Digital Multimedia Standards Series

DIGITAL VIDEO: AN INTRODUCTION TO MPEG-2

Barry G. Haskell,
Atul Puri, and
Arun N. Netravali

Cover design: Curtis Tow Graphics

Printed in the United States of America

Chapman & Hall
115 Fifth Avenue
New York, NY 10003

Chapman & Hall
2-6 Boundary Row
London SE1 8HN
England

Thomas Nelson Australia
102 Dodds Street
South Melbourne, 3205
Victoria, Australia

Chapman & Hall GmbH
Postfach 100 263
D-69442 Weinheim
Germany

International Thomson Editores
Campos Eliseos 385, Piso 7
Col. Polanco
11560 Mexico D.F
Mexico

International Thomson Publishing–Japan
Hirakawacho-cho Kyowa Building, 3F
1-2-1 Hirakawacho-cho
Chiyoda-ku, 102 Tokyo
Japan

International Thomson Publishing Asia
221 Henderson Road #05-10
Henderson Building
Singapore 0315

3 4 5 6 7 8 9 10 XXX 01 00 99 98 97

Library of Congress Cataloging-in-Publication Data

Haskell, Barry G.
Digital video : an introduction to MPEG-2 / Barry G. Haskell, Atul Puri, and Arun N. Netravali.
p. cm.
Includes bibliographical references and index.
ISBN 0-412-08411-2
1. Digital video. 2. Video compression -- Standards. 3. Coding theory. I. Puri, Atul. II. Netravali, Arun N. III. Title.
TK6680.5.H37 1996
621.388'33--dc20 96-14018
CIP

British Library Cataloguing in Publication Data available

To order this or any other Chapman & Hall book, please contact **International Thomson Publishing, 7625 Empire Drive, Florence, KY 41042.** Phone: (606) 525-6600 or 1-800-842-3636. Fax: (606) 525-7778. e-mail: order@chaphall.com.

For a complete listing of Chapman & Hall titles, send your request to **Chapman & Hall, Dept. BC, 115 Fifth Avenue, New York, NY 10003.**

1

Introduction to Digital Multimedia, Compression, and MPEG-2

1.1 THE DIGITAL REVOLUTION

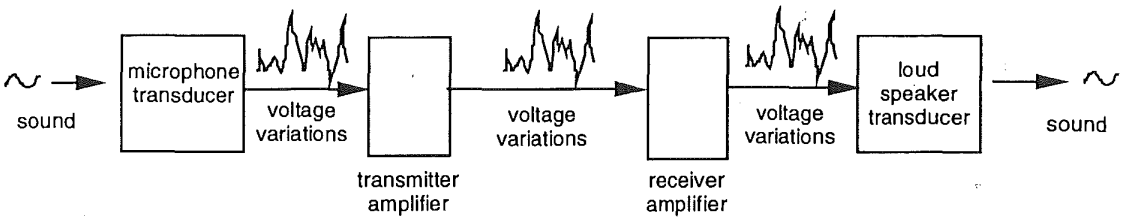
It is practically a cliché these days to claim that all electronic communications is engaged in a digital revolution. Some communications media such as the telegraph and telegrams were always digital. However, almost all other electronic communications started and flourished as analog forms.

In an analog communication system, a transmitter sends voltage variations, which a receiver then uses to control transducers such as loudspeakers, fax printers, and TV cathode ray tubes (CRTs), as shown in Figure 1.1. With digital, on the other hand, the transmitter first converts the controlling voltage into a sequence of 0s and 1s (called *bits*), which are then transmitted. The receiver then reconverts or *decodes* the bits back into a replica of the original voltage variations.

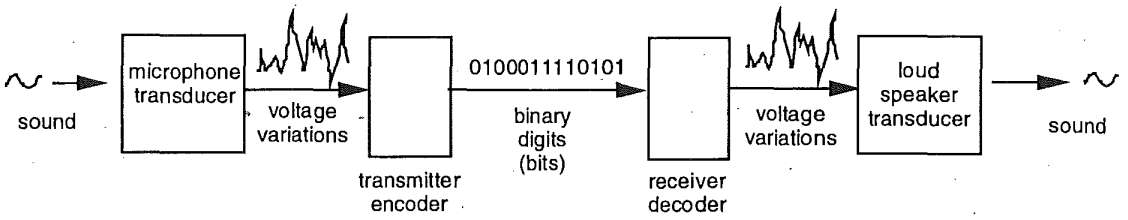
The main advantage of digital representation of information is the robustness of the bitstream. It can be stored and recovered, transmitted and received, processed and manipulated, all virtually without error. The only requirement is that a zero bit be distinguishable from a one bit, a task that is quite easy in all modern signal handling systems. Over the years, long and medium distance telephone transmissions have all become digital. Fax transmission changed to digital in the 1970s and 80s. Digitized entertainment audio on compact disks (CDs) has completely displaced vinyl records and almost replaced cassette tape. Digital video satellite is available in several countries, and digital video disks have just arrived in the market. Many other video delivery media are also about to become digital, including over-the-air, coaxial cable, video tape and even ordinary telephone wires. The pace of this conversion is impressive, even by modern standards of technological innovation.

Almost all sensory signals are analog at the point of origination or at the point of perception. Audio and video signals are no exception. As an example, television, which was invented and standardized over fifty years ago, has remained mostly analog from camera to display, and a plethora of products and services have been built around these analog standards, even though dramatic changes have occurred in technology. However, inexpensive integrated circuits, high-speed communication networks, rapid access dense storage media, and computing architectures that can easily handle video-rate data are now rapidly making the analog standard obsolete.

Digital audio and video signals integrated with computers, telecommunication networks, and consumer products are poised to fuel the information revolution. At the heart of this revolution is the digital compression of audio and video signals.



Analog Transmission System



Digital Transmission System

Fig. 1.1 Analog communication systems send voltage variations that track in analogous fashion the waveforms produced by transducers such as microphones, fax scanners, and so forth. Digital systems first convert or *encode* the waveforms into a string of 0s and 1s called *bits*. The bits are then sent to a receiver, which *decodes* the bits back into an approximation of the original waveform.

This chapter summarizes the benefits of digitization as well as some of the requirements for the compression of the multimedia signals.

1.1.1 Being Digital

Even though most sensory signals are analog, the first step in processing, storage, or communication is usually to digitize the signals into a string of bits. Analog-to-digital converters that digitize such signals with required accuracy and speed have become inexpensive over the years. The cost or quality of digitization therefore is no longer an issue. However, simple digitization usually results in a *bandwidth expansion,* in the sense that transmitting or storage of these bits often takes up more bandwidth or storage space than the original analog signal. In spite of this, digitization is becoming universal because of the relative ease of handling the digital signal versus the analog. In particular, signal processing for enhancement, removal of artifacts, transformation, compression, and so forth is much easier to do in the digital domain using a growing family of specialized integrated circuits. One example of this is the conversion from one video standard to another (e.g., NTSC to PAL*). Sophisticated adaptive algorithms required for good picture quality in standards conversion can be implemented only in the digital domain. Another example is the editing of digitized signals. Edits that require transformation (e.g., rotation, dilation of pictures or time-warp for audio) are significantly more difficult in the analog domain.

The density of today's digital storage media continues to increase, making storage of digital multimedia signals practical. With digital storage,

*NTSC and PAL are analog composite color TV standards. Japan and North America use NTSC. Most of Europe, Australia etc. use PAL.

the quality of the retrieved signal does not degrade in an unpredictable manner with multiple reads as it often does with analog storage. Also, with today's database and user interface technology, a rich set of interactions is possible only with stored digital signals. For example, with medical images the quality can be improved by noise reduction and contrast enhancement algorithms. Through the use of pseudo color, the visibility of minute variations and patterns can be greatly increased.

Mapping the stored signal to displays with different resolutions in space (number of lines per screen and number of samples per line) and time (frame rates) can be done easily in the digital domain. A familiar example of this is the conversion of film,* which is almost always at a different resolution and frame rate than the television signal.

Digital signals are also consistent with the evolving telecommunications infrastructure. Digital transmission allows much better control of the quality of the transmitted signal. In broadcast television, for example, if the signal were digital, the reproduced picture in the home could be identical to the picture in the studio, unlike the present situation where the studio pictures look far better than pictures at home. Finally, analog systems dictate that the entire television system from camera to display operate at a common clock with a standardized display. In the digital domain considerable flexibility exists by which the transmitter and the receiver can negotiate the parameters for scanning, resolution, and so forth, and thus create the best picture consistent with the capability of each sensor and display.

1.2 THE NEED FOR COMPRESSION

With the advent of fax came the desire for faster transmission of documents. With a limited bitrate available on the PSTN,† the only means available to increase transmission speed was to reduce the average number of bits per page, that is, *compress* the digital data. With the advent of video conferencing, the need for digital compression[1] was even more crucial. For video, simple sampling and binary coding of the camera voltage variations produces millions of bits per second, so much so that without digital compression, any video conferencing service would be prohibitively expensive.

For entertainment TV, the shortage of over-the-air bandwidth has led to coaxial cable (CATV‡) connections directly to individual homes. But even today, there is much more programming available via satellite than can be carried on most analog cable systems. Thus, compression of entertainment TV would allow for more programming to be carried over-the-air and through CATV transmission.

For storage of digital video on small CD sized disks, compression is absolutely necessary. There is currently no other way of obtaining the quality demanded by the entertainment industry while at the same time storing feature length films, which may last up to two hours or longer.

Future low bitrate applications will also require compression before service becomes viable. For example, wireless cellular video telephony must operate at bitrates of a few dozen kilobits per second, which can only be achieved through large compression of the data. Likewise, video retrieval on the InterNet or World Wide Web is only feasible with compressed video data.

1.2.1 What Is Compression?

Most sensory signals contain a substantial amount of redundant or superfluous information. For example, a television camera that captures 30 frames per second from a stationary scene produces very similar frames, one after the other. Compression attempts to remove the superfluous information so that a single frame can be represented by a smaller amount of finite data, or in the case of audio or time varying images, by a lower data rate.

Digitized audio and video signals contain a significant amount of *statistical redundancy.* That is,

*Most commercial movie film operates at 24 frames per second, whereas U.S. Broadcast TV operates at approximately 30 frames per second.

†Public Switched Telephone Network.

‡Community Antenna Television.

samples are similar to each other so that one sample can be predicted fairly accurately from another. By removing the predictable or similarity component from a stream of samples, the data rate can be reduced. Such statistical redundancy can be removed without destroying any information whatsoever. That is, the original uncompressed data can be recovered exactly by various inverse operations. Unfortunately, the techniques for accomplishing this depend on the probabilistic characterization of the signal. Although many excellent probabilistic models of audio and video signals have been proposed, serious limitations continue to exist because of the nonstationarity of the signal statistics. In addition, statistics may vary widely from application to application. A football game captured by a video camera shows very different correlations of samples compared to the head and shoulders view of people engaged in video telephony.

The second type of superfluous data is the information that a human audio–visual system can neither hear nor see. We call this *perceptual redundancy.* If the primary receiver of the multimedia signal is a human (rather than a machine as in the case of some pattern recognition applications), then transmission or storage of the information that humans cannot perceive is wasteful. Unlike statistical redundancy, the removal of information based on the limitations of the human perception is irreversible. The original data cannot be recovered following such a removal. Unfortunately, human perception is very complex, varies from person to person, and depends on the context and the application. Therefore, the art and science of compression still has many frontiers to conquer even though substantial progress has been made in the last two decades.

1.2.2 Advantages of Compression

The biggest advantage of compression is in data rate reduction. Data rate reduction reduces transmission costs, and where a fixed transmission capacity is available, results in a better quality of multimedia presentation. As an example, a single 6-MHz analog* cable TV channel can carry between four and ten digitized, compressed, audio–visual programs, thereby increasing the overall capacity (in terms of the number of programs carried) of an existing cable television plant. Alternatively, a single 6-MHz broadcast television channel can carry a digitized, compressed High-Definition Television (HDTV) signal to give a significantly better audio and picture quality without additional bandwidth. We shall discuss in detail the applications of compressed digital TV in Chapters 12 through 15.

Data rate reduction also has a significant impact on reducing the storage requirements for a multimedia database. A CD-ROM will soon be able to carry a full length feature movie compressed to about 4 Mbits/s. Thus, compression not only reduces the storage requirement, but also makes stored multimedia programs portable in inexpensive packages. In addition, the reduction of data rate allows processing of video-rate data without choking various resources (e.g., the main bus) of either a personal computer or a workstation.

Another advantage of digital representation/compression is for packet communication. Much of the data communication in the computer world is by self-addressed packets. Packetization of digitized audio–video and the reduction of packet rate due to compression are important in sharing a transmission channel with other signals as well as maintaining consistency with telecom/computing infrastructure. The desire to share transmission and switching has created a new evolving standard, called Asynchronous Transfer Mode (ATM), which uses packets of small size, called *cells.* Packetization delay, which could otherwise hinder interactive multimedia, becomes less of an issue when packets are small. High compression and large packets make interactive communication difficult, particularly for voice.

1.3 CONSIDERATIONS FOR COMPRESSION

The algorithms used in a compression system depend on the available bandwidth or storage

*NTSC uses 6-MHz analog bandwidth. PAL uses more.

capacity, the features required by the application, and the affordability of the hardware required for implementation of the compression algorithm (encoder as well as decoder). This section describes some of the issues in designing the compression system.

1.3.1 Quality

The quality of presentation that can be derived by decoding the compressed multimedia signal is the most important consideration in the choice of the compression algorithm. The goal is to provide near-transparent coding for the class of multimedia signals that are typically used in a particular service.

The two most important aspects of video quality are *spatial* resolution and *temporal* resolution. Spatial resolution describes the clarity or lack of blurring in the displayed image, while temporal resolution describes the smoothness of motion.

Motion video, like film, consists of a certain number of frames per second to adequately represent motion in the scene. The first step in digitizing video is to partition each frame into a large number of *picture elements,* or *pels** for short. The larger the number of pels, the higher the spatial resolution. Similarly, the more frames per second, the higher the temporal resolution.

Pels are usually arranged in rows and columns, as shown in Fig. 1.2. The next step is to measure the brightness of the red, green, and blue color components within each pel. These three color brightnesses are then represented by three binary numbers.

1.3.2 Uncompressed versus Compressed Bitrates

For entertainment television in North America and Japan, the NTSC† color video image has 29.97 frames per second; approximately 480 visi-

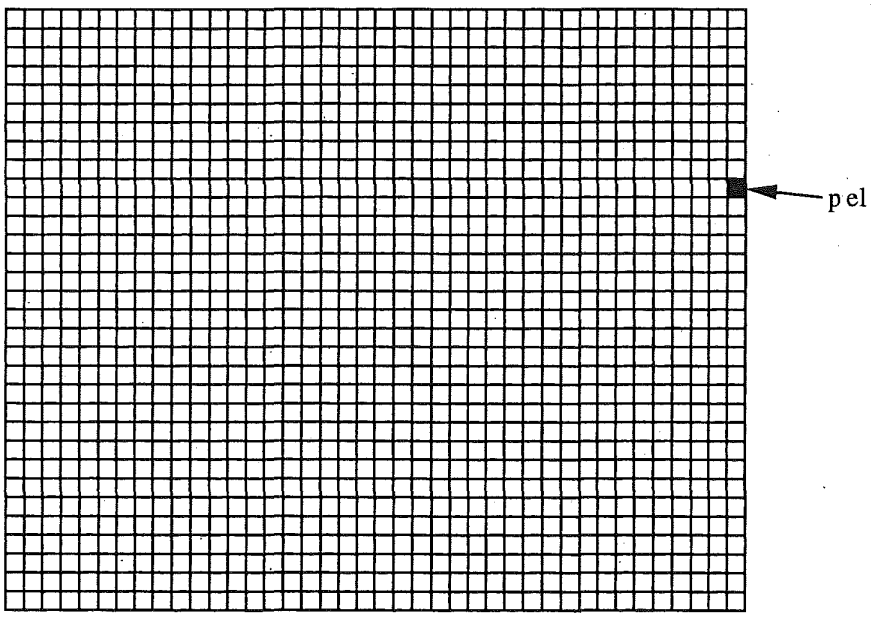


Fig. 1.2 Frames to be digitized are first partitioned into a large number of *picture elements,* or *pels* for short, which are typically arranged in rows and columns. A row of pels is called a *scan line.*

*Some authors prefer the slightly longer term *pixels.*

†National Television Systems Committee. It standardized composite color TV in 1953.

Table 1.1 Raw and compressed bitrates for film, NTSC, PAL, HDTV, videophone, stereo audio, and five-channel audio. Compressed bitrates are typically lower for slow moving drama than for fast live-action sports.

	Video Resolution (pels × lines × frames/s)	Uncompressed Bitrate (RGB)	Compressed Bitrate
Film (USA and Japan)	(480 × 480 × 24Hz)	133 Mbits/s	3 to 6 Mbits/s
NTSC video	(480 × 480 × 29.97Hz)	168 Mbits/s	4 to 8 Mbits/s
PAL video	(576 × 576 × 25Hz)	199 Mbits/s	4 to 9 Mbits/s
HDTV video	(1920 × 1080 × 30Hz)	1493 Mbits/s	18 to 30 Mbits/s
HDTV video	(1280 × 720 × 60Hz)	1327 Mbits/s	18 to 30 Mbits/s
ISDN videophone (CIF)	(352 × 288 × 29.97Hz)	73 Mbits/s	64 to 1920 kbits/s
PSTN videophone (QCIF)	(176 × 144 × 29.97Hz)	18 Mbits/s	10 to 30 kbits/s
Two-channel stereo audio		1.4 Mbits/s	128 to 384 kbits/s
Five-channel stereo audio		3.5 Mbits/s	384 to 968 kbits/s

ble scan lines per frame; and requires approximately 480 pels per scan line in the red, green, and blue color components. If each color component is coded using 8 bits (24 bits/pel total), the bitrate produced is ≈168 Megabits per second (Mbits/s). Table 1.1 shows the raw uncompressed bitrates for film, several video formats including PAL,* HDTV,† and videophone, as well as a few audio formats.

We see from Table 1.1 that uncompressed bitrates are all very high and are not economical in many applications. However, using the MPEG compression algorithms these bitrates can be reduced considerably. We see that MPEG is capable of compressing NTSC or PAL video into 3 to 6 Mbits/s with a quality comparable to analog CATV and far superior to VHS videotape.

1.3.3 Bitrates Available on Various Media

A number of communication channels are available for multimedia transmission and storage. PSTN modems can operate up to about 30 kilobits/second (kbits/s), which is rather low for generic video, but is acceptable for personal videophone. ISDN‡ ranges from 64 kbits/s to 2 Mbits/s, which is often acceptable for moderate quality video.

Table 1.2 Various media and the bitrates they support.

Medium	Bitrate
PSTN modems	Up to 30 kbits/s
ISDN	64 to 1920 kbits/s
LAN	10 to 100 Mbits/s
ATM	135 Mbits/s or more
CD-ROM (normal speed)	1.4 Mbits/s
Digital video disk	9 to 10 Mbits/s
Over-the-air video	18 to 20 Mbits/s
CATV	20 to 40 Mbits/s

LANs§ range from 10 Mbits/s to 100 Mbits/s or more, and ATM** can be many hundreds of Mbits/s. First generation CD-ROMs†† can handle 1.4 Mbits/s at normal speed and often have double or triple speed options. Second generation digital video disks operate at 9 to 10 Mbits/s depending on format. Over-the-air or CATV channels can carry 20 Mbits/s to 40 Mbits/s, depending on distance and interference from other transmitters. These options are summarized in Table 1.2.

1.4 COMPRESSION TECHNOLOGY

In picture coding, for example, compression techniques have been classified according to the groups

*Phase Alternate Line, used in most of Europe and elsewhere.
†High-Definition Television. Several formats are being used.
‡Integrated Services Digital Network. User bitrates are multiples of 64 kbits/s.
§Local Area Networks such as Ethernet, FDDI, etc.
**Asynchronous Transfer Mode. ATM networks are packetized.
††Compact Disk–Read Only Memory.

Table 1.3 Picture coding primitive elements. Pulse code modulation (PCM) is simple binary coding of pels. Differential pulse code modulation (DPCM) sends differences of pels.

Coded Element	Encoding	Example
Picture element	PCM, DPCM	Wirephoto
Block of elements	Spatial transform	JPEG, subband
Translated blocks	Motion compensation	MPEG-1, -2
Regions	Object-based coders	MPEG-4 (1998)
Named content	3-D scene elements	(Research only)

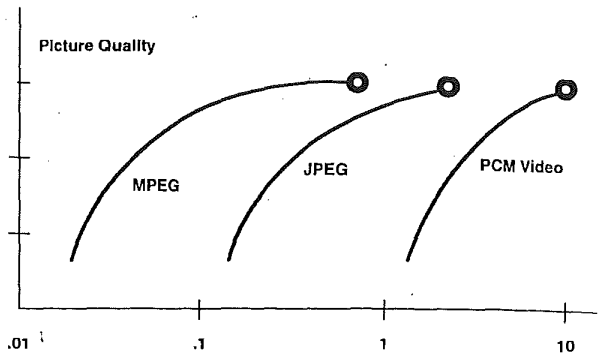


Fig. 1.3 Approximate measure of picture quality versus bitrate in bits/pel of several video compression algorithms.

of pels on which they operate. Table 1.3 shows a simplified view of this. Encoding using a few pels is considerably simpler than encoding that requires a complex spatial–temporal processing of groups of pels. However, performance is lower.

Over the years, to gain compression efficiency (that is, picture quality for a given bitrate), the compression algorithms have become more and more complex. Figure 1.3 shows a rough qualitative comparison of the relative picture quality at various compression ratios for three different coding methods.

It is important to note that the less complex the algorithm, the more rapid the degradation of picture quality with only small changes in compression ratio. In general, entertainment applications using broadcast, cable, or satellite TV have tended to require a much higher quality of compressed signal than video telephony, video groupware, and video games. Since most entertainment signals have a significantly larger amount of motion than video telephone signals, a smaller compression ratio results for the high quality that is required. In any case, both the compression algorithm and the compression ratio need to be tailored for the picture quality that is required for the application at hand.

