or 720 × 480 pixels for ITU-R 601 resolution) and coding of interlaced video. MPEG-1 was primarily designed to support progressive video, where each frame is scanned as a single unit in raster order. At television-quality resolutions, interlaced video (where a frame is made up of two interlaced 'fields' as described in Chapter 2) gives a smoother video image. Because the two fields are captured at separate time intervals (typically 1/50 or 1/60 of a second apart), better performance may be achieved by coding the fields separately.

MPEG-2 consists of three main sections: Video (described below), Audio[18] (based on MPEG-1 audio coding) and Systems[19] (defining, in more detail than MPEG-1 Systems, multiplexing and transmission of the coded audio/visual stream). MPEG-2 Video is (almost) a superset of MPEG-1 Video, i.e. most MPEG-1 video sequences should be decodeable by an MPEG-2 decoder. The main enhancements added by the MPEG-2 standard are as follows:

Efficient coding of television-quality video

The most important application of MPEG-2 is broadcast digital television. The 'core' functions of MPEG-2 (described as 'main profile/main level') are optimised for efficient coding of television resolutions at a bit rate of around 3–5 Mbps.

Support for coding of interlaced video

MPEG-2 video has several features that support flexible coding of interlaced video. The two fields that make up a complete interlaced frame can be encoded as separate pictures (*field pictures*), each of which is coded as an I-, P- or B-picture. P- and B- field pictures may be predicted from a field in another frame or from the other field in the current frame.

Alternatively, the two fields may be handled as a single picture (a *frame picture*) with the luminance samples in each macroblock of a frame picture arranged in one of two ways. *Frame DCT coding* is similar to the MPEG-1 structure, where each of the four luminance blocks contains alternate lines from both fields. With *field DCT coding*, the top two luminance blocks contain only samples from the top field, and the bottom two luminance blocks contain samples from the bottom field. Figure 4.12 illustrates the two coding structures.

In a field picture, the upper and lower 16 × 8 sample regions of a macroblock may be motion-compensated independently: hence each of the two regions has its own vector (or two vectors in the case of a B-picture). This adds an overhead to the macroblock because of the extra vector(s) that must be transmitted. However, this *16 × 8 motion compensation* mode can improve performance because a field picture has half the vertical resolution of a frame picture and so there are more likely to be significant differences in motion between the top and bottom halves of each macroblock.

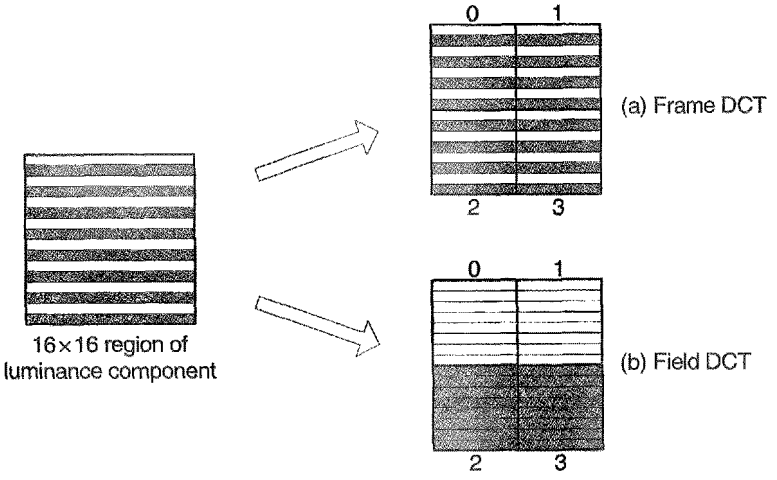


Figure 4.12 (a) Frame and (b) field DCT coding

In *dual-prime motion compensation* mode, the current field (within a field or frame picture) is predicted from the two fields of the reference frame using a single vector together with a transmitted correction factor. The correction factor modifies the motion vector to compensate for the small displacement between the two fields in the reference frame.

Scalability

The progressive modes of JPEG described earlier are forms of *scalable coding*. A scalable coded bit stream consists of a number of layers, a *base layer* and one or more *enhancement layers*. The base layer can be decoded to provide a recognisable video sequence that has a limited visual quality, and a higher-quality sequence may be produced by decoding the base layer plus enhancement layer(s), with each extra enhancement layer improving the quality of the decoded sequence. MPEG-2 video supports four scalable modes.

Spatial scalability This is analogous to hierarchical encoding in the JPEG standard. The base layer is coded at a low spatial resolution and each enhancement layer, when added to the base layer, gives a progressively higher spatial resolution.

Temporal scalability The base layer is encoded at a low temporal resolution (frame rate) and the enhancement layer (s) are coded to provide higher frame rate(s) (Figure 4.13). One application of this mode is stereoscopic video coding: the base layer provides a monoscopic 'view' and an enhancement layer provides a stereoscopic offset 'view'. By combining the two layers, a full stereoscopic image may be decoded.

SNR scalability In a similar way to the successive approximation mode of JPEG, the base layer is encoded at a 'coarse' visual quality (with high compression). Each enhancement layer, when added to the base layer, improves the video quality.

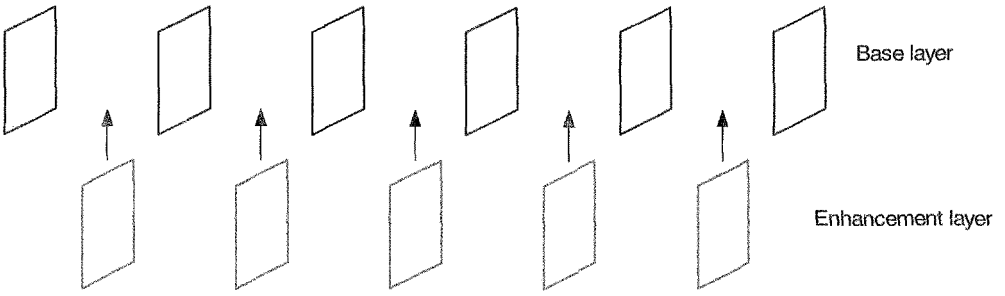


Figure 4.13 Temporal scalability

Data partitioning The coded sequence is partitioned into two layers. The base layer contains the most 'critical' components of the coded sequence such as header information, motion vectors and (optionally) low-frequency transform coefficients. The enhancement layer contains all remaining coded data (usually less critical to successful decoding).

These scalable modes may be used in a number of ways. A decoder may decode the current programme at standard ITU-R 601 resolution (720 × 576 pixels, 25 or 30 frames per second) by decoding just the base layer, whereas a 'high definition' decoder may decode one or more enhancement layer (s) to increase the temporal and/or spatial resolution. The multiple layers can support simultaneous decoding by 'basic' and 'advanced' decoders. Transmission of the base and enhancement layers is usually more efficient than encoding and sending separate bit streams at the lower and higher resolutions.

The base layer is the most 'important' to provide a visually acceptable decoded picture. Transmission errors in the base layer can have a catastrophic effect on picture quality, whereas errors in enhancement layer (s) are likely to have a relatively minor impact on quality. By protecting the base layer (for example using a separate transmission channel with a low error rate or by adding error correction coding), high visual quality can be maintained even when transmission errors occur (see Chapter 11).

Profiles and levels

Most applications require only a limited subset of the wide range of functions supported by MPEG-2. In order to encourage interoperability for certain 'key' applications (such as digital TV), the standard includes a set of recommended *profiles* and *levels* that each define a certain subset of the MPEG-2 functionalities. Each profile defines a set of *capabilities* and the important ones are as follows:

- *Simple*: 4:2:0 sampling, only I- and P-pictures are allowed. Complexity is kept low at the expense of poor compression performance.
- *Main*: This includes all of the core MPEG-2 capabilities including B-pictures and support for interlaced video. 4:2:0 sampling is used.
- *4:2:2*: As the name suggests, 4:2:2 subsampling is used, i.e. the Cr and Cb components have full vertical resolution and half horizontal resolution. Each macroblock contains eight blocks: four luminance, two Cr and two Cb.

- *SNR*: As 'main' profile, except that an enhancement layer is added to provide higher visual quality.
- *Spatial*: As 'SNR' profile, except that spatial scalability may also be used to provide higher-quality enhancement layers.
- *High*: As 'Spatial' profile, with the addition of support for 4 : 2 : 2 sampling.

Each level defines *spatial and temporal resolutions*:

- *Low*: Up to 352 × 288 frame resolution and up to 30 frames per second.
- *Main*: Up to 720 × 576 frame resolution and up to 30 frames per second.
- *High–1440*: Up to 1440 × 1152 frame resolution and up to 60 frames per second.
- *High*: Up to 1920 × 1152 frame resolution and up to 60 frames per second.

The MPEG-2 standard defines certain recommended combinations of profiles and levels. *Main profile / low level* (using only frame encoding) is essentially MPEG-1. *Main profile / main level* is suitable for broadcast digital television and this is the most widely used profile / level combination. *Main profile / high level* is suitable for high-definition television (HDTV). (Originally, the MPEG working group intended to release a further standard, MPEG-3, to support coding for HDTV applications. However, once it became clear that the MPEG-2 syntax could deal with this application adequately, work on this standard was dropped and so there is no MPEG-3 standard.)

In addition to the main features described above, there are some further changes from the MPEG-1 standard. Slices in an MPEG-2 picture are constrained such that they may not overlap from one row of macroblocks to the next (unlike MPEG-1 where a slice may occupy multiple rows of macroblocks). D-pictures in MPEG-1 were felt to be of limited benefit and are not supported in MPEG-2.

4.4.3 MPEG-4

The MPEG-1 and MPEG-2 standards deal with complete video frames, each coded as a single unit. The MPEG-4 standard[6] was developed with the aim of extending the capabilities of the earlier standards in a number of ways.

Support for low bit-rate applications MPEG-1 and MPEG-2 are reasonably efficient for coded bit rates above around 1 Mbps. However, many emerging applications (particularly Internet-based applications) require a much lower transmission bit rate and MPEG-1 and 2 do not support efficient compression at low bit rates (tens of kbps or less).

Support for object-based coding Perhaps the most fundamental shift in the MPEG-4 standard has been towards *object-based* or *content-based* coding, where a video scene can be handled as a set of foreground and background *objects* rather than just as a series of rectangular frames. This type of coding opens up a wide range of possibilities, such as independent coding of different objects in a scene, reuse of scene components, compositing

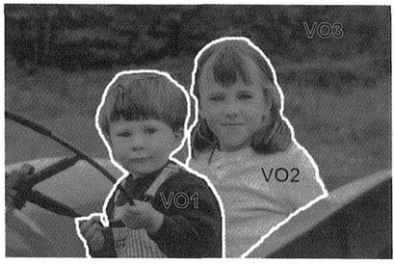


Figure 4.14 Video scene showing multiple video objects

(where objects from a number of sources are combined into a scene) and a high degree of interactivity. The basic concept used in MPEG-4 Visual is that of the *video object* (VO). A video scene (VS) (a sequence of video frames) is made up of a number of VOs. For example, the VS shown in Figure 4.14 consists of a background VO and two foreground VOs. MPEG-4 provides tools that enable each VO to be coded independently, opening up a range of new possibilities. The equivalent of a 'frame' in VO terms, i.e. a 'snapshot' of a VO at a single instant in time, is a *video object plane* (VOP). The entire scene may be coded as a single, rectangular VOP and this is equivalent to a picture in MPEG-1 and MPEG-2 terms.

Toolkit-based coding MPEG-1 has a very limited degree of flexibility; MPEG-2 introduced the concept of a 'toolkit' of profiles and levels that could be combined in different ways for various applications. MPEG-4 extends this towards a highly flexible set of coding tools that enable a range of applications as well as a standardised framework that allows new tools to be added to the 'toolkit'.

The MPEG-4 standard is organised so that new coding tools and functionalities may be added incrementally as new versions of the standard are developed, and so the list of tools continues to grow. However, the main tools for coding of video images can be summarised as follows.

MPEG-4 Visual: very low bit-rate video core

The video coding algorithms that form the 'very low bit-rate video (VLBV) core' of MPEG-4 Visual are almost identical to the baseline H.263 video coding standard (Chapter 5). If the *short header* mode is selected, frame coding is completely identical to baseline H.263. A video sequence is coded as a series of rectangular frames (i.e. a single VOP occupying the whole frame).

Input format Video data is expected to be pre-processed and converted to one of the picture sizes listed in Table 4.2, at a frame rate of up to 30 frames per second and in 4 : 2 : 0 Y : Cr : Cb format (i.e. the chrominance components have half the horizontal and vertical resolution of the luminance component).

Picture types Each frame is coded as an I- or P-frame. An I-frame contains only intra-coded macroblocks, whereas a P-frame can contain either intra- or inter-coded macroblocks.

Table 4.2 MPEG4 VLBV/H.263 picture sizes

Format	Picture size (luminance)
SubQCIF	128 × 96
QCIF	176 × 144
CIF	352 × 288
4CIF	704 × 576
16CIF	1408 × 1152

Motion estimation and compensation This is carried out on 16 × 16 macroblocks or (optionally) on 8 × 8 macroblocks. Motion vectors can have half-pixel resolution.

Transform coding The motion-compensated residual is coded with DCT, quantisation, zigzag scanning and run–level coding.

Variable-length coding The run–level coded transform coefficients, together with header information and motion vectors, are coded using variable-length codes. Each non-zero transform coefficient is coded as a combination of *run, level, last* (where 'last' is a flag to indicate whether this is the last non-zero coefficient in the block) (see Chapter 8).

Syntax

The syntax of an MPEG-4 (VLBV) coded bit stream is illustrated in Figure 4.15.

Picture layer The highest layer of the syntax contains a complete coded picture. The picture header indicates the picture resolution, the type of coded picture (inter or intra) and includes a temporal reference field. This indicates the correct display time for the decoder (relative to other coded pictures) and can help to ensure that a picture is not displayed too early or too late.

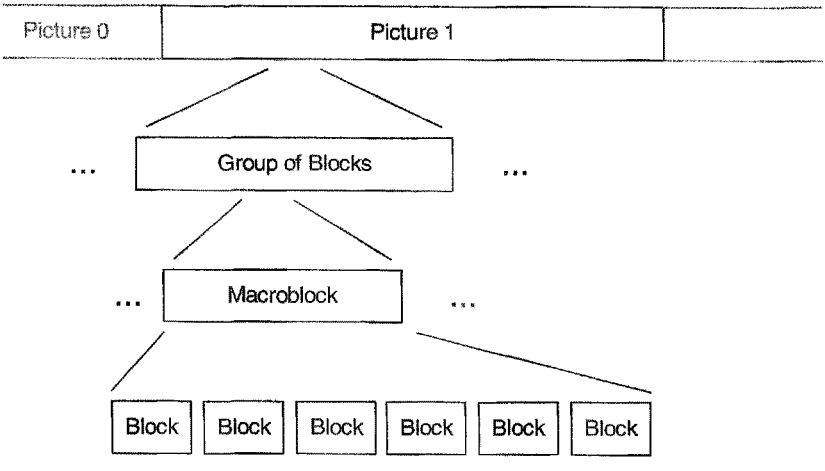


Figure 4.15 MPEG-4/H.263 layered syntax

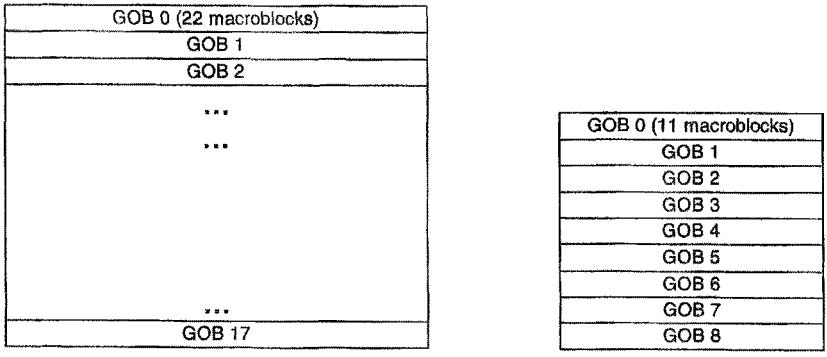


Figure 4.16 GOBs: (a) CIF and (b) QCIF pictures

Group of blocks layer A group of blocks (GOB) consists of one complete row of macroblocks in SQCIF, QCIF and CIF pictures (two rows in a 4CIF picture and four rows in a 16CIF picture). GOBs are similar to slices in MPEG-1 and MPEG-2 in that, if an optional GOB header is inserted in the bit stream, the decoder can resynchronise to the start of the next GOB if an error occurs. However, the size and layout of each GOB are fixed by the standard (unlike slices). The arrangement of GOBs in a QCIF and CIF picture is shown in Figure 4.16.

Macroblock layer A macroblock consists of four luminance blocks and two chrominance blocks. The macroblock header includes information about the type of macroblock, 'coded block pattern' (indicating which of the six blocks actually contain transform coefficients) and coded horizontal and vertical motion vectors (for inter-coded macroblocks).

Block layer A block consists of run–level coded coefficients corresponding to an 8×8 block of samples.

The core CODEC (based on H.263) was designed for efficient coding at low bit rates. The use of 8×8 block motion compensation and the design of the variable-length coding tables make the VLBV MPEG-4 CODEC more efficient than MPEG-1 or MPEG-2 (see Chapter 5 for a comparison of coding efficiency).