1.4.1 Robustness

As the redundancy from the multimedia signal is removed by compression, each compressed bit becomes more important in the sense that it affects a larger number of samples of the audio or picture signal. Therefore, an error either in transmission or storage of the compressed bit can have deleterious effects for either a large region of the picture or over an extended period of time.

For noisy digital transmission channels, video compression algorithms that sacrifice efficiency to allow for graceful degradation of the images in the presence of channel errors are better candidates. Some of these are created by merging source and channel coding to optimize the end-to-end service quality. A good example of this is portable multimedia over a low-bandwidth wireless channel. Here, the requirements on compression efficiency are severe owing to the lack of available bandwidth. Yet a compression algorithm that is overly sensitive to channel errors would be an improper choice. Of course, error correction is usually added to an encoded signal along with a variety of error concealment techniques, which are usually successful in reducing the effects of random isolated errors. Thus, the proper choice of the compression algorithm depends on the transmission error environment in which the application resides.

1.4.2 Interactivity

Both consumer entertainment and business multimedia applications are characterized by picture scanning and browsing. In the home, viewers switch to the channels of their choice. In the business environment, people get to the information of their choice by random access using, for example, on-screen menus. In the television of the future, a much richer interaction based on content rather than channel switching or accessing a different picture will be required.

Many multimedia offerings and locally produced video programs often depend on the con-

catenation of video streams from a variety of sources, sometimes in real time. Commercials are routinely inserted into nationwide broadcasts by network affiliates and cable headends. Thus, the compression algorithm must support a continuous and seamless assembly of these streams for distribution and rapid switching of images at the point of final decoding. It is also desirable that these simple edits as well as richer interactions occur on compressed data rather than reconstructed sequences.

In general, a higher degree of interactivity requires a compression algorithm that operates on a smaller group of pels. MPEG, which operates on spatio-temporal groups of pels, is more difficult to interact with than JPEG,[2] which operates only on spatial groups of pels. As an example, it is much easier to fast forward a compressed JPEG bitstream than a compressed MPEG bitstream. In a cable/broadcast environment or in an application requiring browsing through a compressed multimedia database, a viewer may change from program to program with no opportunity for the encoder to adapt itself. It is important that the buildup of resolution following a program change take place quite rapidly so that the viewer can make a decision to either stay on the program or change to another depending on the content the viewer wishes to watch.

1.4.3 Compression and Packetization Delay

Advances in compression have come predominantly through better analysis of the signal in question. For picture coding this is exemplified in Table 1.3. As models have progressed from element-based to picture blocks to interframe regions, efficiency has grown rapidly. Correspondingly, the complexity of the analysis phase of encoding has also grown, resulting in the encoding delay becoming an important parameter. A compression algorithm that looks at a large number of samples and performs very complex operations usually has a larger encoding delay.

For many applications, such encoding delay at the source is tolerable, but for some it is not. Broadcast television, even in real time, can often admit a delay in the order of seconds. However, teleconferencing or multimedia groupware can tolerate a far smaller delay. In addition to the encoding delay, modern data communications that packetize the information also introduce delay. The more efficient the compression algorithm, the larger is the delay that is introduced by packetization, since the same size packet carries information about many more samples of the multimedia signal. Thus, the compression algorithm must also take into consideration the delay that is tolerable for the application.

1.4.4 Symmetry

A cable, satellite, or broadcast environment has only a few transmitters that compress, but a large number of receivers that have to decompress. Similarly, multimedia databases that store compressed information usually compress it only once. However, the retrieval of this information may happen thousands of times by different viewers. Therefore, the overall economics of many multimedia applications is dictated to a large extent by the cost of decompression. The choice of the compression algorithm ought to make the decompression extremely simple by transferring much of the cost to the transmitter, thereby creating an *asymmetrical* algorithm. The analysis phase of a compression algorithm, which routinely includes motion analysis (done only at the encoder), naturally makes the encoder more expensive. In a number of situations, the cost of the encoder is also important (e.g., camcorder, videotelephone). Therefore, a modular design of the encoder that is able to trade off performance with complexity, but that creates data decodable by a simple decompressor, may be the appropriate solution.

1.4.5 Multiple Encoding

In a number of instances, the original multimedia signal may have to be compressed in stages or may have to be compressed and decompressed several times. In most television studios, for example, it is necessary to store the compressed data and then decompress it for editing as required. Such an edited signal is then compressed and stored again. Any multiple coding–decoding cycle

of the signal is bound to reduce the quality of the multimedia signal, since artifacts are introduced every time the signal is coded. If the application requires such multiple codings, then a higher quality compression is required, at least in the several initial stages.

1.4.6 Scalability

A compressed multimedia signal can be thought of as an alternative representation of the original uncompressed signal. From this alternative representation, it is desirable to create presentations at different resolutions (in space, time, amplitude, etc.) consistent with the limitations of the equipment used in a particular application. For example, if a high-definition television signal compressed to 24 Mbits/s can be simply processed to produce a lower resolution and lower bitrate signal (e.g., NTSC at 6 Mbits/s), the compression is generally considered to be *scalable.* Of course, the scalability can be achieved in a brute force manner by decompressing, reducing the resolution, and compressing again. However, this sequence of operations introduces delay and complexity, and results in a loss of quality. A common compressed representation from which a variety of low-resolution or higher resolution presentations can be easily derived is desirable. Such scalability of the compressed signal puts a constraint on the compression efficiency in the sense that algorithms with the highest compression efficiency usually are not very scalable.

1.5 VIDEOPHONE AND COMPACT DISK STANDARDS—H.320 AND MPEG-1

Digital compression standards for video conferencing were developed in the 1980s by the CCITT, which is now known as the ITU-T. Specifically, the ISDN video conferencing standards are known collectively as H.320, or sometimes P*64 to indicate that it operates at multiples of 64 kbits/s. The video coding portion of the standard is called H.261 and codes pictures at a *Common Intermediate Format* (CIF) of 352 pels by 288 lines. A lower resolution of 176 pels by 144 lines, called QCIF, is available for interoperating with PSTN videophones.

In the late 1980s, a need arose to place motion video and its associated audio onto first generation CD-ROMs at 1.4 Mbits/s. For this purpose, in the late 1980s and early 1990s, the ISO MPEG committee developed digital compression standards[4] for both video and two-channel stereo audio. The standard[3,4,8] is known colloquially as MPEG-1 and officially as ISO 11172. The bitrate of 1.4 Mbits/s available on first generation CD-ROMs is not high enough to allow for full-resolution TV. Thus, MPEG-1 was optimized for the reduced CIF resolution of H.320 video conferencing.

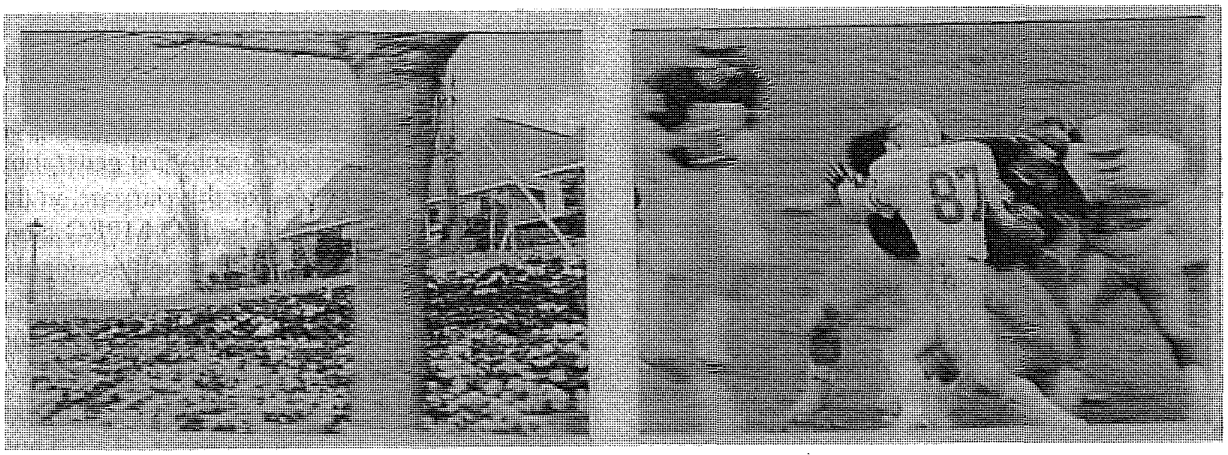
1.6 THE DIGITAL ENTERTAINMENT TV STANDARD—MPEG-2

Following MPEG-1, the need arose to compress entertainment TV for such transmission media as satellite, cassette tape, over-the-air, and CATV. Thus, to have available digital compression methods[4–6,8,10] for full-resolution Standard Definition TV (SDTV) pictures such as shown in Fig. 1.4a or High Definition TV (HDTV) pictures such as shown in Fig. 1.4b, ISO developed a second standard known colloquially as MPEG-2 and officially as ISO 13818. Since the resolution of entertainment TV is approximately four times that of videophone, the bitrate chosen for optimizing MPEG-2 was 4 Mbits/s.

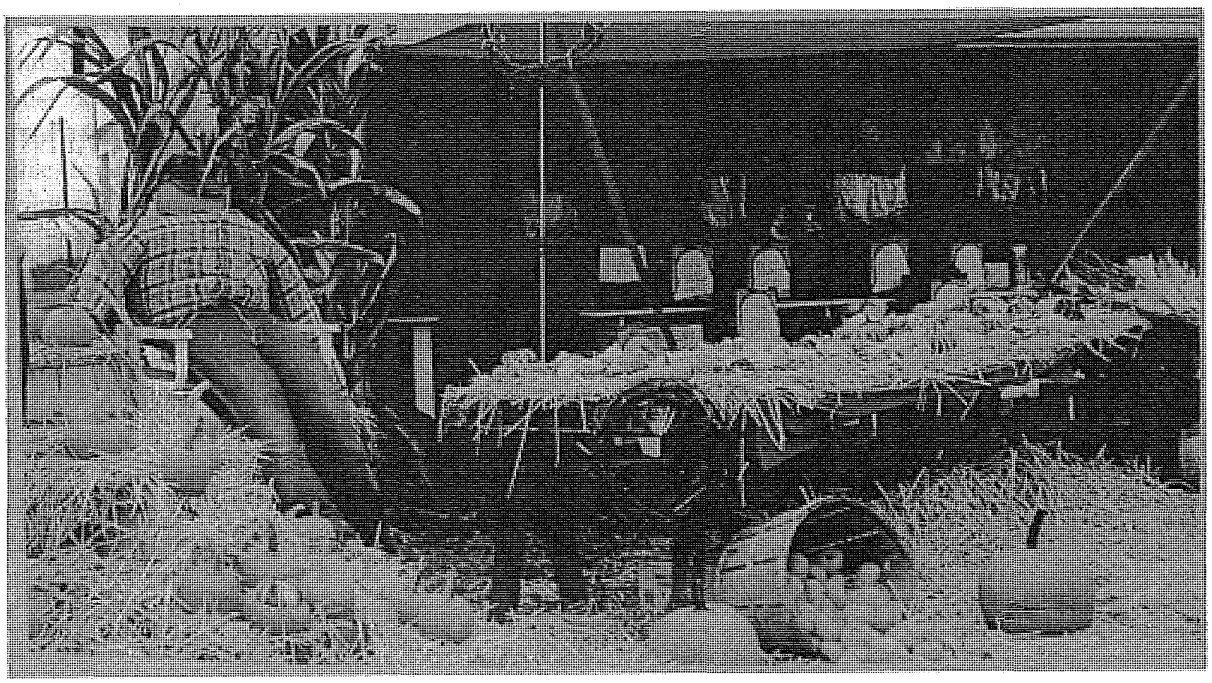
1.6.1 Other Functionalities of MPEG-2

In addition to simply compressing audio and video with high quality and low bitrate, a number of other capabilities[10] are needed for full multimedia service:

Random Access: For stored multimedia or Interactive Television (ITV) applications such as shown in Fig. 1.5 we often need quick access to the interior of the audiovisual data without having to start decoding from the very beginning or waiting a long time for the decoder to start up.



(a)



(b)

Fig. 1.4 a) Example of a full resolution Standard Definition TV (SDTV) picture. b) Example of a High Definition TV (HDTV) picture. Courtesy of Richard Kollarits.

Trick Modes: Users of stored multimedia want all of the features they now have on VCRs. These include fast-play forward and reverse, slow-play forward and reverse, and freeze-frame.

Multicast to Many Terminal Types: Some applications may need to simultaneously broadcast to many different terminal types over a variety of communication channels. Preparing and storing a different bitstream for each of them is impractical. Thus, a single bit stream is desired that is *scalable* to the various requirements of the channels and terminals.

Multiple Audio and Video: Multiple audios are needed, for example, if many languages are to be provided simultaneously. Multiple videos may also be required if closeups and wide area views are to be sent simultaneously in the same program.

Compatible Stereoscopic 3D: Compatible 3D for pictures such as shown in Fig. 1.6, is needed so that viewers with 3D displays can see 3D program-

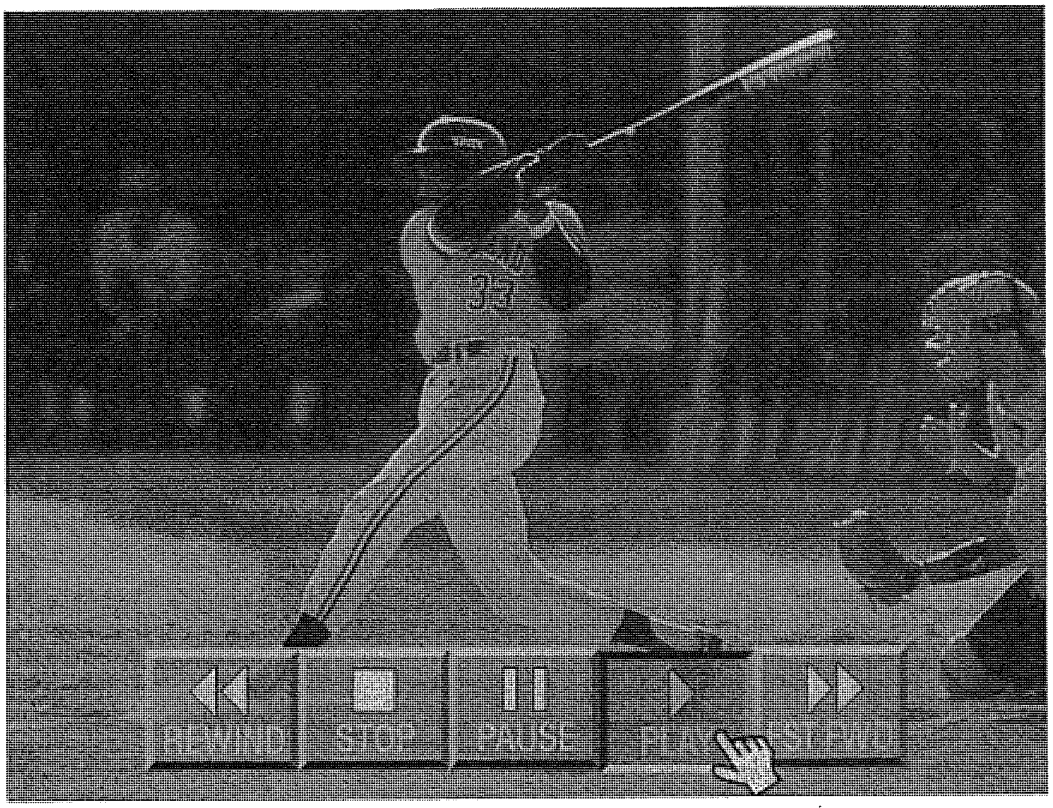


Fig. 1.5 Example of an Interactive Television (ITV) picture.

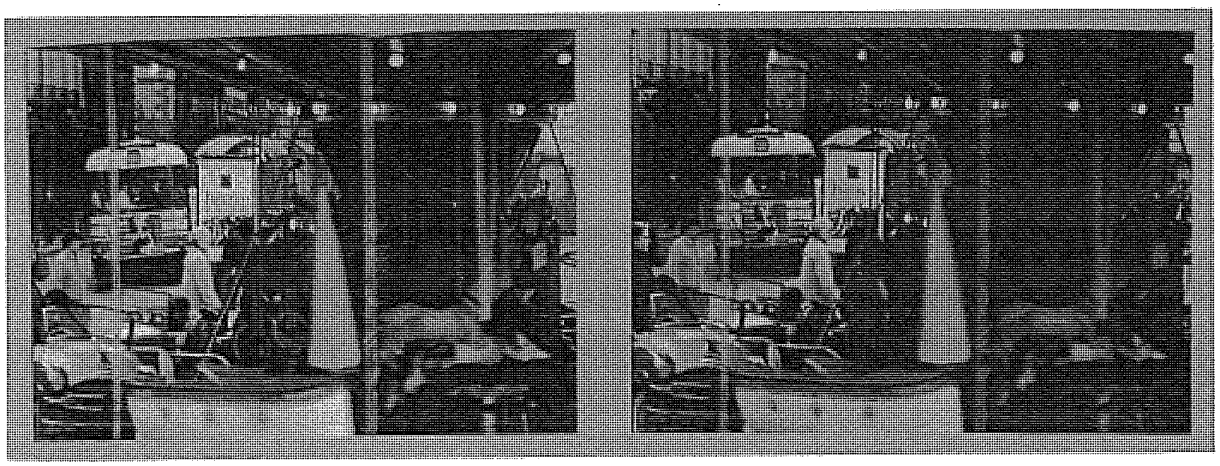


Fig. 1.6 Example of left and right eye views of a stereoscopic 3D picture. Courtesy of C.C.E.T.T., Rennes, France.

ming,[9] while those with normal TVs can see the usual monocular display.

These features of MPEG-2 are summarized in Table 1.4. How all these features can be combined into a single compression and multiplexing standard for utilization in a variety of applications is the subject of the following chapters.

Table 1.4 Features offered by MPEG-2.

MPEG2 Features
Random access
Trick modes
Multicast to many terminal types
Multiple audio and video
Compatible stereoscopic 3D

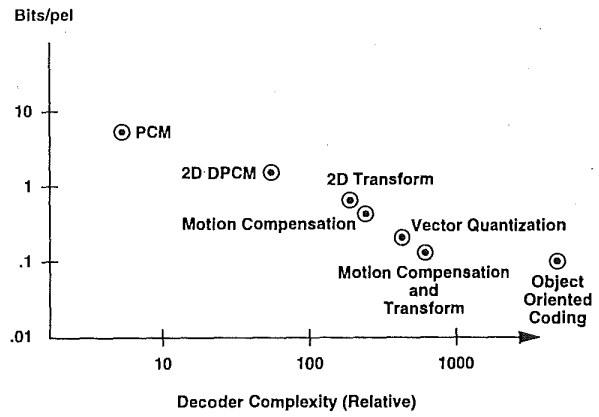


Fig. 1.7 Bits/pel versus complexity of video decoding for several video compression algorithms.

1.7 COMPLEXITY/COST

Since cost is directly linked to complexity, this aspect of a compression algorithm is the most critical for the asymmetrical situations described previously. The decoder cost is most critical. Figure 1.7 represents an approximate tradeoff between the compression efficiency and the complexity under the condition that picture quality is held constant at an 8-bit PCM level. The compression efficiency is in terms of compressed bits per Nyquist sample. Therefore, pictures with different resolution and bandwidth can be compared simply by proper multiplication to get the relevant bitrates. The complexity allocated to each codec should not be taken too literally. Rather, it is an approximate estimate relative to the cost of a PCM codec, which is given a value of 5.

The relation of cost to complexity is controlled by an evolving technology, and codecs* with high complexity are quickly becoming inexpensive through the use of application-specific video DSPs and submicron device technology. In fact, very soon fast microprocessors will be able to decompress the video signal entirely in software. Figure 1.8 shows the computational requirements in millions of instructions per second (MIPS) for encoding and decoding of video at different resolutions. It is clear that in the near future a standard resolution (roughly 500 line by 500 pel TV signal) will be

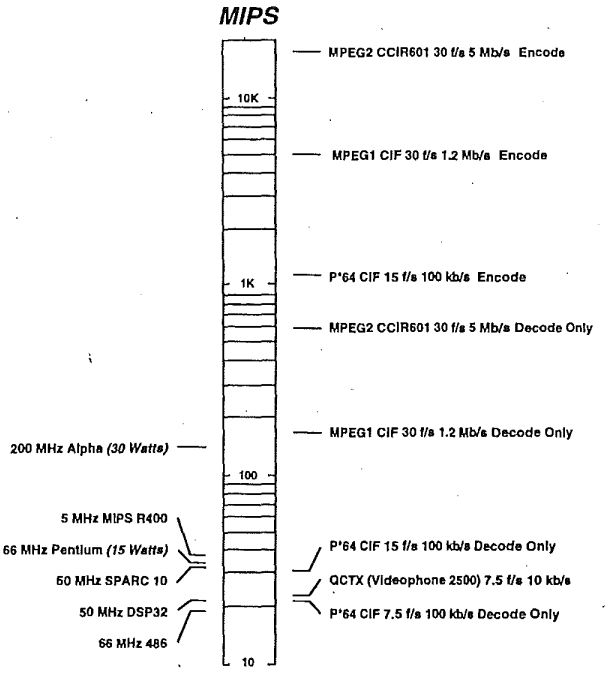


Fig. 1.8 Computational requirements in millions of instructions per second (MIPs) for video encoding and decoding at different image resolutions.

decoded entirely in software for even the MPEG compression algorithm.

On the other hand, the promise of object-based compression systems, in which we recognize objects and code them as a whole instead of coding each pel, will increase the complexity by several orders of magnitude, thereby requiring specialized hardware support. Similarly, higher resolution images (e.g., HDTV[7]) will raise the processing speed requirement even higher. Some object-oriented approaches are only marginally harder to decode, but may require far more analysis at the compressor. Others envisage wholly new architectures for picture reconstruction, thereby requiring a far more complex decoder. The real challenge is to combine the different compression techniques to engineer a cost-effective solution for the given application.

In this chapter, we have examined the advantages of digital representation of the multimedia signal that usually originates as analog signals. Digital representation is consistent with the evolving

*Coder-decoder.

computing/telecommunications infrastructure and to a large extent gives us media independence for storage, processing, and transmission. The bandwidth expansion resulting from digitization can be more than compensated for by efficient compression that removes both the statistical and the perceptual redundancy present in the multimedia signal. Digital compression has many advantages, most of which come as a result of reduced data rates. However, when digital compression is bundled as a part of an application, many constraints need to be attended to.