Other visual coding tools

The features that make MPEG-4 (Visual) unique among the coding standards are the range of further coding tools available to the designer.

Shape coding Shape coding is required to specify the boundaries of each non-rectangular VOP in a scene. Shape information may be *binary* (i.e. identifying the pixels that are internal to the VOP, described as 'opaque', or external to the VOP, described as 'transparent') or *grey scale* (where each pixel position within a VOP is allocated an 8-bit 'grey scale' number that identifies the transparency of the pixel). Grey scale information is more complex and requires more bits to code: however, it introduces the possibility of overlapping, semi-transparent VOPs (similar to the concept of 'alpha planes' in computer graphics). Binary information is simpler to code because each pixel has only two possible states, opaque or transparent. Figure 4.17

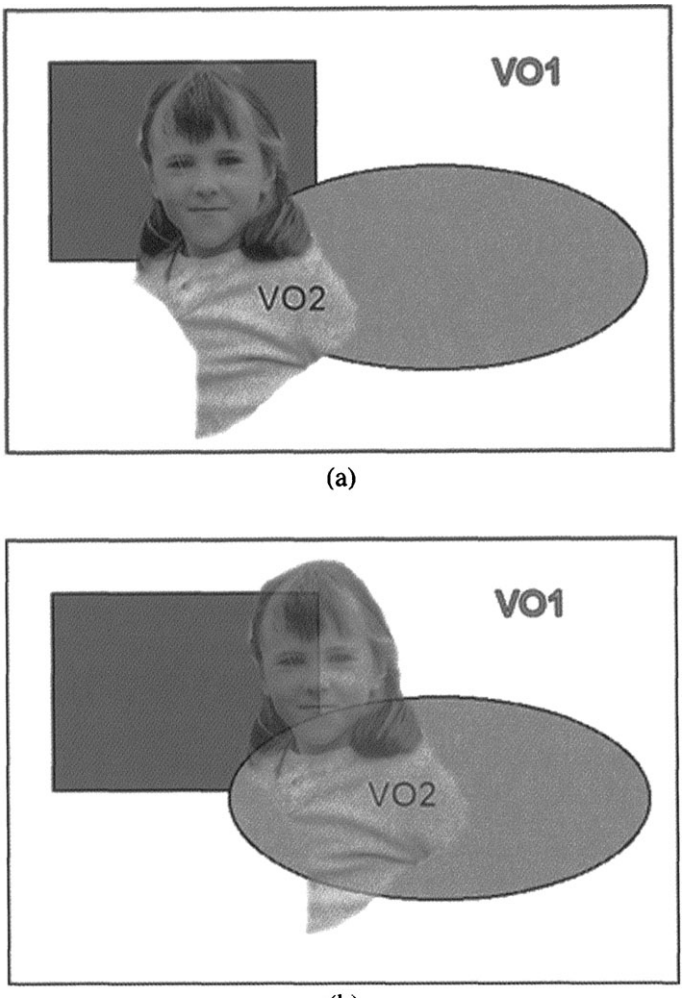


Figure 4.17 (a) Opaque and (b) semi-transparent VOPs

illustrates the concept of opaque and semi-transparent VOPs: in image (a), VOP2 (foreground) is opaque and completely obscures VOP1 (background), whereas in image (b) VOP2 is partly transparent.

Binary shape information is coded in 16 × 16 blocks (binary alpha blocks, BABs). There are three possibilities for each block:

1. All pixels are transparent, i.e. the block is 'outside' the VOP. No shape (or texture) information is coded.
2. All pixels are opaque, i.e. the block is fully 'inside' the VOP. No shape information is coded: the pixel values of the block ('texture') are coded as described in the next section.

3. Some pixels are opaque and some are transparent, i.e. the block crosses a boundary of the VOP. The binary shape values of each pixel (1 or 0) are coded using a form of DPCM and the texture information of the opaque pixels is coded as described below.

Grey scale shape information produces values in the range 0 (transparent) to 255 (opaque) that are compressed using block-based DCT and motion compensation.

Motion compensation Similar options exist to the I-, P- and B-pictures in MPEG-1 and MPEG-2:

1. I-VOP: VOP is encoded without any motion compensation.
2. P-VOP: VOP is predicted using motion-compensated prediction from a past I- or P-VOP.
3. B-VOP: VOP is predicted using motion-compensated prediction from a past and a future I- or P-picture (with forward, backward or bidirectional prediction).

Figure 4.18 shows mode (3), prediction of a B-VOP from a previous I-VOP and future P-VOP. For macroblocks (or 8×8 blocks) that are fully contained within the current and reference VOPs, block-based motion compensation is used in a similar way to MPEG-1 and MPEG-2. The motion compensation process is modified for blocks or macroblocks along the boundary of the VOP. In the reference VOP, pixels in the 16×16 (or 8×8) search area are padded based on the pixels along the edge of the VOP. The macroblock (or block) in the current VOP is matched with this search area using block matching: however, the difference value (mean absolute error or sum of absolute errors) is only computed for those pixel positions that lie within the VOP.

Texture coding Pixels (or motion-compensated residual values) within a VOP are coded as 'texture'. The basic tools are similar to MPEG-1 and MPEG-2: transform using the DCT, quantisation of the DCT coefficients followed by reordering and variable-length coding. To further improve compression efficiency, quantised DCT coefficients may be *predicted* from previously transmitted blocks (similar to the differential prediction of DC coefficients used in JPEG, MPEG-1 and MPEG-2).

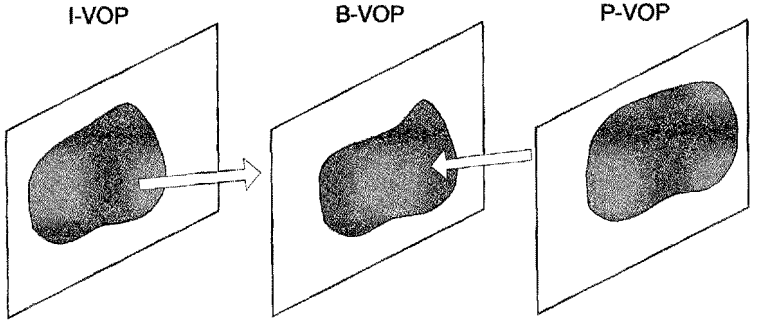


Figure 4.18 B-VOP motion-compensated prediction

A macroblock that covers a boundary of the VOP will contain both opaque and transparent pixels. In order to apply a regular 8 × 8 DCT, it is necessary to use 'padding' to fill up the transparent pixel positions. In an inter-coded VOP, where the texture information is motion-compensated residual data, the transparent positions are simply filled with zeros. In an intra-coded VOP, where the texture is 'original' pixel data, the transparent positions are filled by extrapolating the pixel values along the boundary of the VOP.

Error resilience MPEG-4 incorporates a number of mechanisms that can provide improved performance in the presence of transmission errors (such as bit errors or lost packets). The main tools are:

1. Synchronisation markers: similar to MPEG-1 and MPEG-2 slice start codes, except that these may optionally be positioned so that each resynchronisation interval contains an approximately equal number of encoded bits (rather than a constant number of macroblocks). This means that errors are likely to be evenly distributed among the resynchronisation intervals. Each resynchronisation interval may be transmitted in a separate *video packet*.
2. Data partitioning: similar to the data partitioning mode of MPEG-2.
3. Header extension: redundant copies of header information are inserted at intervals in the bit stream so that if an important header (e.g. a picture header) is lost due to an error, the redundant header may be used to partially recover the coded scene.
4. Reversible VLCs: these variable length codes limit the propagation ('spread') of an errored region in a decoded frame or VOP and are described further in Chapter 8.

Scalability MPEG-4 supports spatial and temporal scalability. Spatial scalability applies to rectangular VOPs in a similar way to MPEG-2: the base layer gives a low spatial resolution and an enhancement layer may be decoded together with the base layer to give a higher resolution. Temporal scalability is extended beyond the MPEG-2 approach in that it may be applied to individual VOPs. For example, a background VOP may be encoded without scalability, whilst a foreground VOP may be encoded with several layers of temporal scalability. This introduces the possibility of decoding a foreground object at a higher frame rate and more static, background objects at a lower frame rate.

Sprite coding A 'sprite' is a VOP that is present for the entire duration of a video sequence (VS). A sprite may be encoded and transmitted once at the start of the sequence, giving a potentially large benefit in compression performance. A good example is a background sprite: the background image to a scene is encoded as a sprite at the start of the VS. For the remainder of the VS, only the foreground VOPs need to be coded and transmitted since the decoder can 'render' the background from the original sprite. If there is camera movement (e.g. panning), then a sprite that is larger than the visible scene is required (Figure 4.19). In order to compensate for more complex camera movements (e.g. zoom or rotation), it may be necessary for the decoder to 'warp' the sprite. A sprite is encoded as an I-VOP as described earlier.