These constraints impact directly on the choice of the compression algorithm. Many of these constraints are qualitatively examined in this chapter. A more detailed analysis of these will be made in the subsequent chapters.

REFERENCES

1. A. N. Netravali and B. G. Haskell, *Digital Pictures—Representation, Compression and Standards,* 2nd edit., Plenum Press, New York, 1995.
2. W. B. Pennebaker and J. L. Mitchell, *JPEG—Still Image Compression Standard,* Van Nostrand Reinhold, New York, 1993.
3. J. L. Mitchell et al, MPEG-1, Chapman & Hall, New York, 1997.
4. L. Chiariglione, "The Development of an Integrated Audiovisual Coding Standard: MPEG," *Proceedings of the IEEE,* Vol. 83, No. 2, (February 1995).
5. D. J. Le Gall, "MPEG Video: The Second Phase of Work," International Symposium: Society for Information Display, pp. 113–116 (May 1992).
6. A. N. Netravali and A. Lippman, "Digital Television: A perpective," *Proceedings of the IEEE,* Vol. 83, No. 6 (June 1995).
7. R. Hopkins, "Progress on HDTV Broadcasting Standards in the United States," *Signal Processing: Image Commun.,* Vol. 5, pp. 355–378 (1993).
8. A. Puri, "MPEG Video Coding Standards," Invited Tutorial: International Society for Circuits and Systems (April 1995).
9. A. Kopernik, R. Sand, and B. Choquet, "The Future of Three-Dimensional TV," International Workshop on HDTV'92, Signal Processing of HDTV, IV pp. 17–29 (1993).
10. S. Okubo, "Requirements for High Quality Video Coding Standards," *Signal Processing: Image Commun.,* Vol. 4, No. 2, pp 141–151 (April 1992).

2

Anatomy of MPEG-2

After many fits and starts, the era of Digital Television[27] is finally here. Although a variety of audio and video coding standards predate MPEG standardization activity, the MPEG-1 and the MPEG-2 standards are truly unique integrated audio-visual standards.[25,28]

Neither the MPEG-1 nor the MPEG-2 standards prescribe which encoding methods to use, the encoding process, or details of encoders. These standards only specify formats for representing data input to the decoder, and a set of rules for interpreting these data. These formats for representing data are referred to as *syntax* and can be used to construct various kinds of valid data streams referred to as *bitstreams.* The rules for interpreting the data are called *decoding semantics;* An ordered set of decoding semantics is referred to as a *decoding process.* Thus, we can say that MPEG standards specify a decoding process; However, this is still different than specifying a decoder implementation.

Given audio or video data to be compressed, an encoder must follow an ordered set of steps called the *encoding process.* This encoding process, however, is not standardized and typically varies, since encoders of different complexities may be used in different applications. Also, since the encoder is not standardized, continuing improvements in quality are still possible because of encoder optimizations even after the standard is complete. The only constraint is that the output of the encoding process result in a syntactically correct bitstream that can be interpreted by following the decoding semantics by a standards compliant decoder.

2.1 MPEG-1

Although we plan to discuss the anatomy of MPEG-2, we first need to understand the anatomy of MPEG-1. This will allow us to make comparisons between the standards to clearly show similarities and differences, and wherever appropriate, compatibilities between the two standards.

The MPEG-1 standard is formally referred to as ISO 11172 and consists of the following parts:

- 11172-1: Systems
- 11172-2: Video
- 11172-3: Audio
- 11172-4: Conformance
- 11172–5: Software

Initially, the MPEG-1 standard was intended for video coding at bitrates of about 1.2 Mbit/s with stereo audio coding at bitrates of around 250 kbits/s.

2.1.1 MPEG-1 Systems

The MPEG-1 Systems[1] part specifies a system coding layer for combining coded audio and video data and to provide the capability for combining with it user-defined private data streams as well as streams that may be specified at a later time. To be more specific,[12] the MPEG-1 Systems standard defines a packet structure for multiplexing coded audio and video data into one stream and keeping it synchronized. It thus supports multiplexing of multiple coded audio and video streams where each stream is referred to as an *elementary stream.* The systems syntax includes data fields to assist in parsing the multiplexed stream after random access and to allow synchronization of elementary streams, management of decoder buffers containing coded data, and identification of timing information of coded program. The MPEG-1 Systems thus specifies syntax to allow generation of systems bitstreams and semantics for decoding these bitstreams.

Since the basic concepts of timing employed in MPEG-1 Systems are also common to MPEG-2 Systems, we briefly introduce the terminology employed. The Systems Time Clock (STC) is the reference time base, which operates at 90 kHz, and may or may not be phase locked to individual audio or video sample clocks. It produces 33-bit time representations that are incremented at 90 kHz. In MPEG Systems, the mechanism for generating the timing information from decoded data is provided by the *Systems Clock Reference* (SCR) fields that indicate the current STC time and appear intermittently in the bitstream spaced no further than 700 ms apart. The presentation playback or display synchronization information is provided by *Presentation Time Stamps* (PTSs), that represent the intended time of presentation of decoded video pictures or audio frames. The audio or video PTS are sampled to an accuracy of 33 bits.

To ensure guaranteed decoder buffer behavior, MPEG Systems employs a Systems Target Decoder (STD) and Decoding Time Stamp (DTS). The DTS differs from PTS only in the case of video pictures that require additional reordering delay during the decoding process.

2.1.2 MPEG-1 Video

The MPEG-1 Video[2] standard was originally aimed at coding of video of SIF resolution (352 × 240 at 30 noninterlaced frames/s or 352 × 288 at 25 noninterlaced frames/s) at bitrates of about 1.2 Mbit/s. In reality it also allows much larger picture sizes and correspondingly higher bitrates. Besides the issue of bitrates, the other significant issue is that MPEG includes functions to support interactivity such as fast forward (FF), fast reverse (FR), and random access into the stored bitstream. These functions exist since MPEG-1 was originally aimed at digital storage media such as video compact discs (CDs).

The MPEG-1 Video standard basically specifies the video bitstream syntax and the corresponding video decoding process. The MPEG-1 syntax supports encoding methods[13,14,15] that exploit both the spatial redundancies and temporal redundancies. Spatial redundancies are exploited by using block-based Discrete Cosine Transform (DCT) coding of 8 × 8 pel blocks followed by quantization, zigzag scan, and variable length coding of runs of zero quantized indices and amplitudes of these indices. Moreover, quantization matrix allowing perceptually weighted quantization of DCT coefficients can be used to discard perceptually irrelevant information, thus increasing the coding efficiency further. Temporal redundancies are exploited by using motion compensated prediction, which results in a significant reduction of interframe prediction error.

The MPEG-1 Video syntax supports three types of coded pictures, Intra (I) pictures, coded separately by themselves, Predictive (P) pictures, coded with respect to immediately previous I- or P-pictures, and Bidirectionally Predictive (B) pictures coded with respect to the immediate previous I- or P-picture, as well as the immediate next I- or P-picture. In terms of coding order, P-pictures are causal, whereas B-pictures are noncausal and use two surrounding causally coded pictures for prediction. In terms of compression efficiency, I-pictures are least efficient, P-pictures are somewhat better, and B-pictures are the most efficient. In typical MPEG-1 encoding[13,15] an input video sequence is divided

into units of group-of-pictures (GOPs), where each GOP consists of an arrangement of one I-picture, P-pictures, and B-pictures. A GOP serves as a basic access unit, with the I-picture serving as the entry point to facilitate random access. Each picture is divided further into one or more *slices* that offer a mechanism for resynchronization and thus limit the propagation of errors. Each slice is composed of a number of macroblocks; each macroblock is basically a 16 × 16 block of luminance pels (or alternatively, four 8 × 8 blocks) with corresponding chrominance blocks. MPEG-1 Video encoding is performed on a macroblock basis. In P-pictures each macroblock can have one motion vector, whereas in B-pictures each macroblock can have as many as two motion vectors.

2.1.3 MPEG-1 Audio

The MPEG-1 Audio[3] standard basically specifies the audio bitstream syntax and the corresponding audio decoding process. MPEG-1 Audio is a generic standard and does not make any assumptions about the nature of the audio source, unlike some vocal-tract-model coders that work well for speech only. The MPEG syntax permits exploitation of perceptual limitations of the human auditory system while encoding, and thus much of the compression comes from removal of perceptually irrelevant parts of the audio signal. MPEG-1 Audio coding[16,23] allows a diverse assortment of compression modes and in addition supports features such as random access, audio fast forwarding, and audio fast reverse.

The MPEG-1 Audio standard consists of three Layers—I, II, and III. These Layers also represent increasing complexity, delay, and coding efficiency. In terms of coding methods, these Layers are somewhat related, since a higher Layer includes the building blocks used for a lower Layer. The sampling rates supported by MPEG-1 Audio are 32, 44.2, and 48 kHz. Several fixed bitrates in the range of 32 to 224 kbits/s per channel can be used. In addition, Layer III also supports variable bitrate coding. Good quality is possible with Layer I above 128 kbits/s, with Layer II around 128 kbit/s, and with Layer III at 64 kbit/s. MPEG-1 Audio has four modes: mono, stereo, dual with two separate channels, and joint stereo. The optional joint stereo mode exploits interchannel redundancies.

A polyphase filter bank is common to all Layers of MPEG-1 Audio coding. This filter bank subdivides the sudio signal into 32 equal-width frequency subbands. The filters are relatively simple and provide good time-resolution with a reasonable frequency-resolution. To achieve these characteristics, some exceptions had to be made. First, the equal widths of subbands do not precisely reflect the human auditory system's frequency-dependent behavior. Second, the filter bank and its inverse are not completely lossless transformations. Third, adjacent filter bands have significant frequency overlap. However, these exceptions do not impose any noticeable limitations, and quite good audio quality is possible. The Layer I algorithm codes audio in frames of 384 samples by grouping 12 samples from each of the 32 subbands. The Layer II algorithm is a straightforward enhancement of Layer I; it codes audio data in larger groups (1 152 samples per audio channel) and imposes some restrictions on possible bit allocations for values from the middle and higher subbands. The Layer II coder gets better quality by redistributing bits to better represent quantized subband values. The Layer III algorithm is more sophisticated and uses audio spectral perceptual entropy coding and optimal coding in the frequency domain. Although based on the same filter bank as used in Layer I and Layer II it compensates for some deficiencies by processing the filter outputs with a modified DCT.

2.2 MPEG-2

After this brief overview of the parts of the MPEG-1 standard, we are now well prepared to delve into a discussion of the various parts of the MPEG-2 standard. This standard is formally referred to as ISO 13818 and consists of the following parts:

- 113818-1: Systems
- 113818-2: Video
- 113818-3: Audio
- 113818-4: Conformance
- 113818-5: Software

- 113818-6: Digital Storage Media—Command and Control (DSM-CC)
- 113818-7: Non Backward Compatible (NBC) Audio
- 113818-8: 10-Bit Video (This work item has been dropped!)
- 113818-9: Real Time Interface
- 113818-10: Digital Storage Media—Command and Control (DSM-CC) Conformance

We now present a brief overview of the aforementioned parts of the MPEG-2 standard.

2.2.1 MPEG-2 Systems

Since the MPEG-1 standard was intended for audio–visual coding for Digital Storage Media (DSM) applications and since DSMs typically have very low or negligible transmission bit error rates, the MPEG-1 Systems part was not designed to be highly robust to bit errors. Also, the MPEG-1 Systems was intended for software oriented processing, and thus large variable length packets were preferred to minimize software overhead.

The MPEG-2 standard on the other hand is more generic and thus intended for a variety of audio–visual coding applications. The MPEG-2 Systems[4] was mandated to improve error resilience plus the ability to carry multiple programs simultaneously without requiring them to have a common time base. Additionally, it was required that MPEG-2 Systems should support ATM networks. Furthermore, MPEG-2 Systems was required to solve the problems addressed by MPEG-1 Systems and be somewhat compatible with it.

The MPEG-2 Systems[17,18] defines two types of streams: the Program Stream and the Transport Stream. The Program Stream is similar to the MPEG-1 Systems stream, but uses a modified syntax and new functions to support advanced functionalities. Further, it provides compatibility with MPEG-1 Systems streams. The requirements of MPEG-2 Program Stream decoders are similar to those of MPEG-1 System Stream decoders. Furthermore, Program Stream decoders are expected to be forward compatible with MPEG-1 System Stream decoders, i.e., capable of decoding MPEG-1 System Streams. Like MPEG-1 Systems decoders, Program streams decoders typically employ long and variable-length packets. Such packets are well suited for software based processing and error-free environments, such as when the compressed data are stored on a disk. The packet sizes are usually in range of 1 to 2 kbytes chosen to match disk sector sizes (typically 2 kbytes); however, packet sizes as large as 64 kbytes are also supported. The Program Stream includes features not supported by MPEG-1 Systems such as scrambling of data; assignment of different priorities to packets; information to assist alignment of elementary stream packets; indication of copyright; indication of fast forward, fast reverse, and other trick modes for storage devices; an optional field for network performance testing; and optional numbering of sequence of packets.

The second type of stream supported by MPEG-2 Systems is the Transport Stream, which differs significantly from MPEG-1 Systems as well as the Program Stream. The Transport Stream offers robustness necessary for noisy channels as well as the ability to include multiple programs in a single stream. The Transport Stream uses fixed length packets of size 188 bytes, with a new header syntax. It is therefore more suited for hardware processing and for error correction schemes. Thus the Transport Stream stream is well suited for delivering compressed video and audio over error-prone channels such as coaxial cable television networks and satellite transponders. Furthermore, multiple programs with independent time bases can be multiplexed in one Transport Stream. In fact the Transport Stream is designed to support many functions such as asynchronous multiplexing of programs, fast access to desired program for channel hopping, multiplex of programs with clocks unrelated to transport clock, and correct synchronization of elementary streams for playback. It also allows control of decoder buffers during startup and playback for both constant bitrates and variable bitrate programs.

A basic data structure that is common to the organization of both the Program Stream and Transport Stream data is called the Packetized Elementary Stream (PES) packet PES packets are generated by packetizing the continuous streams of compressed data generated by video and audio

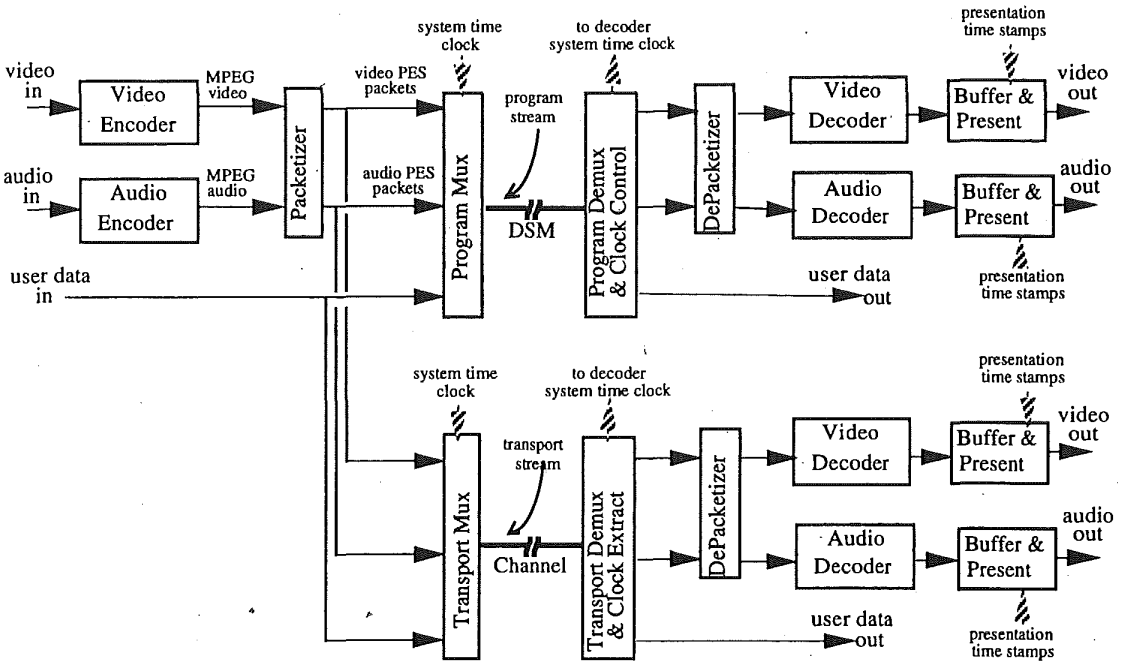


Fig. 2.1 MPEG-2 Systems Multiplex showing Program and Transport Streams.

(i.e., elementary stream) encoders. A Program Stream is generated simply by stringing together PES packets from the various encoders with other packets containing necessary data to generate a single bitstream. A Transport Stream consists of packets of fixed length consisting of 4 bytes of header followed by 184 bytes of data, where data is obtained by chopping up data in PES packets.

In Fig. 2.1 we illustrate both types of MPEG-2 Systems, the ones using Program Stream multiplex and the ones using Transport Stream multiplex. An MPEG-2 System is also capable of combining multiple sources of user data along with the MPEG encoded audio and video. The audio and video streams are packetized to form audio and video PES packets that are then sent to either a Program Multiplexer or a Transport Multiplexer, resulting in a Program Stream or Transport Stream, respectively. As mentioned earlier, Program Streams are intended for error-free environments such as DSMs, whereas Transport Streams are intended for noisier environments such as terrestrial broadcast channels.

Transport Streams are decoded by a Transport Demultiplexer (which includes a clock extraction mechanism), unpacketized by a DePacketizer, and sent for audio and video decoding to Audio and Video Decoders. The decoded signals are sent to respective Buffer and Presentation units that output them to a display device and speaker at the appropriate times. Similarly, if Program Streams are employed, they are decoded by a Program Stream Demultiplexer, DePacketizer, and sent for decoding to Audio and Video Decoders. The decoded signals are sent to respective Buffer and Present and await presentation. Also, as with MPEG-1 Systems, the information about systems timing is carried by Clock Reference fields in the bitstream that are used to synchronize the decoder Systems Time Clock (STC). The presentation of decoded output is controlled by Presentation Time Stamps (PTSs) that are also carried by the bitstream.

2.2.2 MPEG-2 Video

This part of MPEG-2, part 2, addresses coding of video and is referred to as MPEG-2 Video. Originally the MPEG-2 Video[5] standard was primarily intended for coding of interlaced video of standard TV resolution with good quality in the bitrate range of 4 to 9 Mbit/s. However, the scope

of MPEG-2 Video was considerably revised[20] to include video of higher resolutions such as HDTV at higher bitrates as well as hierarchical video coding for a range of applications. Furthermore, it also supports coding of interlaced video of a variety of resolutions.

Actually, MPEG-2 Video does not standardize video encoding; only the video bitstream syntax and decoding semantics are standardized. The standardization process for MPEG-2 Video consisted of a competitive phase followed by a collaborative phase. First an initial set of requirements for MPEG-2 video was defined and used as the basis for a "Call for Proposals" which invited candidate coding schemes for standardization. These schemes underwent formal subjective testing and analysis to determine if they met the listed requirements. During the collaborative phase, the best elements of the top performing schemes in the competitive phase were merged to form an initial *Test Model.* A Test Model contains a description of an encoder, bitstream syntax, and decoding semantics. The encoder description is not to be standardized, but only needed for experimental evaluation of proposed techniques.

Following this, a set of *Core Experiments* were defined that underwent several iterations until convergence was achieved. During the time that various iterations on the Test Model[19] were being performed, the requirements for MPEG-2 Video also underwent further iterations.

In Fig. 2.2 we show a simplified codec[19,21] for MPEG-2 Nonscalable Video Coding. The MPEG-2 Video Encoder shown for illustration purposes consists of an Inter frame/field DCT Encoder, a Frame/field Motion Estimator and Compensator, and a Variable Length Encoder (VLE). Earlier we had mentioned that MPEG-2 Video is optimized for coding of interlaced video, which is why both the DCT coding and the motion estimation and compensation employed by the Video Encoder are frame/field adaptive. The Frame/field DCT Encoder exploits spatial redundancies, and the Frame/field Motion Compensator exploits temporal redundancies in interlaced video signal. The coded video bitstream is sent to a systems multiplexer, Sys Mux, which outputs either a Transport or a Program Stream.

The MPEG-2 Decoder in this codec consists of a Variable Length Decoder (VLD), Inter frame/field DCT Decoder, and the Frame/field Motion Compensator. Sys Demux performs the complementary function of Sys Mux and presents the video bitstream to VLD for decoding of motion vectors and DCT coefficients. The Frame/field Motion Compensator uses a motion vector decoded by VLD to generate motion compensated prediction that is added back to a decoded prediction error signal to generate decoded video out. This type of coding produces video bitstreams called nonscalable, since normally the full spatial and temporal resolution coded is the one that is expected to be decoded. Actually, this is not quite true, since, if B-pictures are used in encoding, because they do not feedback into the interframe coding loop, it is always possible to decode some of them, thus achieving temporal resolution scaling even for nonscalable bitstreams. However, by nonscalable cod-

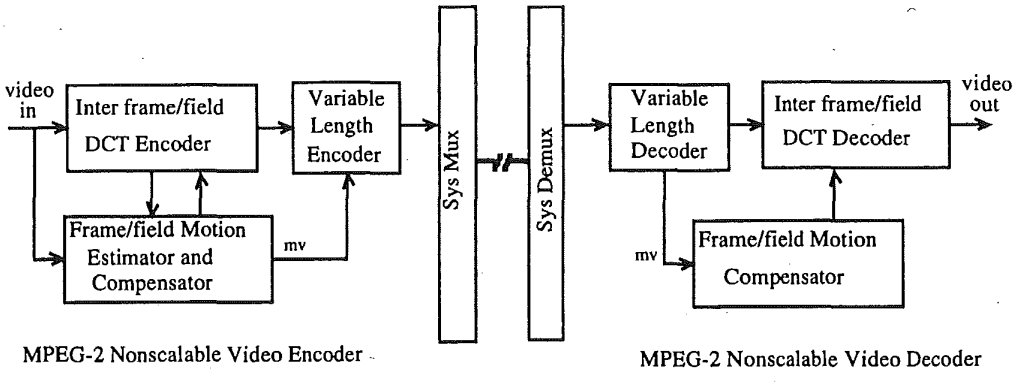


Fig. 2.2 A generalized codec for MPEG-2 Nonscalable Video Coding.

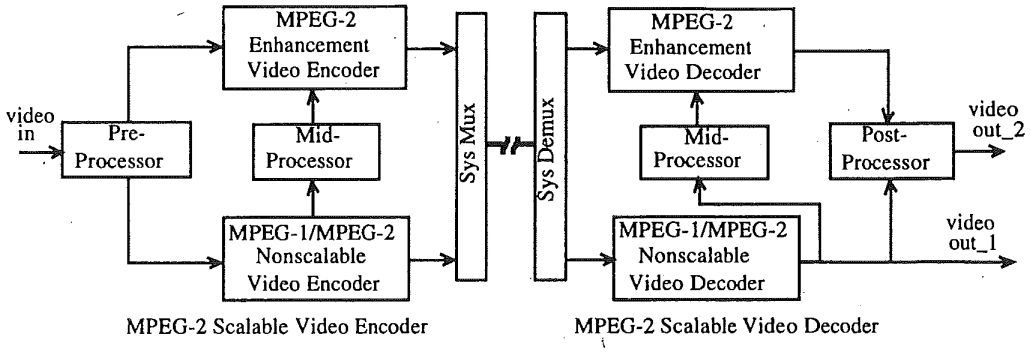


Fig. 2.3 A generalized codec for MPEG-2 Scalable Video Coding.

ing we usually mean that we have not gone out of our way to facilitate scalability.

As you may have guessed by this time, the MPEG-2 Video standard specifies the bitstream syntax and decoding process not only for single layer (MPEG-1 like) video coding but also for scalable video coding. Scalability is the property that allows decoders of various complexities to be able to decode video of resolution/quality commensurate with their complexity from the same bitstream. We will discuss a lot more about MPEG-2 scalable video coding in Chapter 9, but for now we illustrate the general principle by introducing the generalized scalable codec structure shown in Fig. 2.3. In MPEG-2 Video there are many different types of scalability and thus it is difficult to represent all of them with a single generalized codec. Our generalized codec structure basically implies spatial and temporal resolution scalability.

Input video goes through a Pre-Processor that produces two video signals, one of which (the *Base Layer*) is input to an MPEG-1 or MPEG-2 Nonscalable Video Encoder and the other (the *Enhancement Layer*) input to an MPEG-2 Enhancement Video Encoder. Some processing of decoded video from MPEG-1 or MPEG-2 Nonscalable Video Encoder may be needed in the Mid-Processor depending on specific scalability. The two bitstreams, one from each encoder, are multiplexed in Sys Mux (along with coded audio and user data). Thus it becomes possible for two types of decoders to be able to decode a video signal of quality commensurate with their complexity, from the same encoded bitstream. For example, if an MPEG-1 or MPEG-2 Nonscalable Video Decoder is employed, a basic video signal can be decoded. If in addition, an MPEG-2 Enhancement Video Decoder is employed, an enhanced video signal can be decoded. The two decoded signals may undergo further processing in a Post-Processor.

Since the MPEG-2 Video standard, to be truly generic, had to include a bitstream and semantics description for a variety of coding methods and tools, the implementation of all its features in every decoder was considered too complex and thus the standard was partitioned into Profiles. In MPEG-2 Video, applications with somewhat related requirements are addressed via Profiles; a Profile typically contains a collection of coding techniques (or tools) designed to address a set of such applications. We will discuss a lot more about Requirements and Profiles in Chapter 11.

The MPEG-2 Video standard is a syntactic superset of the MPEG-1 Video standard and is thus able to meet the requirement of forward compatibility, meaning that an MPEG-2 video decoder should be capable of decoding MPEG-1 video bitstreams. The requirement of backward compatibility, meaning that subsets of MPEG-2 bitstreams should be decodable by existing MPEG-1 decoders, is achieved via the use of scalability and supported in specific profiles. To confirm that the MPEG-2 Video standard met its quality objective for various Profiles and for individual applications within a profile, a series of verification tests have been conducted in association with other international standardization bodies and a number of organizations. Thus far, these tests confirm that MPEG-2 Video does indeed meet or exceed the performance bounds of its target applications. MPEG-2 Video

quality has also been judged sufficient for HDTV.[34] Recently, newer applications of MPEG-2 Video have also emerged, necessitating combinations of coding tools requiring new Profiles.

At the time of writing of this chapter in early 1996, two new amendments in MPEG-2 video, each adding a new profile, are in progress. One of the two amendments is nearly complete, and the second one is expected to be completed by the end of 1996. Although MPEG-2 video already includes various coding techniques (decoding techniques, really), in reality, these techniques must be included in one of the agreed Profiles also (these profiles can even be defined after completion of the standard). Next, we briefly introduce what application areas are being targeted by these two amendments.

The first amendment to MPEG-2 Video involves a Profile that includes tools for coding of a higher resolution chrominance format called the 4:2:2 format. We will discuss more about this format in Chapter 5; for now, we need to understand why it was relevant for MPEG-2 video to address applications related to this format. When work on MPEG-2 Video was close to completion, it was felt that MPEG-2 video was capable of delivering fairly high quality, and thus it could be used in professional video applications. In sorting out needs of such applications, besides higher bitrates, it was found that some professional applications require higher than normal chrominance spatial (or spatio-temporal) resolution, such as that provided by the 4:2:2 format while another set of applications required higher amplitude resolution for luminance and chrominance signal. Although coding of the 4:2:2 format video is supported by existing tools in the MPEG-2 Video standard, it was necessary to verify that its performance was equal to or better than that of other coding schemes while fulfilling additional requirements such as coding quality after multiple codings and decodings.

The second amendment[32] to MPEG-2 Video involves a Profile that includes tools for coding of a number of simultaneous video signals generated in imaging a scene, each representing a slightly different viewpoint; imaging of such scenes is said to constitute multiviewpoint video. An example of multiviewpoint video is stereoscopic video which uses two slightly different views of a scene. Another example is generalized 3D video where a large number of views of a scene may be used. We will discuss more about this profile in Chapter 5 and a lot more about 3D/stereoscopic video in Chapter 15; for now, as in the case of 4:2:2 coding we need to understand why it was relevant to address multiviewpoint coding applications. As mentioned earlier, MPEG-2 Video already includes several Scalability techniques that allow layered coding which exploits correlations between layers. As the result of a recent rise in applications in video games, movies, education, and entertainment, efficient coding of multiviewpoint video such as stereoscopic video is becoming increasingly important. Not surprisingly, this involves exploiting correlations between different views, and since scalability techniques of MPEG-2 Video already included such techniques, it was considered relatively straightforward to enable use of such techniques via definition of a Profile (which is currently in progress).

2.2.3 MPEG-2 Audio

Digital multichannel audio systems employ a combination of p front and q back channels; for example, three front channels—left, right, center—and two back channels—surround left and surround right—to create surreal experiences. In addition, multichannel systems can also be used to provide multilingual programs, audio augmentation for visually impaired individuals, enhanced audio for hearing impaired individuals, and so forth.

The MPEG-2 Audio[6] standard consists of two parts, one that allows coding of multichannel audio signals in a forward and backward compatible manner with MPEG-1 and one that does not. Part 3 of the MPEG-2 standard is forward and backward compatible with MPEG-1. Here forward compatibility means that the MPEG-2 multichannel audio decoder can decode MPEG-1 mono or stereo audio signals. Backward compatibility (BC) means that an MPEG-1 stereo decoder can reproduce a meaningful downmix of the original five channels from the MPEG-2 audio bitstream.

While forward compatibility is not as hard to achieve, backward compatibility is a much more

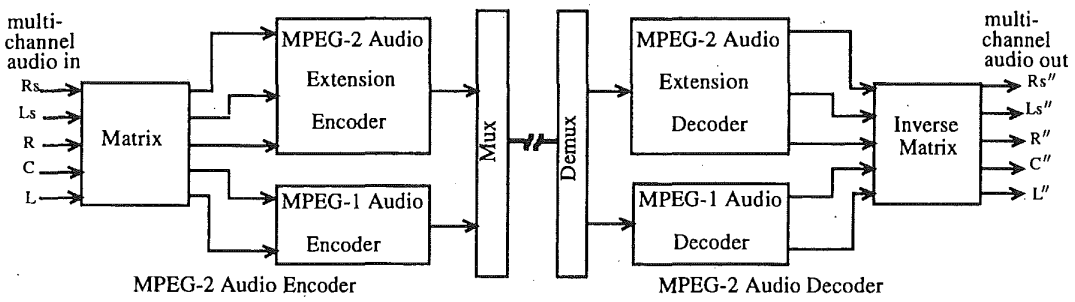


Fig. 2.4 A generalized codec for MPEG-2 Backward Compatible (BC) Multichannel Audio Coding.

difficult and requires some compromise in coding efficiency. However, backward compatibility does offer the possibility of migration to multichannel audio without making existing MPEG-1 stereo decoders obsolete.

Initially, the primary requirements identified for MPEG-2 Audio were multichannel coding and compatibility with MPEG-1 Audio. Candidate coding schemes were invited via a "Call for Proposals." In response, the various proposed solutions were found to be very close to each other, and the collaborative phase was begun without the need for competitive tests.

A generalized codec[24] structure illustrating MPEG-2 Multichannel Audio coding is shown in Fig. 2.4. A Multichannel Audio consisting of five signals, left (L), center (C), right (R), left surround (Ls), and right surround (Rs) is shown undergoing conversion by use of a Matrix operation resulting in five converted signals, two of which are encoded by an MPEG-1 Audio Encoder to provide compatibility with the MPEG-1 standard. The remaining three signals are encoded by an MPEG-2 Audio Extension Encoder. The resulting bitstreams from the two encoders are multiplexed in Mux for storage or transmission.

Since it is possible to have coded MPEG-2 Audio without coded MPEG Video, a generalized multiplexer Mux is shown. However, in an MPEG Audio–Visual System, the specific Mux and Demux used would be MPEG-2 Sys Mux and MPEG-2 Sys Demux. At the decoder, an MPEG-1 Audio Decoder decodes the bitstream input to it by Sys Demux and produces two decoded audio signals; the other three audio signals are decoded by an MPEG-2 Audio Extension Decoder. The decoded audio signals are reconverted back to original domain by using an Inverse Matrix and represent approximated values indicated by L″, C″, R″, Ls″, Rs″.

To verify the syntax two type of tests were conducted: first, bitstreams produced by software encoding were decoded in non-real-time by software decoders, and second, bitstreams produced by software encoders were decoded in real-time. Tests were also conducted comparing the performance of MPEG-2 Audio coders that maintain compatibility with MPEG-1 to that which are not backward compatible. It has been found that for the same bitrate, compatibility does impose a quality loss which may be difficult to justify in some applications. Hence, it has been found necessary to also include a nonbackward compatible coding solution, which is referred to as MPEG-2 Part 7-NBC Audio; further elaboration regarding this work, which is currently in progress, is provided in our discussion on part 7.

Both the MPEG-1 and MPEG-2 Audio coding is discussed in much more detail in Chapter 4.

2.2.4 MPEG-2 Conformance

This part of MPEG-2, part 4, specifies Conformance[7] to the MPEG-2 standard; it is also called Compliance. It specifies how tests can be designed to verify whether bitstreams and decoders meet the requirements as specified in parts 1, 2, and 3. These tests can be used for a variety of purposes, for example, manufacturers of encoders and their customers can ensure whether an encoder produces valid bitstreams, manufacturers of decoders and their customers can verify if decoders meet

requirements, and applications users can use tests to verify whether characteristics of a bitstream meet the application requirements.

Since MPEG-2, like MPEG-1, does not specify an encoder, and for that matter not even a specific decoder implementation but only bitstream syntax and decoding semantics, it is essential to know numerical bounds that various interim computations in the decoding process must satisfy so that it can be assured that the same bitstream decoded on different decoders will provide nearly identical results. If certain numerical bounds in interim processing calculations are not met, decoding on different decoders may produce not only slightly different results, but worse, certain errors in calculations may accumulate over time, eventually resulting in significant degradation in quality. This part of the standard therefore specifies bounds to be met in various steps of video and audio decoding so that errors do not accumulate. Being able to identify a range of numerical bounds on every calculation that goes on in an audio and video decoding process in real hardware is a challenging task since it implicitly requires a knowledge of types of errors and causes of such errors in terms of hardware operations that occur. Furthermore, it is difficult to be ever sure that all potential restrictions on error accumulations have been identified. Nevertheless, MPEG-2 technology has undertaken a significant effort to devise conformance bounds and to design a suite of test bitstreams that have been generated with great care to illustrate potential difficulties in conformance of a decoder. These bitstreams are referred to as "evil bitstreams," and as its name suggests are designed to test conformance of decoders to the MPEG-2 standard almost to the breaking point.

Since part 4 is intended to provide conformance for all the three primary parts of MPEG-2—Systems, Video, and Audio–it provides bitstream characteristics, bitstream tests, decoder characteristics, and decoder tests for these three parts.

In Fig. 2.5 we show an example of a setup for testing video decoder compliance. It is assumed that a Reference Decoder exists that satisfies the criterion for conformance to the MPEG-2 standard. Although ideally conformance testing should be possible on every type of decoder, MPEG-2 specification concerns only the decoding process from a bitstream of a digital output. For example, in testing MPEG-2 Video conformance for decoders supporting MPEG-1 like chrominance resolution (4:2:0), it is expected that the digital output from the Decoder under Test be 4:2:0 for comparison with the Reference Decoder providing a 4:2:0 digital output. In fact the proposed procedure can also be used for decoder tests when the digital output is 4:2:2 (as may be the case when higher chroma resolution is obtained by upsampling 4:2:0 for display); however, the results on the 4:2:2 signal need to be carefully interpreted (assuming that 4:2:0 resolution is coded in the bitstream) since chroma conversion from 4:2:0 to 4:2:2 is outside of the scope of the standard.

The Statistical Analysis operation generates a report on the difference file for each conformance bitstream input. The report consists of first, a histogram of pel differences per field between Reference Decoder and Decoder under Test and second, the number and addresses of erroneous macroblocks and the erroneous component in each erroneous macroblock. Furthermore, if more than 90% of macroblocks are declared erroneous in a picture for a specific bitstream, a complete break-

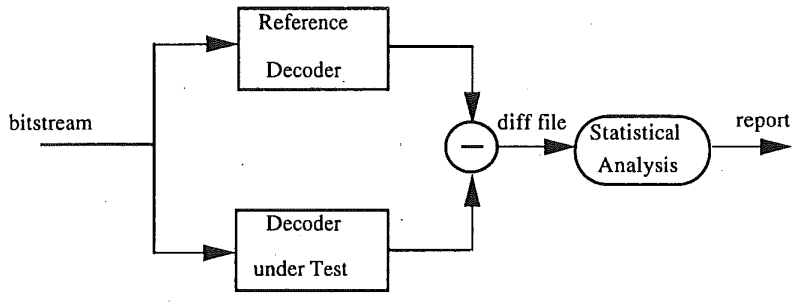


Fig. 2.5 Setup for testing Decoder Compliance.

down is said to have occurred. Other types of breakdown are also identified such as a short breakdown (the decoder does not decode the end of picture and resynchronizes on the next picture) and long breakdown (the decoder loses the information of a picture and resynchronizes on the next I-picture, next GOP, or next sequence startcode). In case of probable breakdown on a conformance bitstream a new stream is created by concatenation of an easy bitstream followed by the conformance bitstream followed by an easy bitstream. A visual test is then conducted to determine if concatenated parts are visually different and if the decoder restarts after a breakdown.

Besides the tests for video decoders, a number of tests are also specified for audio decoders. A suite of audio test sequences has also been standardized and covers various layer decoders. Testing can be done as for video by comparing the output of a Decoder under Test with the output of a Reference Decoder. To be called an MPEG-2 Audio decoder, the decoder shall provide an output such that the root-mean-square level of difference signal between the output of the Decoder under Test and the supplied Reference Decoder output is less than a specified amount for a supplied sine sweep signal. In addition, the difference signal magnitude must have less than a maximum absolute value. Further, to be called a limited accuracy MPEG-2 Audio decoder, a less strict limit is placed on the root-mean-square level of difference signal.

This part is expected to achieve the final status of International Standard by the end of the first quarter of 1996.

2.2.5 Software

This part of MPEG-2, part 5, is referred to as Software Simulation.[8] As the name suggests, it consists of software simulating Systems, Video, and Audio, the three principal parts of the MPEG-2 standard.

The MPEG-2 Systems Software consists of software for encoding and decoding of both the Program and the Transport Streams.

The MPEG-2 Video Software mainly consists of software for encoding and decoding of nonscalable video. In addition, it also includes some software for scalable encoding and decoding.

The MPEG-2 Audio Software consists of software for encoding and decoding. Two different psychoaccoustical models are included in the encoder.

MPEG-2 Software has in early 1996 achieved the final status of International Standard. The source code included in this part of MPEG-2 is copyrighted by the ISO.

2.2.6 DSM-CC

Coded MPEG-2 bitstreams may typically be stored in a variety of digital storage media such as CD-ROM, magnetic tape and disks, optical disks, magneto-optical disks, and others. This presents a problem for users wishing to access coded MPEG-2 data stored on a Digital Storage Media (DSM), since each DSM usually has its own control command language, requiring the user to know many such languages. Moreover, the DSM may be either local to the user or at a remote location. When remote, a common mechanism to access various DSM over a network is needed; otherwise the type of DSM being accessed has to be conveyed to the user. This, however, may not even be known in many cases. A solution to these two problems is offered by MPEG-2 DSM-CC.[1,9,10]

MPEG-2 DSM-CC is a set of generic control commands independent of the type of DSM. The DSM-CC is intended for MPEG applications that need to access a variety of DSM without the need to know details of each DSM. Thus, to allow a set of basic functions specific to MPEG bitstreams on a DSM, control commands are defined as a specific application protocol. The resulting control commands do not depend on the type of the DSM or whether a DSM is local or remote, the network transmission protocol, or the operating system with which it is interfacing. The control command functions can be performed on MPEG-1 systems bitstreams, MPEG-2 Program streams, or MPEG-2 Transport Streams. Some example functions are connection, playback, storage, editing, remultiplexing etc. A basic set of control commands is also included as an informative (nonmandatory) annex in MPEG-2 Systems, part 1.

Depending on the requirements of an application and the complexity that can be handled, DSM control commands can be divided into two categories. In the first category are a set of very basic operations such as selection of stream, play, and store commands. The stream selection command is used to request a specific bitstream and a specific operation mode on that bitstream. The play command is used to request playing of a selected bitstream and involves specification of speed and direction of play to accommodate various trick modes such as fast forward, fast reverse, pause, resume, step through, or stop. The store command is used to request recording of a bitstream on DSM.

The second category consists of a set of more advanced operations such as multiuser mode, session reservation, server capability information, directory information, bitstream editing, and others. With the multiuser mode command, more than one user is allowed to access the same server within a session. With the session reservation command, a user can request a server for a session at a later time. The server capability information command allows the user to be notified of the capabilities of the server; example capabilities include playback, fast forward, fast reverse, slow motion, storage, demultiplex, remultiplex, and others. The directory information command allows user access to information about directory structure and specific attributes of a bitstream such as bitstream type, IDs, sizes, bitrate, entry points for random access, program descriptors, and others. Typically, all this information may not be available through an Application Program Interface (API). Bitstream editing functions allow creation of new bitstreams by insertion of a portion of one bitstream into another bitstream, deletion of a portion of a bitstream, and correctly concatenating portions of various bitstreams.

In Fig. 2.6 we show a simplified relationship of DSM-CC with MHEG (Multimedia and Hypermedia Experts Group standard) Scripting Language, and DSM-CC. The MHEG standard is basically an interchange format for multimedia objects between applications. MHEG specifies a class set that can be used to specify objects containing monomedia information, relationships between objects, dynamic behavior between objects, and

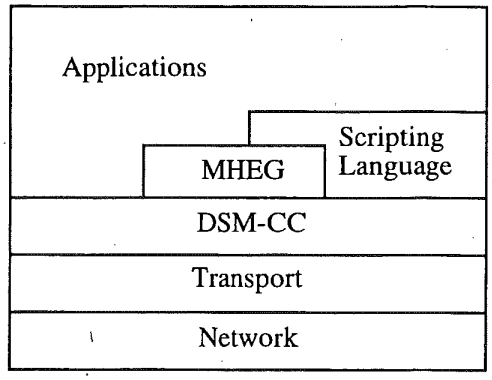


Fig. 2.6 DSM-CC centric View of MHEG, Scripting Language, and Network.

information to optimize real-time handling of objects. MHEG's classes include the content class, composite class, link class, action class, script class, descriptor class, container class, and result class. Furthermore, the MHEG standard does not define an API for handling of objects on its classes nor does it define methods on its classes. Finally, one of the important things to note is that MHEG explicitly supports scripting languages through the use of a Script Class; however, it does not specify a scripting language of its own.

Returning to our discussion on DSM-CC and its relationship to MHEG, applications may access DSM-CC either directly or through an MHEG layer. Moreover, scripting languages may be supported through an MHEG layer on top of DSM-CC. DSM-CC protocols form a layer higher than Transport Protocols. Examples of Transport Protocols are TCP, UDP, and MPEG-2 Transport Stream/Program Streams.

DSM-CC provides access for general applications, MHEG applications, and scripting languages to primitives for establishing or deleting network connections using User-Network (U–N) Primitives and communication between a Client and a Server across a network using User–User (U–U) Primitives. U–U operations may use a Remote Procedure Call (RPC) protocol. Both U–U and U–N operations may employ a message passing scheme that involves a sequence of bit-pattern exchanges.

In Fig. 2.7 we show such a stack of DSM protocols consisting of primitives allowing communications between User–User and User–Network.

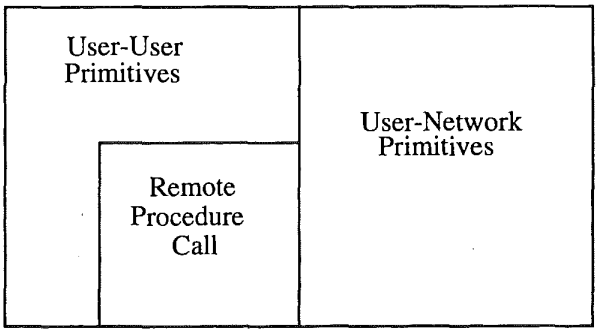


Fig. 2.7 DSM-CC protocol stack.