Static texture An alternative set of tools to the DCT may be used to code 'static' texture, i.e. texture data that does not change rapidly. The main application for this is to code texture

Figure 4.19 Example of background sprite and foreground VOPs

that is mapped onto a 2-D or 3-D surface (described below). Static image texture is coded efficiently using a wavelet transform. The transform coefficients are quantised and coded with a zero-tree algorithm followed by arithmetic coding. Wavelet coding is described further in Chapter 7 and arithmetic coding in Chapter 8.

Mesh and 3-D model coding MPEG-4 supports more advanced object-based coding techniques including:

- 2-D mesh coding, where an object is coded as a mesh of triangular patches in a 2-D plane. Static texture (coded as described above) can be mapped onto the mesh. A moving object can be represented by deforming the mesh and warping the texture as the mesh moves.
- 3-D mesh coding, where an object is described as a mesh in 3-D space. This is more complex than a 2-D mesh representation but gives a higher degree of flexibility in terms of representing objects within a scene.
- Face and body model coding, where a human face or body is rendered at the decoder according to a face or body model. The model is controlled (moved) by changing 'animation parameters'. In this way a 'head-and-shoulders' video scene may be coded by sending only the animation parameters required to 'move' the model at the decoder. Static texture is mapped onto the model surface.

These three tools offer the potential for fundamental improvements in video coding performance and flexibility: however, their application is currently limited because of the high processing resources required to analyse and render even a very simple scene.

MPEG-4 visual profiles and levels

In common with MPEG-2, a number of recommended 'profiles' (sets of MPEG-4 tools) and 'levels' (constraints on bit stream parameters such as frame size and rate) are defined in the

MPEG-4 standard. Each profile is defined in terms of one or more 'object types', where an object type is a subset of the MPEG-4 tools. Table 4.3 lists the main MPEG-4 object types that make up the profiles. The 'Simple' object type contains tools for coding of basic I- and P-rectangular VOPs (complete frames) together with error resilience tools and the 'short header' option (for compatibility with H.263). The 'Core' type adds B-VOPs and basic shape coding (using a binary shape mask only). The main profile adds grey scale shape coding and sprite coding.

MPEG-4 (Visual) is gaining popularity in a number of application areas such as Internet-based video. However, to date the majority of applications use only the simple object type and there has been limited take-up of the content-based features of the standard. This is partly because of technical complexities (for example, it is difficult to accurately segment a video scene into foreground and background objects, e.g. Figure 4.14, using an automatic algorithm) and partly because useful applications for content-based video coding and manipulation have yet to emerge. At the time of writing, the great majority of video coding applications continue to work with complete rectangular frames. However, researchers continue to improve algorithms for segmenting and manipulating video objects.[20–25] The content-based tools have a number of interesting possibilities: for example, they make it

Table 4.3 MPEG-4 video object types

	Video object types							
Visual tools	Simple	Core	Main	Simple scalable	Animated 2-D mesh	Basic animated texture	Still scalable texture	Simple face
Basic (I-VOP, P-VOP, coefficient prediction, 16 × 16 and 8 × 8 motion vectors)	✓	✓	✓	✓	✓			
Error resilience	✓	✓	✓	✓	✓			
Short header	✓	✓	✓		✓			
B-VOP		✓	✓	✓	✓			
P-VOP with overlapped block matching								
Alternative quantisation		✓	✓		✓			
P-VOP based temporal scalability		✓	✓		✓			
Binary shape		✓	✓		✓	✓		
Grey shape			✓					
Interlaced video coding			✓					
Sprite			✓					
Rectangular temporal scalability				✓				
Rectangular spatial scalability				✓				
Scalable still texture					✓	✓	✓	
2-D mesh					✓	✓		
Facial animation parameters								✓

possible to develop 'hybrid' applications with a mixture of 'real' video objects (possibly from a number of different sources) and computer-generated graphics. So-called synthetic natural hybrid coding has the potential to enable a new generation of video applications.

4.5 SUMMARY

The ISO has issued a number of image and video coding standards that have heavily influenced the development of the technology and market for video coding applications. The original JPEG still image compression standard is now a ubiquitous method for storing and transmitting still images and has gained some popularity as a simple and robust algorithm for video compression. The improved subjective and objective performance of its successor, JPEG-2000, may lead to the gradual replacement of the original JPEG algorithm.

The first MPEG standard, MPEG-1, was never a market success in its target application (video CDs) but is widely used for PC and internet video applications and formed the basis for the MPEG-2 standard. MPEG-2 has enabled a worldwide shift towards digital television and is probably the most successful of the video coding standards in terms of market penetration. The MPEG-4 standard offers a plethora of video coding tools which may in time enable many new applications: however, at the present time the most popular element of MPEG-4 (Visual) is the 'core' low bit rate CODEC that is based on the ITU-T H.263 standard. In the next chapter we will examine the H.26x series of coding standards, H.261, H.263 and the emerging H.26L.

REFERENCES

1. http://www.itu.int/ [International Telecommunication Union].
2. http://www.iso.ch/ [International Standards Organisation].
3. ISO/IEC 10918-1/ITU-T Recommendation T.81, 'Digital compression and coding of continuous-tone still images', 1992 [JPEG].
4. ISO/IEC 11172-2, 'Information technology–coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s–part 2: Video', 1993 [MPEG1 Video].
5. ISO/IEC 13818-2, 'Information technology: generic coding of moving pictures and associated audio information: Video', 1995 [MPEG2 Video].
6. ISO/IEC 14996-2, 'Information technology–coding of audio-visual objects–part 2: Visual', 1998 [MPEG-4 Visual].
7. ISO/IEC FCD 15444-1, 'JPEG2000 Final Committee Draft v1.0', March 2000.
8. ISO/IEC JTC1/SC29/WG11 N4031, 'Overview of the MPEG-7 Standard', Singapore, March 2001.
9. ISO/IEC JTC1/SC29/WG11 N4318, 'MPEG-21 Overview', Sydney, July 2001.
10. http://standards.pictel.com/ftp/video-site/ [VCEG working documents].
11. http://www.cselt.it/mpeg/ [MPEG committee official site].
12. http://www.jpeg.org/ [JPEG resources].
13. http://www.mpeg.org/ [MPEG resources].
14. ITU-T Q6/SG16 Draft Document, 'Appendix III for ITU-T Rec H.263', Porto Seguro, May 2001.
15. A. N. Skodras, C. A. Christopoulos and T. Ebrahimi, 'JPEG2000: The upcoming still image compression standard', *Proc. 11th Portuguese Conference on Pattern Recognition*, Porto, 2000.
16. ISO/IEC 11172-1, 'Information technology–coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s–part 1: Systems', 1993 [MPEG1 Systems].

17. ISO/IEC 11172-2, Information technology–coding of moving pictures and associated audio for digital storage mediat at up to about 1.5 Mbit/s–part 3: Audio', 1993 [MPEG1 Audio].
18. ISO/IEC 13818-3, 'Information technology: generic coding of moving pictures and associated audio information: Audio', 1995 [MPEG2 Audio].
19. ISO/IEC 13818-1, 'Information technology: generic coding of moving pictures and associated audio information Systems', 1995 [MPEG2 Systems].
20. P. Salembier and F. Marqués, 'Region-based representations of image and video: segmentation tools for multimedia services', *IEEE Trans. CSVT* **9**(8), December 1999.
21. L. Garrido, A. Oliveras and P. Salembier, 'Motion analysis of image sequences using connected operators', *Proc. VCIP97*, San Jose, February 1997, *SPIE* **3024**.
22. K. Illgner and F. Müller, 'Image segmentation using motion estimation', in *Time-varying Image Processing and Image Recognition*, Elsevier Science, 1997.
23. R. Castagno and T. Ebrahimi, 'Video Segmentation based on multiple features for interactive multimedia applications', *IEEE Trans. CSVT* **8**(5), September, 1998.
24. E. Steinbach, P. Eisert and B. Girod, 'Motion-based analysis and segmentation of image sequences using 3-D scene models', *Signal Processing*, **66**(2), April 1998.
25. M. Chang, M. Teklap and M. Ibrahim Sezan, 'Simultaneous motion estimation and segmentation', *IEEE Trans. Im. Proc.*, **6**(9), 1997.

This Page Intentionally Left Blank

5

Video Coding Standards: H.261, H.263 and H.26L

5.1 INTRODUCTION

The ISO MPEG video coding standards are aimed at storage and distribution of video for entertainment and have tried to meet the needs of providers and consumers in the 'media industries'. The ITU has (historically) been more concerned about the telecommunications industry, and its video coding standards (H.261, H.263, H.26L) have consequently been targeted at real-time, point-to-point or multi-point communications.

The first ITU-T video coding standard to have a significant impact, H.261, was developed during the late 1980s/early 1990s with a particular application and transmission channel in mind. The application was video conferencing (two-way communications via a video 'link') and the channel was N-ISDN. ISDN provides a constant bit rate of $p \times 64$ kbps, where p is an integer in the range 1–30: it was felt at the time that ISDN would be the medium of choice for video communications because of its guaranteed bandwidth and low delay. Modem channels over the analogue POTS/PSTN (at speeds of less than 9600 bps at the time) were considered to be too slow for visual communications and packet-based transmission was not considered to be reliable enough.

H.261 was quite successful and continues to be used in many legacy video conferencing applications. Improvements in processor performance, video coding techniques and the emergence of analogue Modems and Internet Protocol (IP) networks as viable channels led to the development of its successor, H.263, in the mid-1990s. By making a number of improvements to H.261, H.263 provided significantly better compression performance as well as greater flexibility. The original H.263 standard (Version 1) had four optional modes which could be switched on to improve performance (at the expense of greater complexity). These modes were considered to be useful and Version 2 ('H.263+') added 12 further optional modes. The latest (and probably the last) version (v3) will contain a total of 19 modes, each offering improved coding performance, error resilience and/or flexibility.

Version 3 of H.263 has become a rather unwieldy standard because of the large number of options and the need to continue to support the basic ('baseline') CODEC functions. The latest initiative of the ITU-T experts group VCEG is the H.26L standard (where 'L' stands for 'long term'). This is a new standard that makes use of some of the best features of H.263 and aims to improve compression performance by around 50% at lower bit rates. Early indications are that H.26L will outperform H.263+ (but possibly not by 50%).

5.2 H.261[1]

Typical operating bit rates for H.261 applications are between 64 and 384 kbps. At the time of development, packet-based transmission over the Internet was not expected to be a significant requirement, and the limited video compression performance achievable at the time was not considered to be sufficient to support bit rates below 64 kbps.

A typical H.261 CODEC is very similar to the 'generic' motion-compensated DCT-based CODEC described in Chapter 3. Video data is processed in 4:2:0 Y:Cr:Cb format. The basic unit is the 'macroblock', containing four luminance blocks and two chrominance blocks (each 8×8 samples) (see Figure 4.6). At the input to the encoder, 16×16 macroblocks may be (optionally) motion compensated using integer motion vectors. The motion-compensated residual data is coded with an 8×8 DCT followed by quantisation and zigzag reordering. The reordered transform coefficients are run–level coded and compressed with an entropy encoder (see Chapter 8).

Motion compensation performance is improved by use of an optional *loop filter*, a 2-D spatial filter that operates on each 8×8 block in a macroblock prior to motion compensation (if the filter is switched on). The filter has the effect of 'smoothing' the reference picture which can help to provide a better prediction reference. Chapter 9 discusses loop filters in more detail (see for example Figures 9.11 and 9.12).

In addition, a forward error correcting code is defined in the standard that should be inserted into the transmitted bit stream. In practice, this code is often omitted from practical implementations of H.261: the error rate of an ISDN channel is low enough that error correction is not normally required, and the code specified in the standard is not suitable for other channels (such as a noisy wireless channel or packet-based transmission).

Each macroblock may be coded in 'intra' mode (no motion-compensated prediction) or 'inter' mode (with motion-compensated prediction). Only two frame sizes are supported, CIF (352×288 pixels) and QCIF (176×144 pixels).

H.261 was developed at a time when hardware and software processing performance was limited and therefore has the advantage of low complexity. However, its disadvantages include poor compression performance (with poor video quality at bit rates of under about 100 kbps) and lack of flexibility. It has been superseded by H.263, which has higher compression efficiency and greater flexibility, but is still widely used in installed video conferencing systems.

5.3 H.263[2]

In developing the H.263 standard, VCEG aimed to improve upon H.261 in a number of areas. By taking advantage of developments in video coding algorithms and improvements in processing performance, it provides better compression. H.263 provides greater flexibility than H.261: for example, a wider range of frame sizes is supported (listed in Table 4.2). The first version of H.263 introduced four optional modes, each described in an annex to the standard, and further optional modes were introduced in Version 2 of the standard ('H.263+'). The target application of H.263 is low-bit-rate, low-delay two-way video communications. H.263 can support video communications at bit rates below 20 kbps (at a very limited visual quality) and is now widely used both in 'established' applications such as video telephony and video conferencing and an increasing number of new applications (such as Internet-based video).

5.3.1 Features

The baseline H.263 CODEC is functionally identical to the MPEG-4 'short header' CODEC described in Section 4.4.3. Input frames in 4 : 2 : 0 format are motion compensated (with half-pixel resolution motion vectors), transformed with an 8 × 8 DCT, quantised, reordered and entropy coded. The main factors that contribute to the improved coding performance over H.261 are the use of half-pixel motion vectors (providing better motion compensation) and redesigned variable-length code (VLC) tables (described further in Chapter 8). Features such as I- and P-pictures, more frame sizes and optional coding modes give the designer greater flexibility to deal with different application requirements and transmission scenarios.

5.4 THE H.263 OPTIONAL MODES/H.263+

The original H.263 standard (Version 1) included four optional coding modes (Annexes D, E, F and G). Version 2 of the standard added 12 further modes (Annexes I to T) and a new release is scheduled with yet more coding modes (Annexes U, V and W). CODECs that implement some of the optional modes are sometimes described as 'H.263+' or 'H.263++' CODECs depending on which modes are implemented.

Each mode adds to or modifies the functionality of H.263, usually at the expense of increased complexity. An H.263-compliant CODEC must support the 'baseline' syntax described above: the use of optional modes may be negotiated between an encoder and a decoder prior to starting a video communications session. The optional modes have a number of potential benefits: some of the modes improve compression performance, others improve error resilience or provide tools that are useful for particular transmission environments such as packet-based transmission.

Annex D, Unrestricted motion vectors The optional mode described in Annex D of H.263 allows motion vectors to point outside the boundaries of the picture. This can provide a coding performance gain, particularly if objects are moving into or out of the picture. The pixels at the edges of the picture are extrapolated to form a 'border' outside the picture that vectors may point to (Figure 5.1). In addition, the motion vector range is extended so that

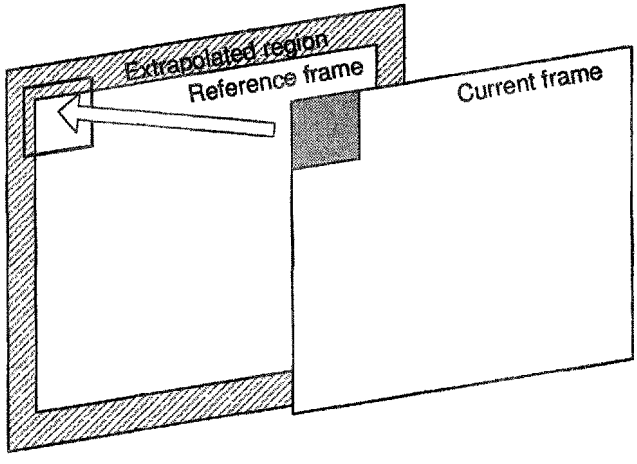


Figure 5.1 Unrestricted motion vectors

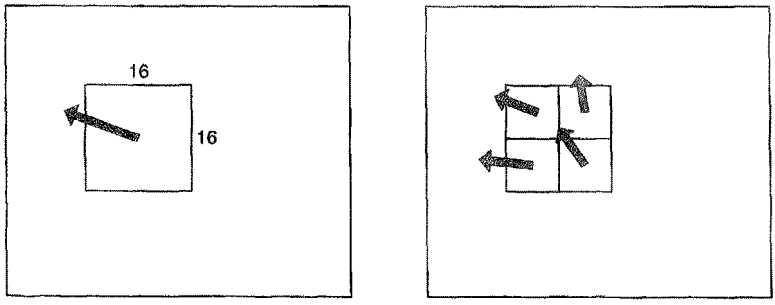


Figure 5.2 One or four motion vectors per macroblock

longer vectors are allowed. Finally, Annex D contains an optional alternative set of VLCs for encoding motion vector data. These VLCs are reversible, making it easier to recover from transmission errors (see Chapter 11).

Annex E, Syntax-based arithmetic coding Arithmetic coding is used instead of variable-length coding. Each of the VLCs defined in the standard is replaced with a probability value that is used by an arithmetic coder (see Chapter 8).

Annex F, Advanced prediction The efficiency of motion estimation and compensation is improved by allowing the use of four vectors per macroblock (a separate motion vector for each 8×8 luminance block, Figure 5.2). Overlapped block motion compensation (described in Chapter 6) is used to improve motion compensation and reduce 'blockiness' in the decoded image. Annex F requires the CODEC to support unrestricted motion vectors (Annex D).