Next in Fig. 2.8 we clarify DSM-CC User–Network and User–User interaction. A Client can connect to a Server either directly via a Network or through a Resource Manager located within the Network.

A Client setup typically is expected to include a Session Gateway, which is a User–Network interface point, and a library of DSM-CC routines, which is a User–User interface point. A Server setup typically consists of a Session Gateway, which is a User–Network interface point, and a Service Gateway, which is a User–User interface point. Depending on the requirements of an application both a User–User connection and a User–Network connection can be established.

DSM-CC may be carried as a stream within an MPEG-1 Systems Stream, an MPEG-2 Transport Stream, or an MPEG-2 Program Stream. Alternatively, DSM-CC can also be carried over other transport networks such as TCP or UDP Internet Protocols.

DSM-CC is expected to achieve the final status of International Standard by mid 1996.

2.2.7 MPEG-2 NBC Audio

Earlier, while discussing part 3 of MPEG-2, we mentioned that MPEG-2 audio comes in two parts, one that is backward compatible with MPEG-1 (part 3), and one that is not. As you may have guessed, this part, part 7, is not backward compatible with MPEG-1 and is thus referred to as Non Backward Compatible (NBC) Audio. Thus this part of MPEG-2 is intended for applications that do not require compatibility with MPEG-1 stereo and thus do not need to compromise on coding efficiency at all.

MPEG-2 NBC Audio is expected to support the sampling rates, audio bandwidth, and channel configurations of MPEG-2 Audio, but will operate at bitrates from 32 kbit/s to the bitrate required for high-quality audio. This work item was started late, around the time when work for part 3 was in an advanced stage, and so this work item is currently still in progress.[26,30] It is expected to achieve a draft standard status by the end of 1996.

The results from NBC Audio tests[33] indicate that notably better quality is obtained by nonbackward compatible coding as compared to backward compatible coding. About 10 organizations participated in NBC Audio competitive tests. A collaborative phase is currently in progress, including the definition of an initial Reference Model that is the framework for conducting core experiments. It is

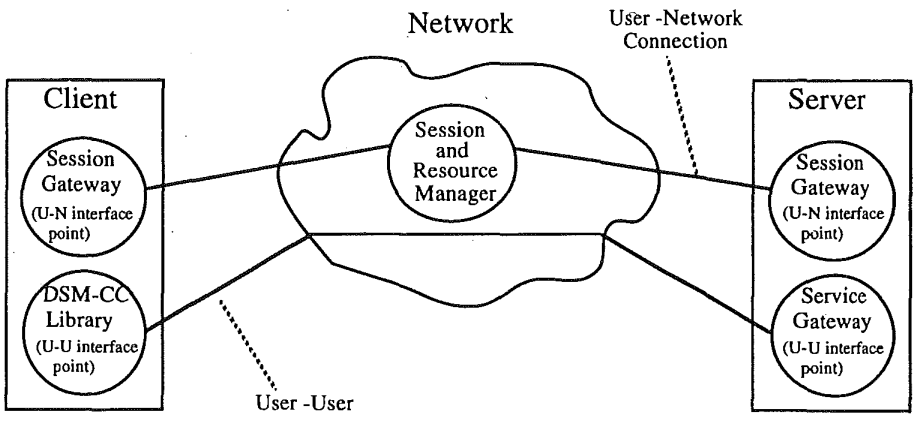


Fig. 2.8 DSM-CC User–Network and User–User interaction.

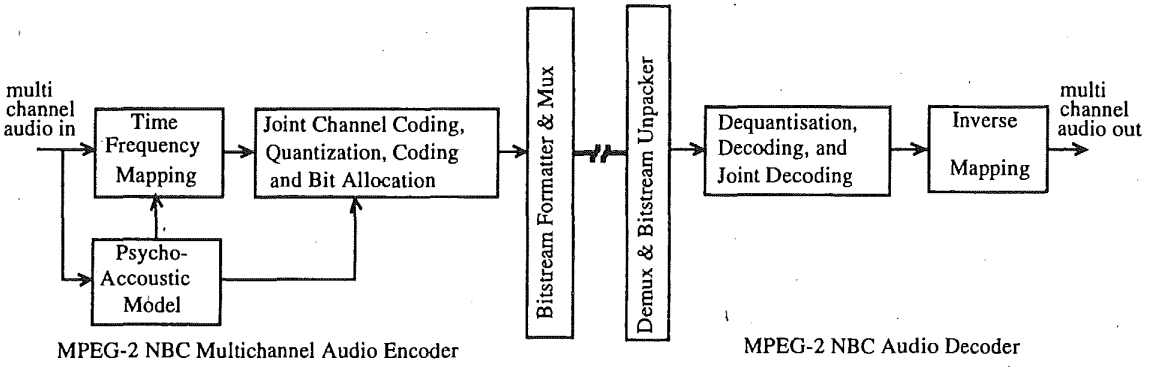


Fig. 2.9 A generalized Codec for MPEG-2 Non Backward Compatible (NBC) Multichannel Audio Coding.

expected to undergo several iterations before finally achieving convergence.

In Fig. 2.9 we show a simplified Reference Model configuration of the NBC Audio coder under consideration. We now briefly introduce the structure employed without going into specific details.

Multichannel audio undergoes conversion of domain in Time-Frequency Mapping, whose output is subject to various operations currently being optimized via experiments such as Joint Channel Coding, Quantization, Coding, and Bit Allocation. A psychoacoustical model is employed at the encoder. A Bitstream Formatter generates the bitstream for storage or transmission. At the decoder, an inverse operation is performed by a Bitstream Unpacker, which is followed by Dequantization, Decoding, and Joint Decoding. Finally, an Inverse Mapping is performed to go back from frequency domain to time domain representation, resulting in the reconstructed multichannel audio output.

NBC Audio is expected to achieve the mature stage of Committee Draft by mid 1996 and the final status of International Standard by the end of the first quarter of 1997.

2.2.8 10-Bit Video (Cancelled!)

Earlier, during the discussion of part 2, we had mentioned that when MPEG-2 video work was close to completion, it was envisaged that the MPEG-2 video would be useful in professional applications because of its ability to achieve high quality. The two different types of requirements for these applications were found to be higher spatial resolution for chrominance (4:2:2 format) and higher amplitude resolution for luminance and chrominance, called the 10-bit video. In 10-bit video, as the name suggests, each video sample is digitized to 10-bit accuracy rather than the normal 8-bit accuracy. Specific applications of 10-bit video are production of high-quality video and motion picture material, archiving, specialized visual effects, and efficient transmission and distribution. Although there was sufficient initial interest[29] in this part of MPEG-2, the work on this part had to be abandoned. This was partly due to insufficient availability of systems for capture, storage, processing, and display of 10-bit video. Another reason had to do with limitations in resources of organizations participating in MPEG-2 owing to various continuing work items in parallel. Moreover, some organizations felt that at least for the near term, applications of 10-bit video were too specialized and lacked a broad base in consumer markets.

This work item has now officially been removed from the MPEG-2 program of work.

2.2.9 MPEG-2 RTI

This part of MPEG-2, part 9,[11] is related to MPEG-2 Systems, part 1, and is primarily intended for applications where the MPEG-2 System may be used for real-time interchange of data such as in telecommunications applications. It specifies a Real-Time Interface (RTI) between Channel Adapters and MPEG-2 Systems Transport Stream decoders. The RTI makes it possible to specify network interconnects and to verify the performance of various products that include MPEG decoders.

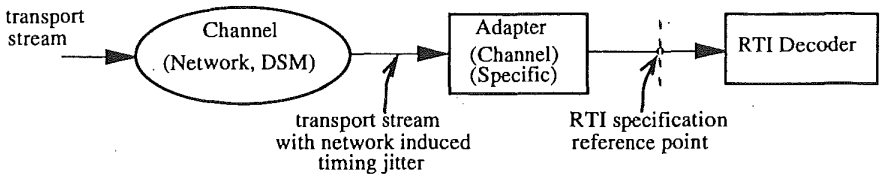


Fig. 2.10 Configuration for illustrating Real-Time Interface (RTI).

Such interconnects may be specified for networks such as LAN, ATM, and others by other organizations and standards bodies.

Thus with the specification of an RTI, it will be possible to build network adaptation layers that guarantee required timing performance and also guarantee that decoders will have correct behavior in terms of buffers and timing recovery. The RTI is an optional supplement to the System's Transport Stream System Target Decoder (STD) model. In Fig. 2.10 we show an example configuration for illustrating the real-time interface specification.

In this example configuration, a transport stream generated by an MPEG encoder passes over a channel such as a network and through a channel adapter on its way to an RTI decoder. The transport stream in going through the network is assumed to suffer from some timing jitter. Then the reference point at which the RTI specification really applies is between the channel adapter and the RTI decoder. The exact value of acceptable jitter depends on the application.

It is expected that RTI will achieve the final stage of International Standard by the end of the first quarter of 1996.

2.2.10 MPEG-2 DSM-CC Conformance

This part of MPEG-2, part 10, is a relatively new part and is related to part 6, called DSM-CC. Part 10 of MPEG-2 was originally aimed to be an exchange format for multimedia scripts; it was then referred to as Reference Scripting format (RSF) or more specifically the DSM-RSF, DSM-RSF was expected to be closely linked to the Multimedia and Hypermedia Experts Group (MHEG) standard which mainly deals with presentation aspects of coded audio-visual data. The MHEG standard, although it does not explicitly include a scripting language, does implicitly support such language.

With DSM-CC, it was envisaged[31] to be possible to code a script controlling presentation aspect of the audio-visual data along with the coded representation (MPEG bitstreams) of the audio-visual data. DSM-RSF was thus expected to play the role of a specialized multimedia scripting language. Although many such languages exist, it was felt that DSM-RSF was necessary to take advantage of special features of MPEG Audio and Video coding as well as that of DSM-CC.

The need for this part of the MPEG-2 standard arose during the DSM-CC work and thus this part of the standard was started late. During a recent (March '96) MPEG meeting, this part was renamed from DSM-RSF to DSM-CC conformance. Thus the original goal of this part has been revised and thus this part will mainly provide a means to verify if a specific DSM-CC implementation is conformant to part 6 MPEG-2 or not. Part 10 is expected to achieve the stage of Committee Draft by July 1996 and the final status of International Standard by March 1997.

2.3 SUMMARY

In this chapter we have presented a brief overview of the various parts of the MPEG-2 standard and shown the relationship of parts of MPEG-2 to parts of MPEG-1 where appropriate. As the reader may have guessed, from among the various parts of the MPEG-2 standard, this book primarily focuses on MPEG-2 Video. In the following chapters we will certainly be discussing coding algorithms, syntax, semantics, profiles, and implementation aspects of MPEG-2 Video. However, since we believe that any discussion about an MPEG standard would be incomplete without a working

knowledge of MPEG Audio and MPEG Systems, we include one chapter on MPEG-2 Audio and another on MPEG-2 Systems. We will then introduce example applications of MPEG-2 and discuss characteristics of transmission channels and storage media that may be used to transmit or store MPEG compressed audio and video.

We now summarize a few key points discussed in this chapter.

- Systems, Video, and Audio are three primary parts of the MPEG-1 and the MPEG-2 standards. Conformance and Software are two other parts of the MPEG-1 and MPEG-2 standards. The MPEG-2 standard has four other parts: Digital Storage Media-Control and Comand (DSM-CC), Non Backward Compatible Audio (NBC Audio), Real-Time Interface (RTI), and DSM-CC Conformance. Another part of MPEG-2 called 10-bit video has been dropped.
- MPEG-2 Systems comes in two forms. The first one, called the Program Stream, is MPEG-1 like and intended for error-free media. The second one, called the Transport Stream, is intended for noisier environments such as terrestrial and satellite channels.
- MPEG-2 Video is a syntactic superset of MPEG-1 Video. It is based on motion compensated DCT coding, B-pictures, and a Group-of-Pictures structure as in MPEG-1 Video. Furthermore, MPEG-2 Video supports tools for efficient coding of interlace. MPEG-2 Video also allows scalable video coding. Compatibility is a specific form of scalability that can allow interoperability with MPEG-1 Video. From an applications standpoint, MPEG-2 video is organized as profiles, where each profile contains a subset of all the coding tools of MPEG-2 Video.
- MPEG-2 Audio comes as two parts. The first is Backward Compatible (BC) with MPEG-1, and the second is Non Backward Compatible (NBC) with MPEG-1. MPEG-2 BC Audio allows coding of multichannel (such as five-channel) audio in a backward compatible manner so that from an MPEG-2 bitstream an MPEG-1 stereo decoder can reproduce a meaningful downmix of five channels to deliver correct sounding stereo. On the other hand, MPEG-2 NBC Audio achieves notably higher coding performance as it does not impose the restriction of compatibility with MPEG-1; this part of MPEG-2 was started late and is currently in progress.
- MPEG-2 Conformance specifies tests to verify compliance of bitstreams and decoders claiming to meet the requirements specified for MPEG-2 Systems, Video, and Audio. MPEG-2 Software, also referred to as Technical Report, is a C-programming implementation of the MPEG-2 Systems, Video, and Audio standards. The implementation includes a limited encoder and a full decoder for all three parts. This software is copyrighted by ISO.
- MPEG-2 DSM-CC specifies protocols to manage interaction of users with stored MPEG-1 and MPEG-2 bitstreams. A very basic type of interaction consists of actions of DSM in response to commands by a user; Annex A in MPEG-2 Systems supports such single user, single DSM applications. The DSM-CC, however, includes more extensive protocols to extend the basic idea of interaction to heterogeneous networked environments and interactive-video applications. Another part of MPEG-2 that is related to DSM is currently ongoing and is referred to as MPEG-2 DSM-CC Conformance. This part was originally intended to specify formats for Reference Scripts and was expected to be related to Multimedia and Hypermedia Experts Group Standard (MHEG) which deals with presentation aspects. However, it is now expected to deal mainly with compliance aspects of MPEG-2 DSM-CC.
- MPEG-2 RTI imposes some restrictions on MPEG-2 Systems and in particular Transport Streams. It specifies timing of the real-time delivery of bytes of the Transport Stream as a Real-Time Interface. This part does not override any of the requirements of MPEG-2 Systems, and compliance to this part of the standard is optional. One of the important parameters for RTI specification is the network induced timing jitter, which may depend on the application.

REFERENCES

1. "Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5

Mbit/s," *ISO/IEC 11172–1:Systems* (November 1991).

2. "Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s," *ISO/IEC 11172–2: Video* (November 1991).
3. "Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s," *ISO/IEC 11172–3: Audio* (November 1991).
4. "Generic Coding of Moving Pictures and Associated Audio Information: Systems," ISO/IEC 13818–1 : Draft International Standard (November 1994).
5. "Generic Coding of Moving Pictures and Associated Audio Information: Video," ISO/IEC 13818–2 : Draft International Standard (November 1994).
6. "Generic Coding of Moving Pictures and Associated Audio Information: Audio," ISO/IEC 13818–3 : International Standard (November 1994).
7. Conformance Editing Committee, "Generic Coding of Moving Pictures and Associated Audio: ISO/IEC 13818–4: Conformance," *ISO/IEC/JTC1/SC29/WG11 N0742* (July 1994).
8. "Generic Coding of Moving Pictures and Associated Audio Information: Software," ISO/IEC 13818–5: Draft Technical Report, March 1995.
9. S. J. Huang, "MPEG Digital Storage Media (DSM) Control Command," *Signal Proc. Image Commun.*, pp. 521–524, (February 1995).
10. M. S. Goldman, D. F. Hooper, and C. Adams, "ISO/IEC 13818–6 MPEG-2 Digital Storage Media Command & Control (DSM-CC) Tutorial," Tokyo (July 1995).
11. Systems Editing Committee, "Generic Coding of Moving Pictures and Associated Audio: Real-Time Interface Specification," *ISO/IEC/JTC1/SC29/WG11 N0809* (November 1994).
12. A. G. MacInnis, "The MPEG Systems Coding Specification," *Signal Proc. Image Commun.* pp. 153–159, (April 1992).
13. Video Simulation Model Editing Committee, "MPEG Video Simulation Model 3 (SM3)" (July 1990).
14. D. J. Le Gall, "MPEG: A Video Compression Standard for Multimedia Applications," *Commun. ACM* 34:47–58 (April 1991).
15. A. Puri, "Video Coding Using the MPEG-1 Compression Standard," *Society for Information Display: International Symposium,* pp. 123–126, Boston (May 1992).
16. D. Pan, "An Overview of the MPEG/Audio Compression Algorithm," *Proc. SPIE Digital Video Compression on Personal Computers: Algorithms and Technologies,*" 2187:260–273, San Jose (February 1994).
17. A. G. MacInnis, "MPEG-2 Systems," *Proc. SPIE Digital Video Compression on Personal Computers: Algorithms and Technologies,* 2187:274–278, San Jose (February 1994).
18. C. E. Holborow, "MPEG-2 Systems: A Standard Packet Multiplex Format for Cable Digital Services," *Proc. Society of Cable TV Engineers–Conference on Emerging Technologies,* Phoenix (January 1994).
19. Test Model Editing Committee, "Test Model 5," *ISO/IEC JTC1/S29/WG11/N0400* (April 1993).
20. D. J. Le Gall, "MPEG Video: The Second Phase of Work," *Society for Information Display: International Symposium,* pp. 113–116, Boston (May 1992).
21. A. Puri, "Video Coding Using the MPEG-2 Compression Standard," *Proc. SPIE Visual Communication and Image Processing,* Boston (November 1993).
22. M. M. Stojancic and C. Ngai, "Architecture and VLSI Implementation of the MPEG-2:MP@ML Video Decoding Process," *SMPTE* J. (February 1995).
23. D. Pan, "A Tutorial on MPEG/Audio Compression," *IEEE Multimedia,* pp. 60–74 (Summer 1995).
24. P. Noll, "Digital Audio Coding for Visual Communications," *Proceedings of the IEEE,* 83 (6) (June 1995).
25. L. Chiariglione, "Development of Multi-Industry Information Technology Standards: The MPEG Case," *Proceedings of the International Workshop on HDTV'93,* pp. 55–64 (October 1993).
26. Adhoc Group on NBC Syntax Development, "Report of Adhoc Group on NBC Syntax Development," *ISO/IEC/JTC1/SC29/WG11 MPEG96/0626* (January 1996).
27. A. N. Netravali and A. Lippman, "Digital Television: A Perspective," *Proceedings of the IEEE,* 83 (6) (June 1995).
28. L. Chiariglione, "The Development of an Integrated Audiovisual Coding Standard: MPEG," *Proceedings of the IEEE,* 83, (2) (February 1995).

29. A. T. Erdem and M. I. Sezan, "Compression of 10-bit Video Using Tools of MPEG-2," *Signal Proc. Image Commun.* pp. 27–56, (March 1995).

30. L. Chiariglione, "Report of the 30th Meeting," *ISO/IEC/JTC1/SC29/WG11 N0890* (March 1995).

31. C. Reader, "MPEG Update, Where It's at, Where It's Going," *19th International Broadcaster's Symposium,* pp. 117–127 (June 1995).

32. Video Subgroup, "Proposed Draft Amendment No. 3 to 13818–2 (Multi-view Profile)," *ISO/IEC JTC1/SC29/WG11 Doc. N1088* (November 1995).

33. Audio Subgroup, "NBC Time/Frequency Module Subjective Tests: Overall Results," *ISO/IEC/JTC1/SC29/WG11 N0973,* Tokyo (July 1995).

34. United States Advanced Television Systems Committee, "Digital Television Standard for HDTV Transmission," ATSC Standard (April 1995).

3

MPEG-2 Systems

Part 1 of ISO 13818 is called *Systems*. As we shall see, this term has many meanings and applies to many different contexts. In fact, Systems is the framework that allows MPEG-2 to be used in an extraordinary number of applications. For example, audio may be sent in several languages. Video may be sent at several resolutions and bitrates. Multiple views or 3D stereoscopic video may be sent that is completely compatible with ordinary MPEG-2 video.

3.1 MULTIPLEXING

The main function of MPEG-2 Systems is to provide a means of combining, or *multiplexing*, several types of multimedia information into one stream that can be either transmitted on a single communication channel or stored in one file of a DSM.* Since MPEG data are always byte aligned, the bitstreams are actually bytestreams, and we refer to them simply as *streams*.

There are two methods in wide use for multiplexing data from several sources into a single stream. One is called Time Division Multiplexing (TDM), which basically assigns periodic time slots to each of the elementary streams of audio, video, data, etc. H.320 uses TDM in its multiplexing standard, which is called H.221.

MPEG-1 Systems[1] and MPEG-2 Systems use the other common method of multiplexing, called *packet multiplexing*. With packet multiplexing, data packets from the several elementary streams of audio, video, data, etc. are interleaved† one after the other into a single *MPEG-2 stream*, as shown in Fig. 3.1. Elementary streams and MPEG-2 streams can be sent either at constant bitrate (CBR) or at variable bitrates (VBR) simply by varying the lengths or the frequency of the packets appropriately. In particular, elementary streams can be sent as VBR, even though the multiplexed MPEG-2 stream itself is transmitted as CBR. This enables some elementary streams that temporarily do not require very many bits to give up their proportionate channel capacity in favor of other elementary streams that temporarily require more than their share of the overall channel capacity. This functionality is called *statistical multiplexing*.

3.2 SYNCHRONIZATION

With multimedia information, there is a strong requirement to maintain proper synchronization

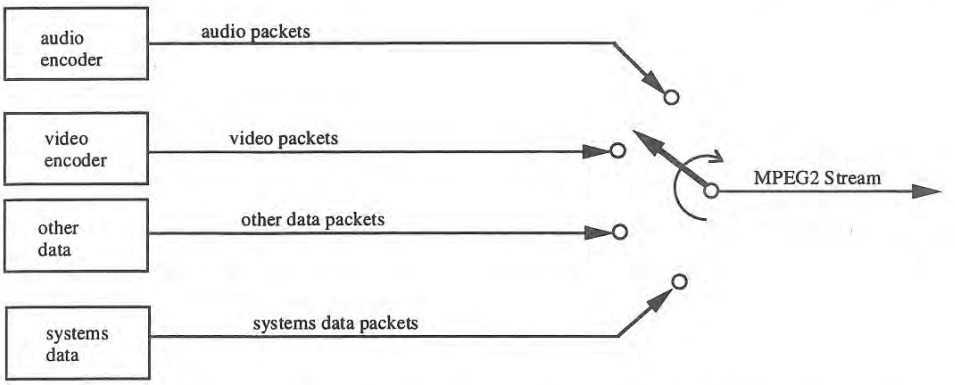


Fig. 3.1 Packet Multiplexing. At the multiplexer, packets from several elementary streams of audio, video, data, and so forth are multiplexed into a single MPEG-2 stream.

between the elementary streams when they are decoded and passed to their various output displays or transducer devices.