Annex G, PB-frames A PB-frame is a pair of frames coded as a combined unit. The first frame is coded as a 'B-picture' and the second as a P-picture. The P-picture is forward predicted from the previous I- or P-picture and the B-picture is bidirectionally predicted from the previous and current I- or P-pictures. Unlike MPEG-1 (where a B-picture is coded as a separate unit), each macroblock of the PB-frame contains data from both the P-picture and the B-picture (Figure 5.3). PB-frames can give an improvement in compression efficiency.

Annex I, Advanced intra-coding This mode exploits the correlation between DCT coefficients in neighbouring intra-coded blocks in an image. The DC coefficient and the first row or column of AC coefficients may be predicted from the coefficients of neighbouring blocks (Figure 5.4). The zigzag scan, quantisation procedure and variable-length code tables are modified and the result is an improvement in compression efficiency for intra-coded macroblocks.

Annex J, Deblocking filter The edges of each 8×8 block are 'smoothed' using a spatial filter (described in Chapter 9). This reduces 'blockiness' in the decoded picture and also improves motion compensation performance. When the deblocking filter is switched on, four

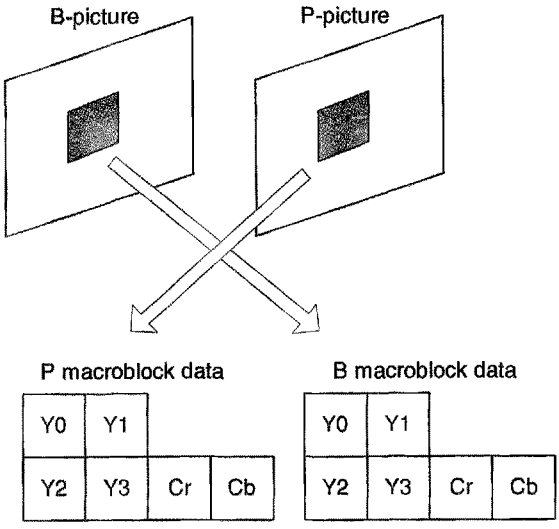


Figure 5.3 Macroblock in PB-frame

motion vectors per macroblock and unrestricted motion vectors are also enabled (Annexes D and F).

Annex K, Slice structured mode This mode provides support for resynchronisation intervals that are similar to MPEG-1 'slices'. A slice is a series of coded macroblocks

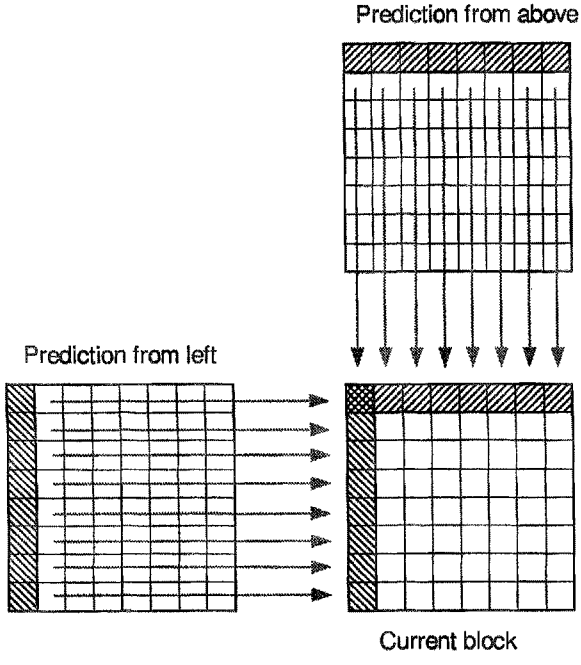


Figure 5.4 Prediction of intra-coefficients, H.263 Annex I

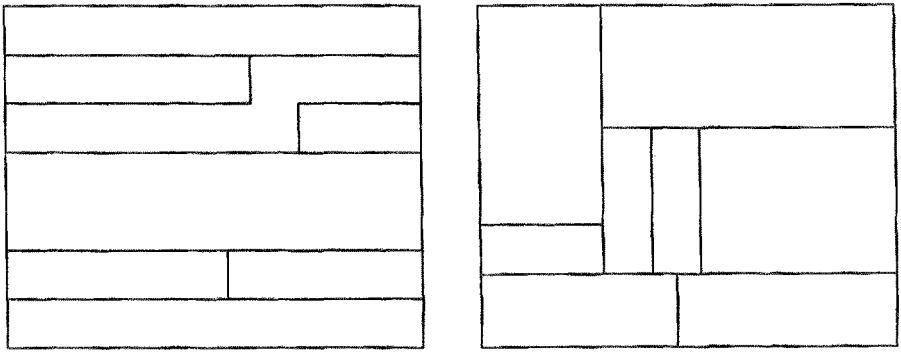


Figure 5.5 H.263 Annex K: slice options

starting with a slice header. Slices may contain macroblocks in raster order, or in any rectangular region of the picture (Figure 5.5). Slices may optionally be sent in an arbitrary order. Each slice may be decoded independently of any other slice in the picture and so slices can be useful for error resilience (see Chapter 11) since an error in one slice will not affect the decoding of any other slice.

Annex L, Supplemental enhancement information This annex contains a number of supplementary codes that may be sent by an encoder to a decoder. These codes indicate display-related information about the video sequence, such as picture freeze and timing information.

Annex M, Improved PB-frames As the name suggests, this is an improved version of the original PB-frames mode (Annex G). Annex M adds the options of forward or backward prediction for the B-frame part of each macroblock (as well as the bidirectional prediction defined in Annex G), resulting in improved compression efficiency.

Annex N, Reference picture selection This mode enables an encoder to choose from a number of previously coded pictures for predicting the current picture. The use of this mode to limit error propagation in a noisy transmission environment is discussed in Chapter 11. At the start of each GOB or slice, the encoder may choose the preferred reference picture for prediction of macroblocks in that GOB or slice.

Annex O, Scalability Temporal, spatial and SNR scalability are supported by this optional mode. In a similar way to the MPEG-2 optional scalability modes, spatial scalability increases frame resolution, SNR scalability increases picture quality and temporal scalability increases frame rate. In each case, a 'base layer' provides basic performance and the increased performance is obtained by decoding the base layer together with an 'enhancement layer'. Temporal scalability is particularly useful because it supports B-pictures: these are similar to the 'true' B-pictures in the MPEG standards (where a B-picture is a separate coded unit) and are more flexible than the combined PB-frames described in Annexes G and M.

Annex P, Reference picture resampling The prediction reference frame used by the encoder and decoder may be resampled prior to motion compensation. This has several possible applications. For example, an encoder can change the frame resolution 'on the fly' whilst continuing to use motion-compensated prediction. The prediction reference frame is resampled to match the new resolution and the current frame can then be predicted from the resampled reference. This mode may also be used to support *warping*, i.e. the reference picture is warped (deformed) prior to prediction, perhaps to compensate for nonlinear camera movements such as zoom or rotation.

Annex Q, Reduced resolution update An encoder may choose to update selected macroblocks at a lower resolution than the normal spatial resolution of the frame. This may be useful, for example, to enable a CODEC to refresh moving parts of a frame at a low resolution using a small number of coded bits whilst keeping the static parts of the frame at the original higher resolution.

Annex R, Independent segment decoding This annex extends the concept of the independently decodeable slices (Annex K) or GOBs. Segments of the picture (where a segment is one slice or an integral number of GOBs) may be decoded completely independently of any other segment. In the slice structured mode (Annex K), motion vectors can point to areas of the reference picture that are outside the current slice; with independent segment decoding, motion vectors and other predictions can only reference areas within the current segment in the reference picture (Figure 5.6). A segment can be decoded (over a series of frames) independently of the rest of the frame.

Annex S, Alternative inter-VLC The encoder may use an alternative variable-length code table for transform coefficients in inter-coded blocks. The alternative VLCs (actually the same VLCs used for intra-coded blocks in Annex I) can provide better coding efficiency when there are a large number of high-valued quantised DCT coefficients (e.g. if the coded bit rate is high and/or there is a lot of variation in the video scene).

Annex T, Modified quantisation This mode introduces some changes to the way the quantiser and rescaling operations are carried out. Annex T allows the encoder to change the

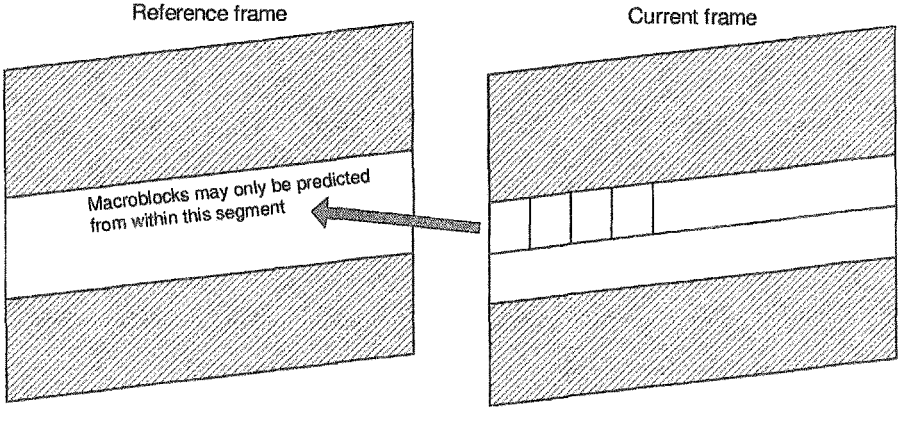


Figure 5.6 Independent segments

quantiser scale factor in a more flexible way during encoding, making it possible to control the encoder output bit rate more accurately.