With TDM, the delay from encoder to decoder is usually fixed. However, in packetized systems the delay may vary owing both to the variation in packet lengths and frequencies during multiplexing. Moreover, if audio and video are prepared and edited separately prior to multiplexing, some means for restoring proper synchronization must be provided. For both MPEG-1 and MPEG-2, synchronization is achieved through the use of *Time Stamps* and *Clock References*. Time Stamps are 33-bit data fields indicating the appropriate time, according to a *Systems Time Clock* (STC) that a particular *Presentation Unit* (video pictures, audio frames, etc.) should be decoded by the decoder and presented to the output device. Clock References are 42-bit data fields indicating to the demultiplexer what the STC time should be when each clock reference is received.

There are two types of time stamps. A *Decoding Time Stamp* (DTS) indicates when the associated Presentation Unit* is to be decoded. A *Presentation Time Stamp* (PTS) indicates when the decoded Presentation Unit is to be passed to the output device for display. If the decoding and presentation time are the same† for a particular Presentation Unit, then either no time stamp is sent or only the PTS is sent. If no time stamp is sent, then the presentation (and decoding) time is extrapolated based on the presentation time for the previous Presentation Unit and the known sampling rate for that source.

Synchronization and decoding requires buffers at the decoder to temporarily store the compressed data until the correct time of decoding, as shown in Fig. 3.2. When the decoder STC advances to the time specified by the DTS, the corresponding Presentation Unit is removed from the buffer and passed to the decoder. If the PTS is later than the DTS, the decoded Presentation Unit waits inside the decoder until the STC advances to the PTS time.‡ Otherwise, the decoded Presentation Unit is presented to the display immediately. It is the responsibility of encoders and multiplexers to ensure that data are not lost due to buffer overflow at the receiver.

3.3 ELEMENTARY STREAMS

Elementary streams consist of compressed data from a single source (e.g., audio, video, data, etc.)

*MPEG defines *Access Units* for DTS timing. Here we use the term Presentation Unit to apply to both Presentation and Access Units since the difference is minor.

†PTS and DTS normally differ only in the case of video.

‡Practical MPEG receivers may add a delay to the DTS and PTS values for implementation purposes, in which case buffer sizes must be increased.

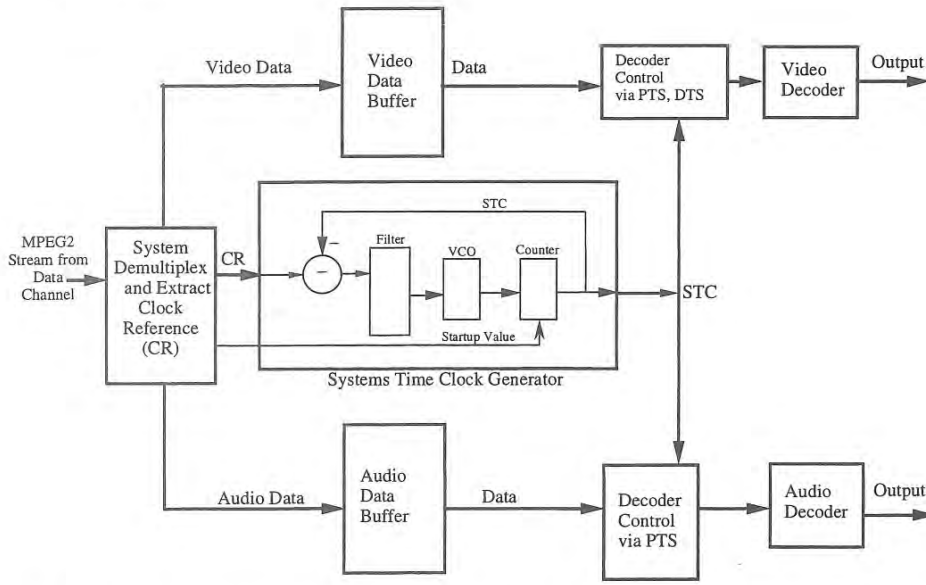


Fig. 3.2 MPEG receiver. A demultiplexer switch separates the incoming MPEG-2 stream into its component elementary streams. Compressed elementary stream Presentation Units are stored in buffers, where they wait until the proper time for decoding, as specified by a decoding time stamp (DTS) and the System Time Clock (STC). The STC generator is a simple phase locked loop (PLL) that tracks incoming *Clock References* (CRs). PLLs and CRs will be described later.

plus ancillary data needed for synchronization, identification, and characterization of the source information. The elementary streams themselves are first packetized into either constant-length or variable-length packets to form a *Packetized Elementary Stream* (PES). Each PES packet* consists of a header followed by stream data called the *payload.*

The PES header begins with one of 66 possible PES Start-Codes. MPEG-2 also has many other types of Start-Codes that are used for delineation of various other types of data. All MPEG-2 Start-Codes begin with a **start_code_prefix** (Psc), which is a string of 23 or more† binary 0s, followed by a binary 1, followed by an 8-bit *Start-Code ID* that is usually specified in hexadecimal.‡ Furthermore, the audio and video compressed data are designed to rarely if ever contain 23 consecutive zeros, thus easing random access to Start-Codes as well as speeding recovery after the occurrence of transmission errors.

In PES Start-Codes, the 8-bit Start-Code ID is called the **stream_id.** It ranges in value from 0xBD to 0xFE and serves to label the stream, as well as to specify the type of the stream, as shown in Table 3.1. Different stream_ids may be used to distinguish between separate streams of the same type.

Following the stream_id in the PES header are **PES_packet_length,** flags and indicators for various features and, finally, optional data whose presence is determined by the stream type as well as the values of the flags and indicators. The syntax of the PES header is shown in Fig. 3.3.

The semantics of the PES header variables§ are as follows:

*MPEG also defines other *specialized* streams for various purposes, as we shall see later. They are also packetized, but they are not usually called PESs.

†More that 23 zeros may be used in video streams, if desired, for byte alignment or padding.

‡Hexadecimal numbers are denoted by the prefix "0x." For example, 0xBC = binary "1011 1100." Numbers without prefixes or quotes are decimal.

§Flag and indicator variables are 1 bit, unless otherwise indicated.

start_code_prefix 23 or more binary 0s, followed by a 1. This is the same for all MPEG Start-Codes.

stream_id (8 bits) Hexadecimal values ranging from 0xBD to 0xFE. Defines ID type according to Table 3.1. May also be used as an ID number for different streams of the same type.

PES_packet_length (16 bits) The number of bytes that follow. For Transport Stream packets carrying video, a value of 0 indicates an unspecified length.

padding_bytes Fixed 8-bit values equal to 0xFF, which are discarded by the decoder.

marker_bits Fixed valued bits usually equal to all binary 1s.

Table 3.1 MPEG-2 Systems Start-Code IDs and stream_ids. Stream_ids range from 0xBD to 0xFE*. Program Streams (PS) will be described later. Entitlement Control Messages (ECM) and Entitlement Management Messages (EMM) are part of the Conditional Access, that is, scrambling control.

Start-Code ID's and stream_id's (hexadecimal)	Data or Stream Type
B9	PS End Code
BA	PS Pack Header
BB	PS System Header
BC	PS Program Map Table (PS-PMT)
BD	private_stream_1
BE	padding_stream
BF	private_stream_2
C0 to DF	MPEG1 or MPEG2 audio
E0 to EF	MPEG1 or MPEG2 video
F0	ECM private
F1	EMM private
F2	DSM Command & Control
F4	ITU-T type A (defined by ITU-T)
F5	ITU-T type B
F6	ITU-T type C
F7	ITU-T type D
F8	ITU-T type E
F9	ancillary_stream
FA to FE	reserved streams
FF	PS Directory

*MPEG's definition of elementary stream is somewhat looser than ours. MPEG includes values 0xBC and 0xFF as stream_ids, whereas we prefer to call them Start-Code IDs, since they never occur in PES packets.

PES_scrambling_control 2 bits, indicates the scrambling mode, if any.

PES_priority 1 indicates the priority of this PES packet. 1 indicates high priority.

data_alignment_indicator 1 indicates the payload begins with a video Start-Code or audio sync-word.

copyright 1 indicates the PES packet payload is copyrighted.

original_or_copy 1 indicates the PES packet payload is an original.

PTS_DTS_flags 2 bit field. Values of 2 and 3 indicate a PTS is present. 3 indicates a DTS is present. 0 means neither are present.

ESCR_flag 1 indicates a clock reference (called ESCR) is present in the PES packet header.

ES_rate_flag 1 indicates a bitrate value is present in the PES packet header.

DSM_trick_mode_flag 1 indicates the presence of an 8-bit trick mode field.

additional_copy_info_flag 1 indicates the presence of copyright information.

PES_CRC_flag 1 indicates a CRC† is present in the PES packet.

PES_extension_flag 1 indicates an extension of the PES packet header.

PES_header_data_length (8-bits) Specifies the total number of bytes following in the optional fields plus stuffing data, but not counting PES_packet_data. The presence or absence of optional fields is indicated by the above indicators.

PTS (presentation time stamp) Presentation times for presentation units.

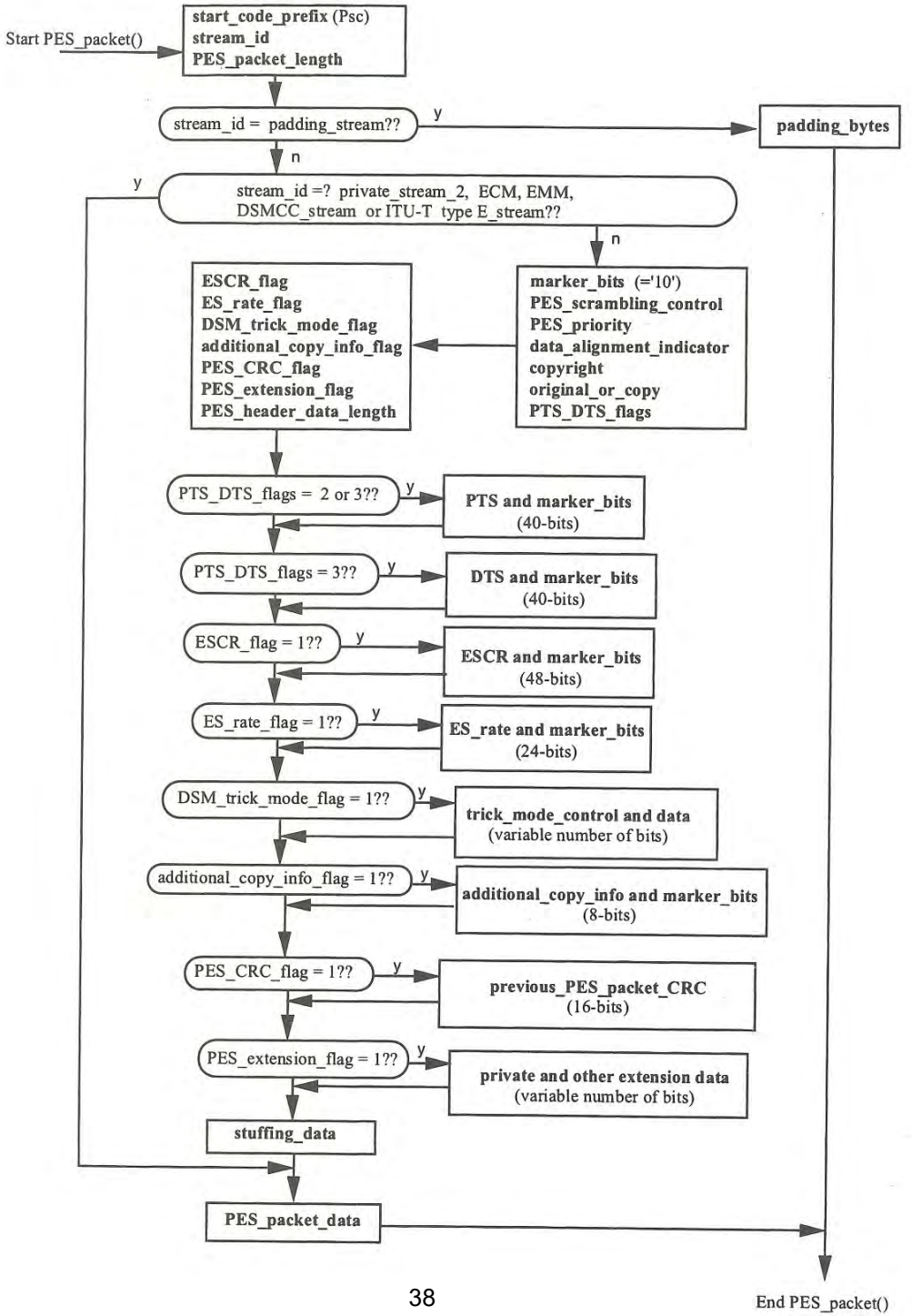
DTS (decoding time stamp) Decoding times for presentation units. PTS and DTS can differ only for video.

ESCR Elementary stream clock reference for PES Streams (see below).

ES_rate (elementary stream rate) The bitrate of a PES Stream. Normally, ESCR and ES_rate are used only if a PES is transmitted all by itself, in which case it is called a PES Stream.

trick_mode_control Indicates which trick mode (fast forward, rewind, pause, etc.) is applied to the video.

†Cyclic Redundancy Check, a parity check for error detection purposes.


Start PES_packet()
start_code_prefix (Psc)
stream_id
PES_packet_length
stream_id = padding_stream??
y
padding_bytes
n
y
stream_id =? private_stream_2, ECM, EMM,
DSMCC_stream or ITU-T type E_stream??
n
ESCR_flag
ES_rate_flag
DSM_trick_mode_flag
additional_copy_info_flag
PES_CRC_flag
PES_extension_flag
PES_header_data_length
marker_bits (='10')
PES_scrambling_control
PES_priority
data_alignment_indicator
copyright
original_or_copy
PTS_DTS_flags
PTS_DTS_flags = 2 or 3??
y
PTS and marker_bits
(40-bits)
PTS_DTS_flags = 3??
y
DTS and marker_bits
(40-bits)
ESCR_flag = 1??
y
ESCR and marker_bits
(48-bits)
ES_rate_flag = 1??
y
ES_rate and marker_bits
(24-bits)
DSM_trick_mode_flag = 1??
y
trick_mode_control and data
(variable number of bits)
additional_copy_info_flag = 1??
y
additional_copy_info and marker_bits
(8-bits)
PES_CRC_flag = 1??
y
previous_PES_packet_CRC
(16-bits)
PES_extension_flag = 1??
y
private and other extension data
(variable number of bits)
stuffing_data
PES_packet_data
End PES_packet()

additional_copy_info Contains private data for copyright information.

previous_PES_packet_CRC (16 bits) Contains the CRC of the previous PES packet, exclusive of the PES packet header.

extension_data May include buffer sizes, sequence counters, bitrates, etc.

stuffing data Fixed 8-bit values equal to 0xFF that may be inserted by the encoder. It is discarded by the decoder. No more than 32 stuffing bytes are allowed in one PES header.

PES_packet_data Contiguous bytes of data from the elementary stream.

3.4 PROGRAMS AND NONELEMENTARY STREAMS

PES packets from various elementary streams are combined to form a *Program*. A program has its own STC, and all elementary streams in that program are synchronized using PTSs and DTSs referenced to that STC.

MPEG has defined three nonelementary streams for transmitting or storing programs. The first is called a *Program Stream* and is aimed at applications having negligible transmission errors. The second is called a *Transport Stream* and is aimed at applications having nonnegligible transmission errors. The third has the strange appellation *PES Stream** and is useful for certain operations in specific implementations; however, it is not recommended for transmission or storage because of error resilience and random access limitations.

3.4.1 Program Streams

A Program Stream carries one *Program* and is made up of *Packs* of multiplexed data. A Pack consists of a pack header followed by a variable number of multiplexed PES packets from the various elementary streams plus other descriptive data to be described below. In a PS, each elementary stream must have its own unique stream_id.

A Pack Header, shown in Fig. 3.4, consists of a *Pack Start-Code*, followed by a 42-bit *Systems Clock Reference* (SCR), which is used to reset the decoder STC, followed by the assumed PS bitrate (called *program_mux_rate*), followed finally by optional stuffing bytes.

Program Streams (PSs) are suitable for Digital Storage Media (DSM) or transmission networks that have either negligible or correctable transmission errors.

It is the responsibility of the multiplexer to guarantee that Program Streams are decodable by all standard MPEG decoders. For this to be possible, MPEG specifies certain rules for multiplexers in the form of a *Program Stream–System Target Decoder* (PS-STD). The PS-STD is a hypothetical program decoder, shown in Fig. 3.5, that is synchronized exactly with the encoder. It consists of a demultiplexer, system control, STC, and elementary stream buffers and decoders.

Bytes from the multiplexed Program Stream are fed to the PS-STD at the rate specified in the Pack header.† The demultiplexer switch then routes the PES packets to the appropriate decoder buffers or Systems control as indicated by the Start-Code IDs and stream_ids.

The STC is a 42-bit counter incrementing at a rate of 27MHz.‡ Each received SCR specifies the desired value of the STC at the exact arrival time of that SCR at the PS-STD. However, in real systems the SCR may have a small error. Thus, most practical program receivers will use a Phase Locked Loop (PLL), as shown in Fig. 3.2, to control their STCs. PLLs will be covered later in this chapter.

Time Stamps are used for PS-STD control. A coded Presentation Unit is removed from its buffer and passed to its decoder at the time specified by its DTS, which is either included in the stream or obtained from the PTS. The decoded Presentation Unit is then passed to the output device at the time specified by its PTS, which is either included in the stream or extrapolated from a past PTS.

The multiplexer must construct the Program Stream so that PS-STD buffers never overflow.

*Some members of MPEG deny the existence of PES Streams.

†There may be gaps between packs, during which no data is transferred.

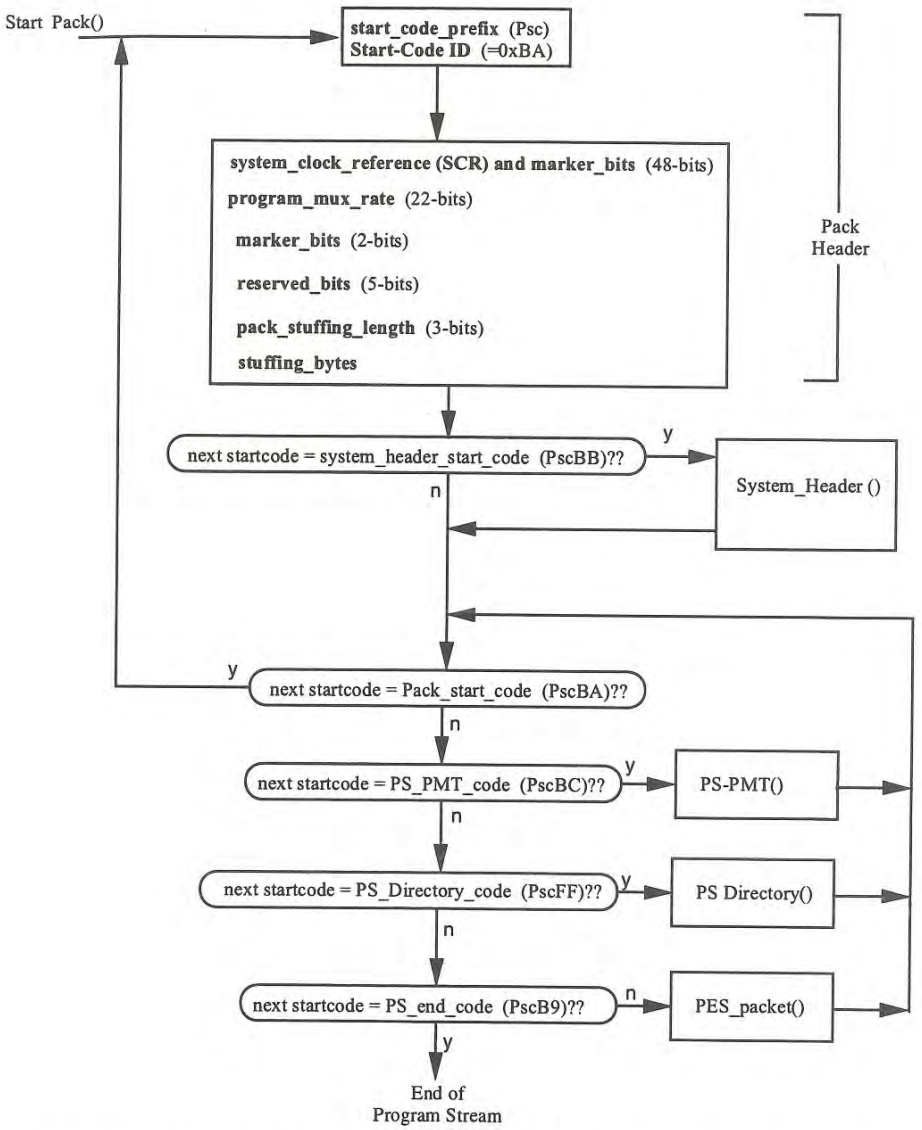


Fig. 3.4 Syntax of a Pack. A Program Stream (PS) is made up of Packs of multiplexed PES packets from the various elementary streams plus other descriptive data to be described later. Each pack begins with a pack header. The PS ends with Start-Code ID 0xB9.