Annex U, Enhanced reference picture selection Annex U modifies the reference picture selection mode of Annex N to provide improved error resilience and coding efficiency. There are a number of changes, including a mechanism to reduce the memory requirements for storing previously coded pictures and the ability to select a reference picture for motion compensation on a macroblock-by-macroblock basis. This means that the 'best' match for each macroblock may be selected from any of a number of stored previous pictures (also known as *long-term memory prediction*).

Annex V, Data partitioned slice Modified from Annex K, this mode improves the resilience of slice structured data to transmission errors. Within each slice, the macroblock data is rearranged so that all of the macroblock headers are transmitted first, followed by all of the motion vectors and finally by all of the transform coefficient data. An error occurring in header or motion vector data usually has a more serious effect on the decoded picture than an error in transform coefficient data: by rearranging the data in this way, an error occurring part-way through a slice should only affect the less-sensitive transform coefficient data.

Annex W, Additional supplemental enhancement information Two extra enhancement information items are defined (in addition to those defined in Annex L). The 'fixed-point IDCT' function indicates that an approximate inverse DCT (IDCT) may be used rather than the 'exact' definition of the IDCT given in the standard: this can be useful for low-complexity fixed-point implementations of the standard. The 'picture message' function allows the insertion of a user-definable message into the coded bit stream.

5.4.1 H.263 Profiles

It is very unlikely that all 19 optional modes will be required for any one application. Instead, certain combinations of modes may be useful for particular transmission scenarios. In common with MPEG-2 and MPEG-4, H.263 defines a set of recommended *profiles* (where a profile is a subset of the optional tools) and *levels* (where a level sets a maximum value on certain coding parameters such as frame resolution, frame rate and bit rate). Profiles and levels are defined in the final annex of H.263, Annex X. There are a total of nine profiles, as follows.

Profile 0, Baseline This is simply the baseline H.263 functionality, without any optional modes.

Profile 1, Coding efficiency (Version 2) This profile provides efficient coding using only tools available in Versions 1 and 2 of the standard (i.e. up to Annex T). The selected optional modes are Annex I (Advanced Intra-coding), Annex J (De-blocking Filter), Annex L (Supplemental Information: only the full picture freeze function is supported) and Annex T (Modified Quantisation). Annexes I, J and T provide improved coding efficiency compared with the baseline mode. Annex J incorporates the 'best' features of the first version of the standard, four motion vectors per macroblock and unrestricted motion vectors.

Profile 2, Coding efficiency (Version 1) Only tools available in Version 1 of the standard are used in this profile and in fact only Annex F (Advanced Prediction) is included. The other three annexes (D, E, G) from the original standard are not (with hindsight) considered to offer sufficient coding gains to warrant their use.

Profiles 3 and 4, Interactive and streaming wireless These profiles incorporate efficient coding tools (Annexes I, J and T) together with the slice structured mode (Annex K) and, in the case of Profile 4, the data partitioned slice mode (Annex V). These slice modes can support increased error resilience which is important for 'noisy' wireless transmission environments.

Profiles 5, 6, 7, Conversational These three profiles support low-delay, high-compression 'conversational' applications (such as video telephony). Profile 5 includes tools that provide efficient coding; Profile 6 adds the slice structured mode (Annex K) for Internet conferencing; Profile 7 adds support for interlaced camera sources (part of Annex W).

Profile 8, High latency For applications that can tolerate a higher latency (delay), such as streaming video, Profile 8 adds further efficient coding tools such as B-pictures (Annex O) and reference picture resampling (Annex P). B-pictures increase coding efficiency at the expense of a greater delay.

The remaining tools within the 19 annexes are not included in any profile, either because they are considered to be too complex for anything other than special-purpose applications, or because more efficient tools have superseded them.

5.5 H.26L[3]

The 19 optional modes of H.263 improved coding efficiency and transmission capabilities: however, development of H.263 standard is constrained by the requirement to continue to support the original 'baseline' syntax. The latest standardisation effort by the Video Coding Experts Group is to develop a new coding syntax that offers significant benefits over the older H.261 and H.263 standards. This new standard is currently described as 'H.26L', where the L stands for 'long term' and refers to the fact that this standard was planned as a long-term solution beyond the 'near-term' additions to H.263 (Versions 2 and 3).

The aim of H.26L is to provide a 'next generation' solution for video coding applications offering significantly improved coding efficiency whilst reducing the 'clutter' of the many optional modes in H.263. The new standard also aims to take account of the changing nature of video coding applications. Early applications of H.261 used dedicated CODEC hardware over the low-delay, low-error-rate ISDN. The recent trend is towards software-only or mixed software/hardware CODECs (where computational resources are limited, but greater flexibility is possible than with a dedicated hardware CODEC) and more challenging transmission scenarios (such as wireless links with high error rates and packet-based transmission over the Internet).

H.26L is currently at the test model development stage and may continue to evolve before standardisation. The main features can be summarised as follows.

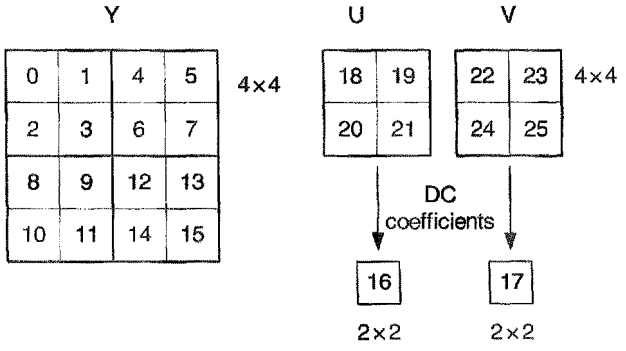


Figure 5.7 H.26L blocks in a macroblock

Processing units The basic unit is the macroblock, as with the previous standards. However, the subunit is now a 4×4 block (rather than an 8×8 block). A macroblock contains 26 blocks in total (Figure 5.7): 16 blocks for the luminance (each 4×4), four 4×4 blocks each for the chrominance components and two 2×2 'sub-blocks' which hold the DC coefficients of each of the eight chrominance blocks. It is more efficient to code these DC coefficients together because they are likely to be highly correlated.

Intra-prediction Before coding a 4×4 block within an intra-macroblock, each pixel in the block is predicted from previously coded pixels. This prediction reduces the amount of data coded in low-detail areas of the picture.

Prediction reference for inter-coding In a similar way to Annexes N and U of H.263, the reference frame for predicting the current inter-coded macroblock may be selected from a range of previously coded frames. This can improve coding efficiency and error resilience at the expense of increased complexity and storage.

Sub-pixel motion vectors H.26L supports motion vectors with $\frac{1}{4}$ pixel and (optionally) $\frac{1}{8}$ pixel accuracy; $\frac{1}{4}$-pixel vectors can give an appreciable improvement in coding efficiency over $\frac{1}{2}$-pixel vectors (e.g. H.263, MPEG-4) and $\frac{1}{8}$-pixel vectors can give a small further improvement (at the expense of increased complexity).

Motion vector options H.26L offers seven different options for allocating motion vectors within a macroblock, ranging from one vector per macroblock (Mode 1 in Figure 5.8) to an individual vector for each of the 16 luminance blocks (Mode 7 in Figure 5.8). This makes it possible to model the motion of irregular-shaped objects with reasonable accuracy. More motion vectors require extra bits to encode and transmit and so the encoder must balance the choice of motion vectors against coding efficiency.

De-blocking filter The de-blocking filter defined in Annex J of H.263 significantly improves motion compensation efficiency because it improves the 'smoothness' of the reference frame used for motion compensation. H.26L includes an integral de-blocking filter that operates across the edges of the 4×4 blocks within each macroblock.

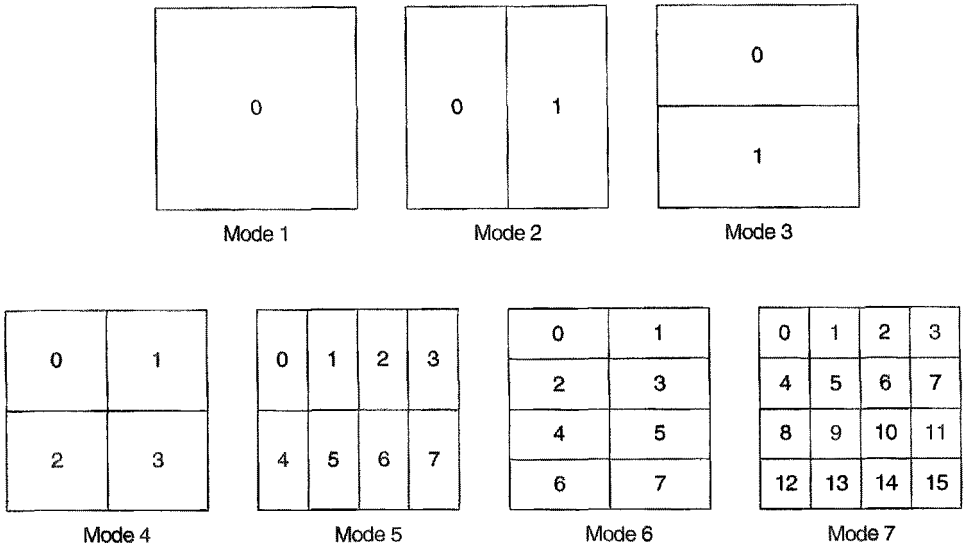


Figure 5.8 H.26L motion vector modes

4 × 4 Block transform After motion compensation, the residual data within each block is transformed using a 4 × 4 block transform. This is based on a 4 × 4 DCT but is an integer transform (rather than the floating-point 'true' DCT). An integer transform avoids problems caused by mismatches between different implementations of the DCT and is well suited to implementation in fixed-point arithmetic units (such as low-power embedded processors, Chapter 13).

Universal variable-length code The VLC tables in H.263 are replaced with a single 'universal' VLC. A transmitted code is created by building up a regular VLC from the 'universal' codeword. These codes have two advantages: they can be implemented efficiently in software without the need for storage of large tables and they are reversible, making it easier to recover from transmission errors (see Chapters 8 and 11 for further discussion of VLCs and error resilience).

Content-based adaptive binary arithmetic coding This alternative entropy encoder uses arithmetic coding (described in Chapter 8) to give higher compression efficiency than variable-length coding. In addition, the encoder can adapt to local image statistics, i.e. it can generate and use accurate probability statistics rather than using predefined probability tables.

B-pictures These are recognised to be a very useful coding tool, particularly for applications that are not very sensitive to transmission delays. H.26L supports B-pictures in a similar way to MPEG-1 and MPEG-2, i.e. there is no restriction on the number of B-pictures that may be transmitted between pairs of I- and/or P-pictures.

At the time of writing it remains to be seen whether H.26L will supersede the popular H.261 and H.263 standards. Early indications are that it offers a reasonably impressive performance gain over H.263 (see the next section): whether these gains are sufficient to merit a 'switch' to the new standard is not yet clear.

5.6 PERFORMANCE OF THE VIDEO CODING STANDARDS

Each of the image and video coding standards described in Chapters 4 and 5 was designed for a different purpose and includes different features. This makes it difficult to compare them directly. Figure 5.9 compares the PSNR performance of each of the video coding standards for one particular test video sequence, 'Foreman', encoded at QCIF resolution and a frame rate of 10 frames per second. The results shown in the figure should be interpreted with caution, since different performance will be measured depending on the video sequence, frame rate and so on. However, the trend in performance is clear. MJPEG performs poorly (i.e. it requires a relatively high data rate to support a given picture 'quality') because it does not use any inter-frame compression. H.261 achieves a substantial gain over MJPEG, due to the use of integer-pixel motion compensation. MPEG-2 (with half-pixel motion compensation) is next, followed by H.263/MPEG-4 (which achieve a further gain by using four motion vectors per macroblock). The emerging H.26L test model achieves the best performance of all. (Note that MPEG-1 achieves the same performance as MPEG-2 in this test because the video sequence is not interlaced.)

This comparison is not the complete picture because it does not take into account the special features of particular standards (for example, the content-based tools of MPEG-4 or the interlaced video tools of MPEG-2). Table 5.1 compares the standards in terms of coding performance and features. At the present time, MPEG-2, H.263 and MPEG-4 are each viable

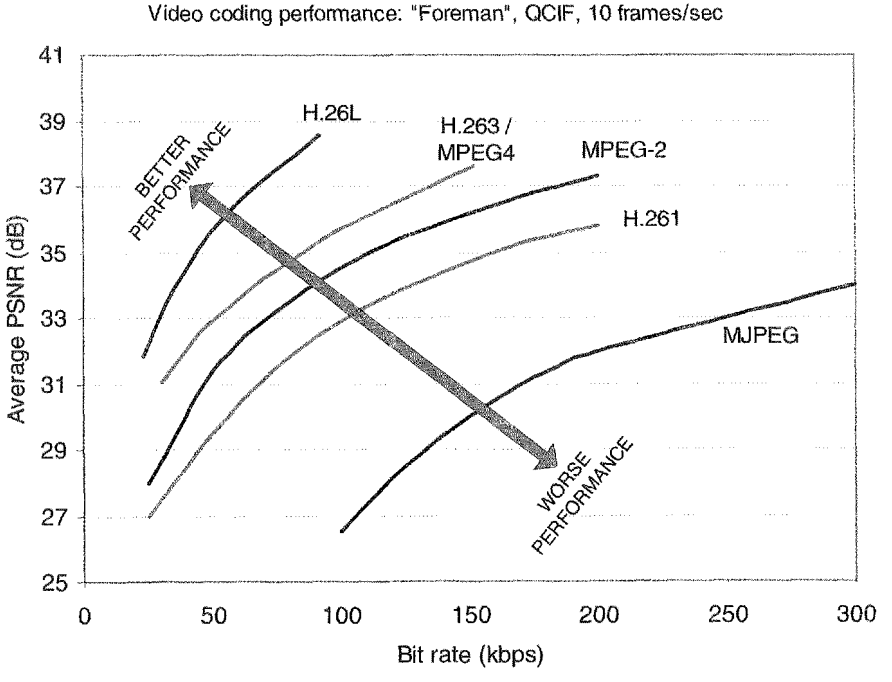


Figure 5.9 Coding performance comparison

Table 5.1 Comparison of the video coding standards

Standard	Target application	Coding performance	Features
MJPEG	Image coding	1 (worst)	Scalable and lossless coding modes
H.261	Video conferencing	2	Integer-pixel motion compensation
MPEG-1	Video-CD	3 (equal)	I, P, B-pictures, half-pixel compensation
MPEG-2	Digital TV	3 (equal)	As above; field coding, scalable coding
H.263	Video conferencing	4 (equal)	Optimised for low bit rates; many optional modes
MPEG-4	Multimedia coding	4 (equal)	Many options including content-based tools
H.26L	Video conferencing	5 (best)	Full feature set not yet defined

alternatives for designers of video communication systems. MPEG-2 is a relatively mature technology for the mass-market digital television applications; H.263 offers good coding performance and options to support a range of transmission scenarios; MPEG-4 provides a large toolkit with the potential for new and innovative content-based applications. The emerging H.26L standard promises to outperform the H.263 and MPEG-4 standards in terms of video compression efficiency[4] but is not yet finalised.

5.7 SUMMARY

The ITU-T Video Coding Experts Group developed the H.261 standard for video conferencing applications which offered reasonable compression performance with relatively low complexity. This was superseded by the popular H.263 standard, offering better performance through features such as half-pixel motion compensation and improved variable-length coding. Two further versions of H.263 have been released, each offering additional optional coding modes to support better compression efficiency and greater flexibility. The latest version (Version 3) includes 19 optional modes, but is constrained by the requirement to support the original, 'baseline' H.263 CODEC. The H.26L standard, under development at the time of writing, incorporates a number of new coding tools such as a 4×4 block transform and flexible motion vector options and promises to outperform earlier standards.

Comparing the performance of the various coding standards is difficult because a direct 'rate-distortion' comparison does not take into account other factors such as features, flexibility and market penetration. It seems clear that the H.263, MPEG-2 and MPEG-4 standards each have their advantages for designers of video communication systems. Each of these standards makes use of common coding technologies: motion estimation and compensation, block transformation and entropy coding. In the next section of this book we will examine these core technologies in detail.

REFERENCES

1. ITU-T Recommendation H.261, 'Video CODEC for audiovisual services at px64 kbit/s', 1993.
2. ITU-T Recommendation H.263, 'Video coding for low bit rate communication', Version 2, 1998.
3. ITU-T Q6/SG16 VCEG-L45, 'H.26L Test Model Long Term Number 6 (TML-6) draft 0', March 2001.
4. ITU-T Q6/SG16 VCEG-M08, 'Objective coding performance of [H.26L] TML 5.9 and H.263+', March 2001.