Also, buffers are allowed to underflow only in certain prescribed situations. The most straightforward way to guarantee this is for the encoder/multiplexer to keep track of the timing and buffer fullness in each of the elementary stream decoders,

In addition to PES packets, PS Packs also contain other descriptive data called *PS Program Specific Information* (PS-PSI), which defines the program and its constituent parts. One specialized type of PS-PSI is called the *System Header,* which if present

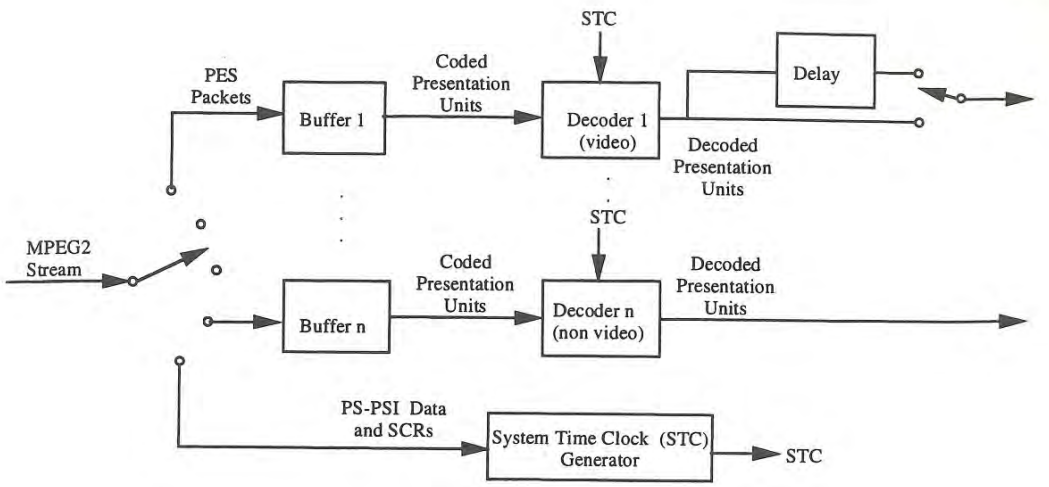


Fig. 3.5 Program Stream–System Target Decoder (PS-STD).

tem Header certain specifications are provided such as bitrate bound, buffer sizes for the various elementary streams, and so forth. The syntax of the PS System Header is shown in Fig. 3.6. Note that it is different than the PES format of Fig. 3.3.

The semantics of the PS System Header are as follows:

header_length (16 bits) The length in bytes of the following data.

rate_bound (22 bits) Upper bound on the program_mux_rate field.

audio_bound (6 bits) Upper bound on the number of active audio streams.

fixed_flag 1 indicates fixed bitrate for the MPEG-2 stream.

CSPS_flag 1 indicates constrained parameters (a set of limitations on bitrate and picture size).

system_audio_lock_flag Indicates the audio sampling is locked to the STC.

system_video_lock_flag Indicates the video sampling is locked to the STC.

video_bound (5 bits) Upper bound on the number of video streams.

packet_rate_restriction_flag Indicates packet rate is restricted.

stream_id Elementary stream ID for the following data.

P-STD_buffer_bound (14 bits) Upper bound on the PS-STD buffer size specified in the PES packet **extension_data.** A practical decoder buffer size should exceed that specified by *P-STD_buffer_bound.*

Another specialized type of PS-PSI is called the *PS Program Map Table** (PS-PMT), shown in Fig. 3.7, which contains information about the program and its constituent elementary streams. The semantics for the PS-PMT are as follows:

program_map_table_length (16 bits) The number of bytes that follow.

current_next_indicator 1 indicates this PS-PMT is currently valid. (More correctly, it will be valid as soon as the final byte is received.) 0 indicates that it is not yet valid, but is about to be valid and is being sent for preparatory purposes only.

program_map_table_version (5 bits) Version number, which increments with each change.

program_stream_info_length (16 bits) Number of bytes in the program descriptors (to be described later).

program_descriptors( ) Descriptive information for this program (to be discussed later).

*MPEG uses the term "Program Stream Map." We use "PS Program Map Table" for clarity later in the chapter.

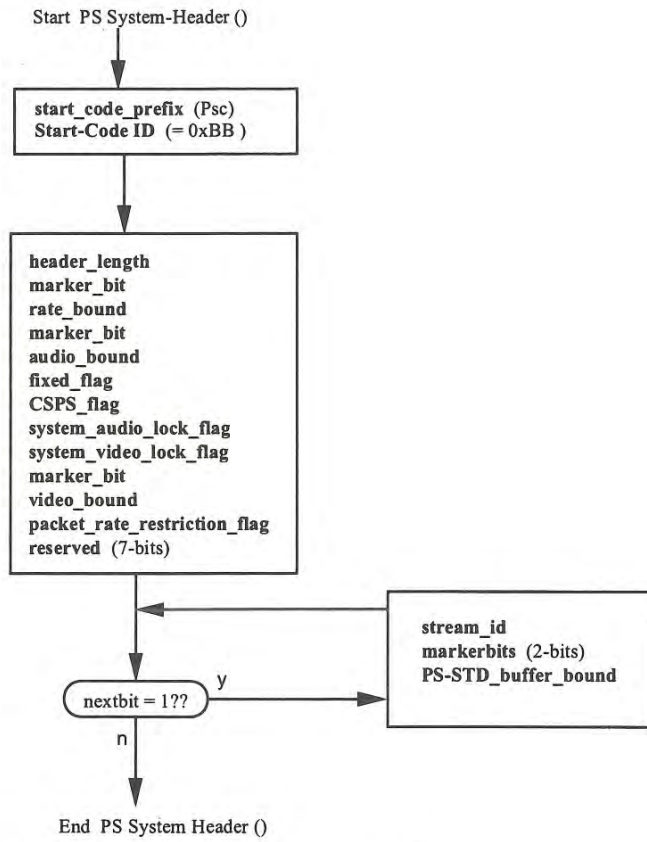


Fig. 3.6 The PS System Header provides parameters needed for the PS-STD.

elementary_stream_map_length (16 bits) Total number of bytes in all of the elementary stream (ES) data that follows, not including CRC.

stream_type (8 bits) Type of the elementary stream according to Table 3.2.

elementary stream_id (8 bits) Each PS Elementary Stream requires a unique stream_id (see Table 3.1).

elementary_stream_info_length (16 bits) Number of bytes in the ES descriptors for this ES.

ES_descriptors( ) Descriptive information for this ES (to be discussed later).

CRC_32 (32 bits) CRC for the entire program stream map.

Another specialized type of PS-PSI is called the stored stream. Usually, it lists selected video I-pictures, their locations and sizes, plus time stamps. However, it can also be used for audio.

3.4.2 Transport Streams

Transport Streams (TSs) are defined for transmission networks that may suffer from occasional transmission errors. They are similar to Program Streams in that they consist of multiplexed data from Packetized Elementary Streams (PESs) plus other descriptive data. However, one more layer of packetization is added in the case of TS. In general, relatively long, variable-length PES packets are further packetized into shorter TS packets of

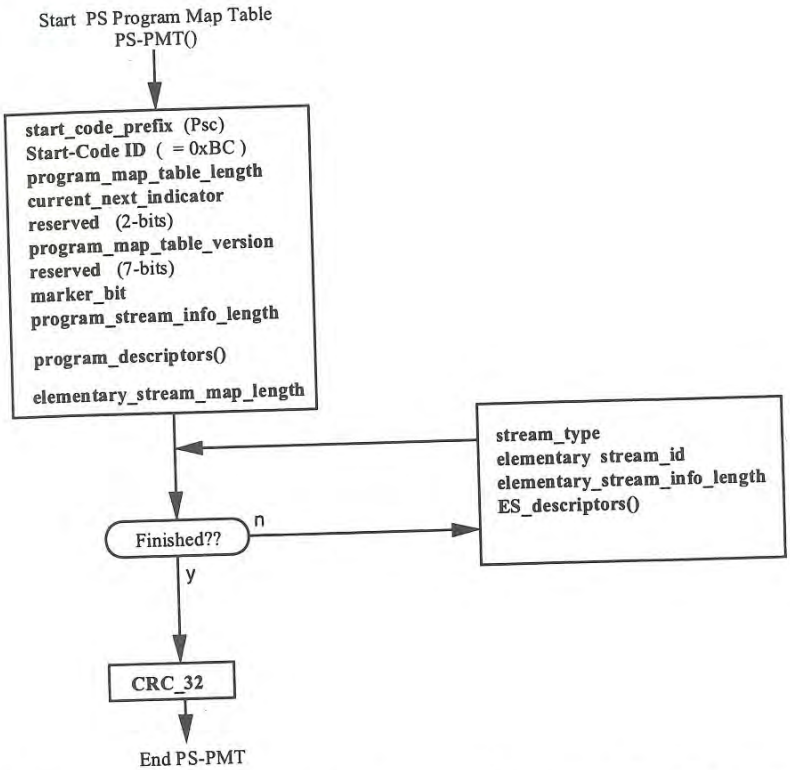


Fig. 3.7 The PS Program Map Table (PS-PMT) describes the program and its constituent elementary streams.

Table 3.2 Elementary stream types.

stream_type	Description
0x00	Reserved
0x01	MPEG1 Video
0x02	MPEG2 video
0x03	MPEG1 Audio
0x04	MPEG2 Audio
0x05	private_sections (see Fig. 3.11)
0x06	PES packets containing a private stream (see Fig. 3.3)
0x07	MHEG
0x08	DSM CC
0x09	ITU-T Rec. H.222.1
0x0A	ISO/IEC 13818–6 type A
0x0B	ISO/IEC 13818–6 type B
0x0C	ISO/IEC 13818–6 type C
0x0D	ISO/IEC 13818–6 type D

tion between TS and PS is that TS can carry several programs, each with its own STC. Program Specific Information (TS-PSI) is also carried differently in TSs than in PSs.

Each TS packet consists of a TS Header, followed optionally by ancillary data called the *Adaptation Field,* followed typically by some or all of the data from one PES packet.* The TS Header consists of a sync byte, flags and indicators, Packet Identifier (PID), plus other information for error detection, timing, etc., as shown in Fig. 3.8. PIDs are used to distinguish between elementary streams. Thus, in a Transport Stream, elementary streams need not have unique **stream_id** values.

The sematics for the TS packet are as follows:

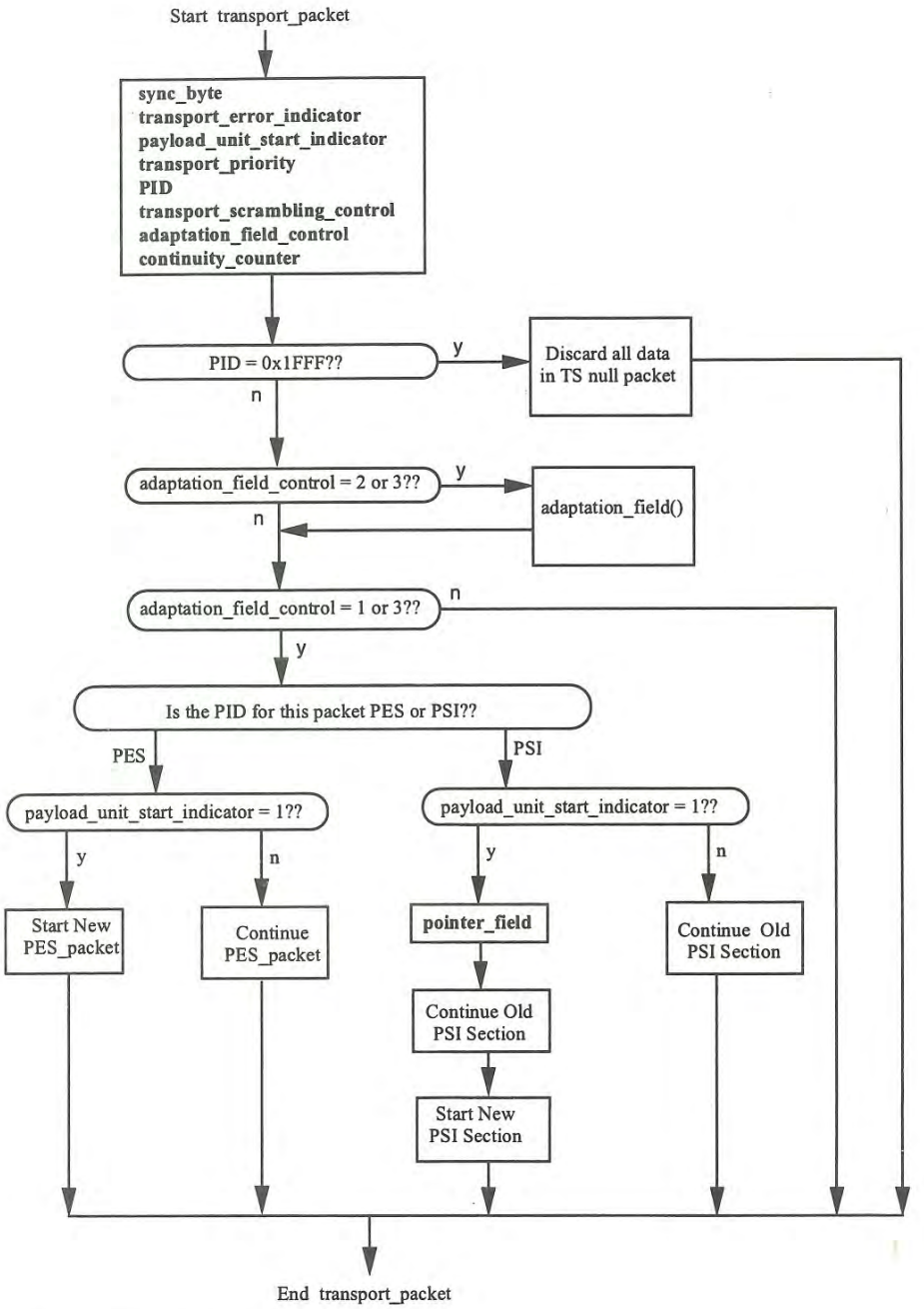


Fig. 3.8 Transport Stream (TS) Packet Syntax.

payload_unit_start_indicator Indicates the presence of a new PES packet or a new TS-PSI Section.

transport_priority 1 Indicates a higher priority than other packets.

PID 13-bit packet IDs. Values 0 and 1 are preassigned, while values 2 to 15 are reserved. Values 0x0010 to 0x1FFE may be assigned by the Program Specific Information (PSI), which will be described later. Value 0x1FFF is for null packets.

transport_scrambling_control (2 bits) Indicates the scrambling mode of the packet payload.

adaptation_field_control (2 bits) Indicates the presence of an adaptation field or payload.

continuity_counter (4 bits) One continuity_counter per PID. It increments with each nonrepeated Transport Stream packet having the corresponding PID.

pointer_field (8 bits) Used only in TS-PSI packets that contain a new TS-PSI Section (to be described later). Indicates the number of bytes until the start of the new TS-PSI Section.

The Adaptation Field contains flags and indicators, STC timing information in the form of a *Program Clock Reference* (PCR), which is used in exactly the same way as the SCR of the Program Stream, plus other data as shown in Fig. 3.9.

Semantics of the adaptation field are as follows:

adaptation_field_length (8 bits) The number of bytes following.

discontinuity_indicator 1 indicates a discontinuity in the clock reference or the continuity counter or both.

random_access_indicator 1 indicates the next PES packet is the start of a video sequence or an audio frame.

elementary_stream_priority_indicator 1 indicates a higher priority.

PCR_flag 1 indicates the presence of a PCR.

OPCR_flag 1 indicates the presence of an original PCR.

splicing_point_flag Indicates that a splice_countdown is present.

transport_private_data_flag Indicates the presence of private_data bytes.

adaptation_field_extension_flag Indicates the presence of an adaptation field extension.

program_clock_reference (PCR) defined above.

original_program_clock_reference (OPCR) This PCR may be used when extracting an original single program from a multiprogram TS (use with caution).

splice_countdown (8 bits) Specifies the remaining number of Transport Stream packets, of the same PID, until a splicing point is reached. Splice points mark the end of an audio frame or a video picture.

transport_private_data_length (8 bits) The number of private_data bytes immediately following.

private_data_bytes Private data.

adaptation_field_extension_length (8 bits) The number of bytes of the extended adaptation field.

stuffing_bytes Fixed 8-bit values equal to 0xFF that may be inserted by the encoder. They are discarded by the decoder.

Within a TS, PIDs are used to distinguish between different streams and different PSI. Different streams may belong to different programs or the same program. In any event, they all have different PIDs.

The program descriptions plus the assignments of PESs and PIDs to programs are contained in specialized TS streams called *TS-Program Specific Information* (TS-PSI). The specialized TS-PSI streams are shown in Table 3.3.

For convenience as well as length limitations, some of the TS-PSI may be sent in *Sections*. If a TS packet contains the start of any TS-PSI Section, then the TS packet data starts with an 8-bit **pointer_field** that tells how many bytes are contained in a partial section at the beginning of the TS packet data. If there is no such partial section, then **pointer_field** = 0.

If the TS-PSI is spread over many TS packets and is sent periodically to assist random tuning to a broadcast TS, then the pointer_field enables rapid access to the very next TS-PSI Section. Again for convenience, after the end of any TS-PSI Section, stuffing bytes = 0xFF may be inserted until the end of the TS packet.

The first TS-PSI stream is called the *Program Association Table* (TS-PAT). The TS-PAT, shown in Fig. 3.10, is always carried in TS packets having PID = 0, which are typically the first TS packets read by a receiver. For each program in the TS, the TS-PAT

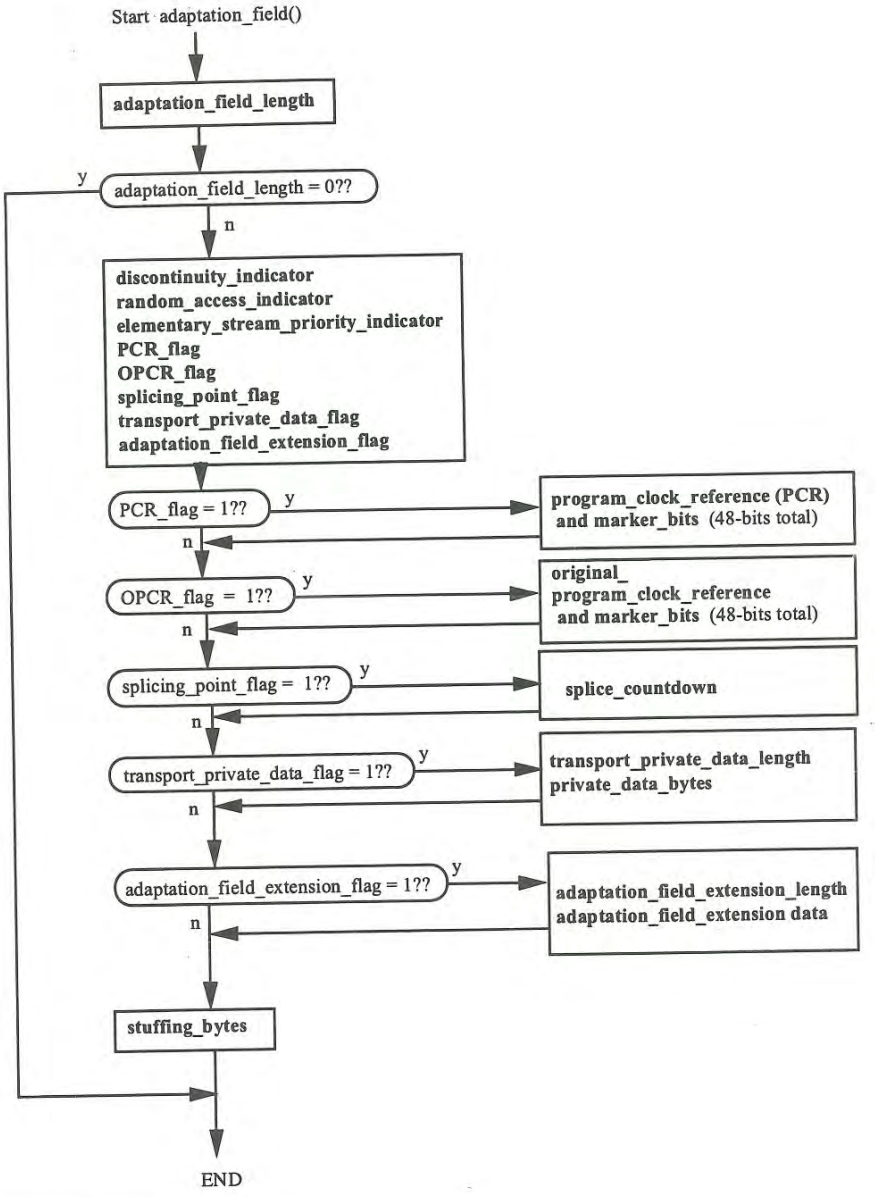


Fig. 3.9 Transport Stream Adaptation Field.

defines a unique nonzero **program_number** and an associated **TS_PMT_PID*** to be used later by packets that carry the Program Map Table (TS-PMT) for that program. Several (or in fact all) programs may use the same TS_PMT_PID for carrying their TS_PMTs, or each program could use a differ-

Table 3.3 Transport Stream–Program Specific Information (TS-PSI) is carried in specialized TS-PSI streams. Private data may be carried in Private_sections.

TS-PSI Type	PID Value (13-bits)	table_id (8-bits)	Function
Program Association Table (TS-PAT)	0x0000	0x00	Assigns Program Numbers and Program Map Table PID's
Network Information Table (TS-NIT)	Assigned in TS-PAT	0x40 to 0xFE	Specifies physical network parameters
Program Map Table (TS-PMT)	Assigned in TS-PAT	0x02	Specifies PID values for program components
Conditional Access Table (TS-CAT)	0x0001	0x01	Carries information and data for scrambling
Private_section	Assigned in TS-PAT	0x40 to 0xFE	Defined by application
Defined by DSM-CC	Assigned in DSM-CC	0x38 to 0x3F	Defined by DSM-CC
TS Description	0x0003	0x03	Optional TS Description (under study)

ent TS_PMT_PID. Programs may be added or deleted at any time* during the TS by retransmitting the entire TS-PAT with the version number increased† by 1. If there are many programs, the TS-PAT may be sent in several sections, with each section typically defining one or more programs.

The semantics of the TS-PAT are as follows:

table_id (8 bits) Set to 0x00

section_syntax_indicator Set to 1.

section_length (12 bits) The number of bytes (≤1021) below in this section.

transport_stream_id (16 bits) ID defined by the user.

version_number (5 bits) Version number of the whole Program Association Table, that is, every section of a given TS-PAT has the same version_number. A new TS-PAT with an incremented version_number becomes *valid* when the last section is received.

current_next_indicator 1 indicates that this Program Association Table is currently valid. (More correctly, it will be valid as soon as the final section is received.) 0 indicates that it is not yet valid, but is about to be valid and is being sent for preparatory purposes only.

section_number (8 bits) The number of this section. It is incremented by 1 for each additional section.

last_section_number (8 bits) The number of the final section of the complete Program Association Table.

program_number (16 bits) The program to which the following TS_PMT_PID is applicable. Program_number 0x0000 refers to the network PID.

TS_PMT_PID (13 bits) The PID of packets that carry the Program Map Table (TS-PMT) for the above program_number. This PID may be used to carry TS-PMT's for several (or all) programs. It may also carry private data with the format of Fig. 3.11.

network_PID (13 bits) The PID of the Network Information Table.

CRC_32 (32 bits) CRC for this TS-PAT section.

stuffing_bytes Fixed 8-bit values equal to 0xFF that may be inserted by the encoder. They are discarded by the decoder.

The second TS-PSI stream, which is optional, is called the *Network Information Table* (TS-NIT). Its TS packets use the **network_PID** defined in the TS-PAT. The Network Information Table carries network-specific data describing characteristics and features of the transmission network carrying the TS. Such data are very dependent on network implementation and are therefore considered *Private.* The TS-NIT is carried using the *Private_section* data structure shown in Fig. 3.11.

Other private data may also be carried with the Private_section structure by indicating a

*A programs **TS_PMT_PID** is not allowed to change during the program.

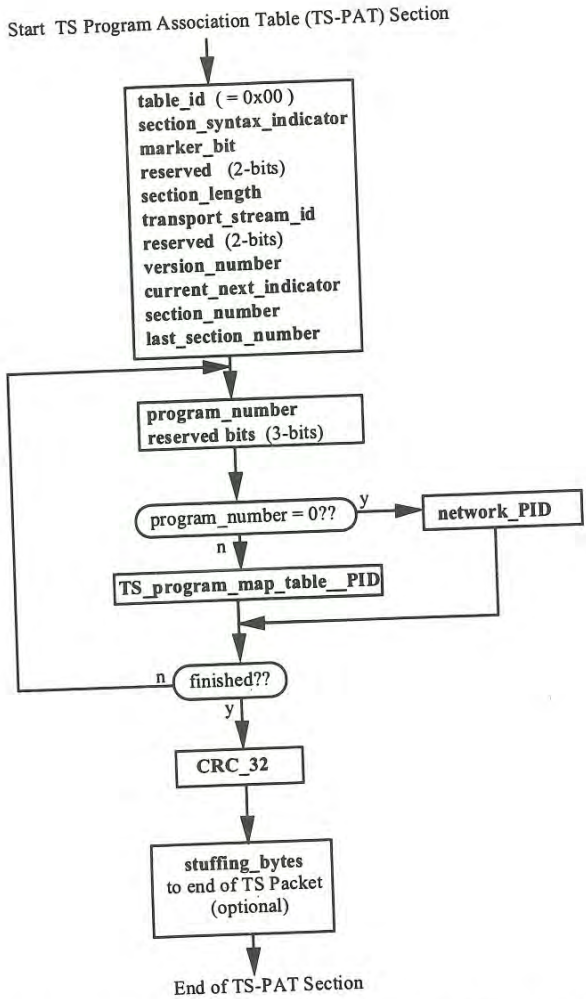


Fig. 3.10 One section of the TS Program Association Table (TS-PAT). The entire TS-PAT consists of one or more such sections. PID = 0x00 is always assigned to the TS-PAT.

stream_type = 0x05 (see Table 3.2) or by private definition of table_id values and then using a PID assigned for TS-PSI Sections.

The semantics of the TS Private_section are as follows:

table_id Values from 0x40 to 0xFE. Chosen by the user.

section_syntax_indicator 1 indicates that this private section follows the generic section syntax.

private_indicator User-definable flag.

private_section_length (12 bits) The number of remaining bytes in this section (≤4093).

private_data_byte User-definable.

table_id_extension (16 bits) Defined by the user.

version_number (5 bits) Version number only for this section and table_id. A new version with an incremented version_number becomes *valid* when the last byte of the section is received.

current_next_indicator 1 indicates that this section and table_id is currently valid. (More correctly, it will be valid as soon as the final byte is received.)

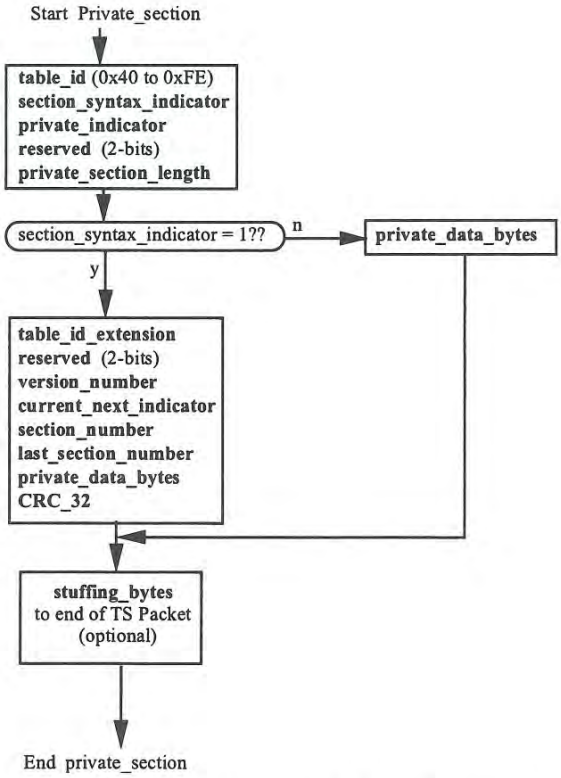


Fig. 3.11 A Transport Stream Private_section. It is used for carrying the Network Information Table (TS-NIT). It may also be used to carry other implementation data, network parameters or information outside the MPEG standard.

0 indicates that it is not yet valid, but is about to be valid and is being sent for preparatory purposes only.

section_number Same as TS-PAT.

last_section_number Same as TS-PAT.

CRC_32 Same as TS-PAT.

stuffing_bytes Fixed 8-bit values equal to 0xFF that may be inserted by the encoder. They are discarded by the decoder.

The third TS-PSI stream is comprised of *Program MAP Tables* (TS-PMT). Each TS-PMT, shown in Fig. 3.12, carries the defining information for one program as indicated by its *program_number.* This includes assignment of unique PID values* called **elementary_PIDs** to each packetized elementary stream (PES) or other type of stream in the program along with the **stream_type** of each stream. It also carries descriptive and relational information about the program and the elementary streams therein. This information is carried in data structures called *Descriptors.* More will be said about Descriptors later.

The semantics of the TS-PMT are as follows:

table_id (8 bits) Set to 0x02

section_syntax_indicator Set to 1.

section_length (12 bits) The number of bytes (≤1021) following in this section.

program_number (16 bits) Same as TS-PAT.

version_number (5 bits) Version number only for this TS-PMT Section and for this program. A new version with an incremented version_number

*User-defined PIDs are restricted to the range 0x0010 to 0x1FFE.

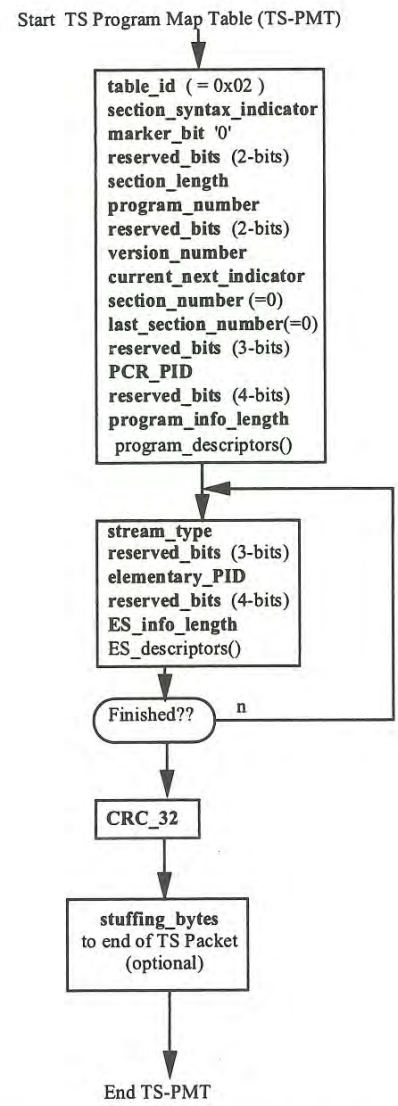


Fig. 3.12 Program Map Table (TS-PMT). Each program has a unique program_number and a corresponding TS-PMT that defines the constituent parts of the program. TS-PMTs are carried in TS packets having TS_PMT_PID identifiers defined in the TS-PAT.

becomes *valid* when the last byte of the new TS-PMT Section is received.

current_next_indicator 1 indicates that this section is currently valid. (More correctly, it will be valid as soon as the final byte is received.) 0 indicates that it is not yet valid, but is about to be valid and is being sent for preparatory purposes only.

section_number (8 bits) Set to 0x00. The entire TS-PMT for a program must fit in one PSI Section.

last_section_number (8 bits) Set to 0x00.

PCR_PID (13 bits) The PID that contains the PCR for this program.

program_info_length (12 bits) The number of bytes of the program_descriptors( ) immediately following.

program_descriptors( ) Descriptive information for this program (to be discussed later).

stream_type (8 bits) Elementary stream (ES) type as specified in Table 3.2.

elementary_PID (13 bits) The PID that will carry this ES.

ES_info_length (12 bits) The number of bytes in the ES_descriptors( ) for this ES.

ES_descriptors( ) Descriptive information for this ES (to be discussed later).

CRC_32 Same as TS-PAT.

stuffing_bytes Same as TS-PAT.

TS-PMTs for programs that use the same TS_PMT_PID value may be concatenated* and sent in TS packets using that TS_PMT_PID. Also, private data may be sent in any TS_PMT_PID packets using the Private_section data structure of Fig. 3.14. Any program definition may be changed at any time during the TS by simply retransmitting its changed TS-PMT with its version number incremented by 1.

The fourth TS-PSI stream is called the *Conditional Access Table* (TS-CAT). The TS-CAT, shown in Fig. 3.13, is always carried in TS packets having PID = 1. The TS-CAT, which may be sent in Sections, carries entitlement and management information to limit access to authorized recipients. This data may be changed at any time during the TS by retransmitting the entire TS-CAT with the version number increased by 1.

The semantics of the TS-CAT are as follows:

table_id (8 bits) Set to 0x01.

section_syntax_indicator Set to 1.

section_length (12 bits) The number of bytes (≤1021) of this section that follow.

version_number (5 bits) Version number of the whole Conditional Access Table, that is, every section of a given TS-CAT has the same version_number. A new TS-CAT with an incremented version_number becomes *valid* when the last section is received.

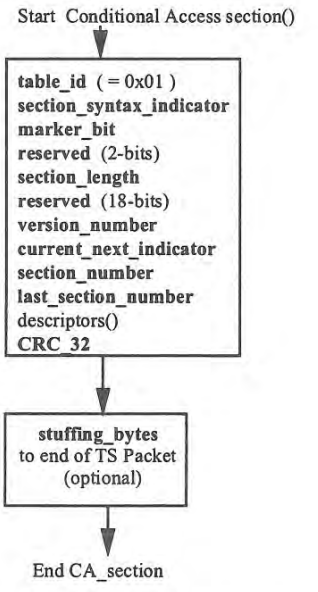


Fig. 3.13 The Transport Stream Conditional Access Table (TS-CAT) is carried in TS packets having PID = 1. It provides information on scrambling of the multimedia data.

current_next_indicator 1 indicates that this Conditional Access Table is currently valid. (More correctly, it will be valid as soon as the final section is received.) 0 indicates that it is not yet valid, but is about to be valid and is being sent for preparatory purposes only.

section_number Same as TS-PAT.

last_section_number Same as TS-PAT.

CA_descriptors To be discussed later.

CRC_32 Same as TS-PAT.

stuffing_bytes Same as TS-PAT.

As with the Program Stream, a Transport Stream–System Target Decoder (TS-STD) is defined to enable multiplexers to create data streams that are decodable by all receivers. Also as in the the PS-STD, the TS-STD inputs data at a piecewise constant rate† between PCRs. However, the TS-STD attempts to take into account several

*TS-PMTs are considered unnumbered PSI Sections.

†However, unlike the PS, this rate is unspecified in the TS itself.

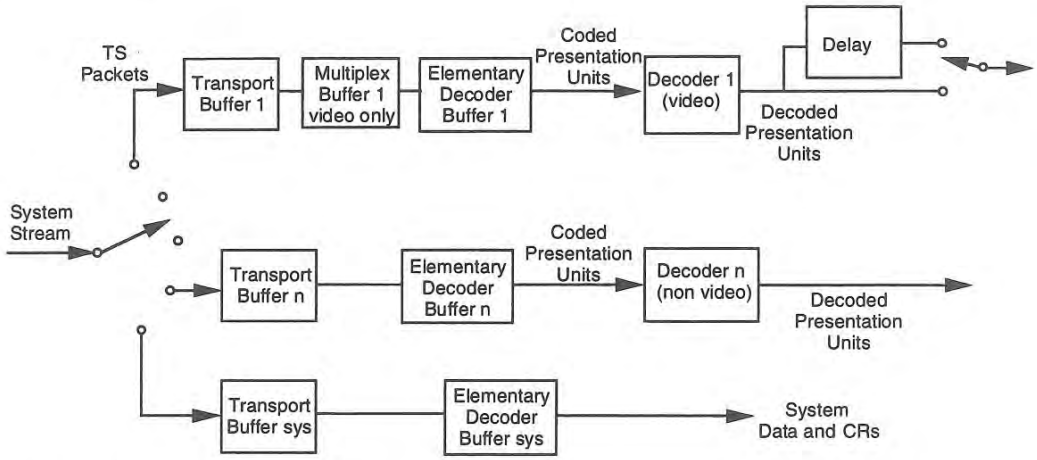


Fig. 3.14 Transport Stream–System Target Decoder (TS-STD). Three buffer types are defined: Transport Buffers (TB), Multiplex Buffers (MB), and Elementary Stream Buffers (EB). Only video uses MBs.

transmission impairments ignored by the PS-STD, as shown by Fig. 3.14.

The main difference is the definition of three types of buffers. The first type is the *Transport Buffer* (TB) of size 512 bytes. It inputs data at the full rate of the TS, but outputs data (if present) at a rate Rx that depends on data type and specification.

The second buffer type, used only for video, is the *Multiplexing Buffer* (MB), which is provided to alleviate the effects of TS packet multiplexing. Its output rate is specified in one of two ways. With the *Leak Method* the MB outputs data (if present and if EB is not full) at a rate Rbx that depends on the data type and specification. With the *Vbv_delay Method* the output rate is piecewise constant for each picture and is specified by the encoder using parameters* in the video bitstream.

The third buffer is the normal elementary stream decoder buffer (EB), whose size is fixed for audio and systems and specified in the video stream for video (see vbv_buffer_size). Several TS-STD transfer rates and buffer sizes of interest are shown for Main Profile Video, Audio, and Systems in Tables 3.4 and 3.5.

3.4.3 Descriptors

As we have seen, the PS and TS Program Map Tables carry descriptive and relational information about the program and the PESs therein. This information is carried in data structures called *Descriptors.*

For example, some programs may consist of multiple audios, in which several languages may be present in a movie. Also, several videos may sometimes be desired, such as several views of a sports contest.

Program_descriptors apply to programs, and ES_descriptors apply to elementary streams. Initially, a total of 256 possible descriptors are provided for, as shown in Table 3.6a. A few descriptors of interest are shown in Tables 3.6b to 3.6g.

In certain implementations the ITU-T timing descriptor allows for much more rapid receiver clock acquisition than can be achieved by using Clock References alone.

The semantics of the ITU-T timing descriptor are as follows:

SC_PESPktR (24 bits) If ≠ 0xFFFFFF, this parameter indicates that PES packets are generated by

Table 3.4 Transport Stream–System Target Decoder (TS-STD) transfer rates for Main Profile Video,* Audio, and Systems streams.

Elementary Stream	TB Rate (Rx)	MB Rate (Rbx)
Low Level Video	4.8 Mbits/s	4 Mbits/s
Main Level Video	18 Mbits/s	15 Mbits/s
High 1440 Level Video	72 Mbits/s	MIN[1.05xbit_rate, 60Mbs]
High Level Video	96 Mbits/s	MIN[1.05xbit_rate, 80Mbs]
Audio	2 Mbits/s	n.a.
Systems data	1 Mbits/s	n.a.

*The parameters bit_rate and vbv_buffer_size are given in the video stream.

Table 3.5 Transport Stream–System Target Decoder (TS-STD) buffer sizes for Main Profile Video, Audio and Systems streams.

Elementary Stream	TB Size	MB Size	EB Size
Low Level Video	4096 bits	496 469 bits − EB size	vbv_buffer_size
Main Level Video	4096 bits	1 915 008 bits − EB size	vbv_buffer_size
High 1440 Level Video	4096 bits	320 000 bits	vbv_buffer_size
High Level Video	4096 bits	426 667 bits	vbv_buffer_size
Audio	4096 bits	n.a.	28 672 bits
Systems data	4096 bits	n.a.	12 288 bits

Table 3.6a A total of 256 descriptors are provided. The ITU-T has defined additional descriptors using descriptor_tags 64 to 69.

descriptor_tag (8-bits)	Descriptor Function
0 and 1	Reserved
2	video_descriptor
3	audio_descriptor
4	hierarchy_descriptor
5	registration_descriptor
6	data_alignment_descriptor
7	target_background_grid_descriptor
8	video_window_descriptor
9	CA_descriptor
10	ISO_639_language_descriptor
11	system_clock_descriptor
12	multiplex_buffer_utilization_descriptor
13	copyright_descriptor
14	maximum bitrate descriptor
15	private data indicator descriptor
16	smoothing buffer descriptor
17	STD_descriptor
18	IBP Picture descriptor
19 to 22	DSM − CC
23 to 63	Reserved
64 to 255	User Private

Table 3.6b Video descriptor syntax and semantics.

video_descriptor( ) Syntax	Semantics
descriptor_tag (8-bits)	= 2
descriptor_length (8-bits)	number of bytes following
multiple_frame_rate_flag	0 indicates a single frame-rate
frame_rate_code (4-bits)	same as video stream
MPEG_1_only_flag	1 indicates only MPEG1
constrained_parameter_flag	1 indicates constrained parameters
still_picture_flag	1 indicates only still pictures
MPEG_1_only_flag = 1?? if yes, quit	
profile_and_level (8-bits)	same as video stream
chroma_format (2-bits)	same as video stream
frame_rate_extension_flag	1 indicates frame-rate extensions
reserved bits	

Table 3.6c Audio descriptor syntax and semantics.

audio_descriptor( ) Syntax	Semantics
descriptor_tag (8-bits)	= 3
descriptor_length (8-bits)	number of bytes following
free_format_flag	same as audio stream
ID	same as audio stream
layer (2-bits)	same as audio stream
variable_rate_audio_indicator	1 indicates bit-rate may change
reserved (3-bits)	

Table 3.6d Hierarchy descriptor syntax and semantics.

hierarchy_descriptor( ) Syntax	Semantics
descriptor_tag (8-bits)	=4
descriptor_length (8-bits)	number of bytes following
reserved (4-bits)	
hierarchy_type (4-bits)	1 to 4 indicates video scalabilities 5 indicates audio external stream 6 indicates private stream 15 indicates base layer
reserved (2-bits)	
hierarchy_layer_index (6-bits)	unique id for this layer
reserved (2-bits)	
hierarchy_embedded_layer_index (6-bits)	id for the next lower layer
reserved (2-bits)	
hierarchy_channel (6-bits)	transmission channel (optional)

Table 3.6e Conditional Access descriptor syntax and semantics.

CA_descriptor( ) Syntax	Semantics
descriptor_tag (8-bits)	= 9
descriptor_length (8-bits)	number of bytes following
CA_system_ID (16-bits)	type of CA system
reserved (3-bits)	
CA_PID (13-bits)	PID to which CA applies
private_data_bytes	

the encoder at a constant rate. SC_PESPktR is the ratio of 27MHz to the PES packet rate.

SC_TESPktR (24 bits) If ≠ 0xFFFFFF, this parameter indicates that TS packets for the specified e-

Table 3.6f IPB descriptor syntax and semantics.

IPB_Picture_descriptor( ) Syntax	Semantics
descriptor_tag (8-bits)	= 18
descriptor_length (8-bits)	number of bytes following
closed_gop_flag	1 indicates GOP before each I frame, and no prediction outside the GOP.
identical_gop_flag	1 indicates GOPs 2,3,4 . . . are all the same size
max_gop-length (14-bits)	maximum number of pictures in a GOP

Table 3.6g ITU-T Timing descriptor syntax.

ITU-T_timing_descriptor( ) Syntax
descriptor_tag (8-bits) (=69)
descriptor_length (8-bits)
SC_PESPktR
SC_TESPktR
SC_TSPktR
SC_byte_rate
vbv_delay_flag
reserved (33-bits)

mentary stream are generated by the encoder at a constant rate. SC_TESPktR is the ratio of 27MHz to the TES packet rate.

SC_TSPktR (24 bits) If ≠ 0xFFFFFF, this parameter indicates a constant Transport Stream packet rate. SC_TSPktR is the ratio of 27MHz to the TS packet rate.

SC_byterate (30 bits) If ≠ 0x3FFFFFFF, this parameter indicates that PES bytes are generated by the encoder at a constant rate. SC_byterate is the ratio of 27MHz to (byte_rate/50), that is, SC_byterate = 1 350 000 000/byte_rate

vbv_delay_flag "1" indicates that the video parameter vbv_delay may be used for timing recovery.

3.5 PRACTICAL SYSTEMS DECODERS

Practical Systems decoders must take into account several types of transmission and multiplexing impairments that are unknown at the time of encoding of the elementary streams. Aside from

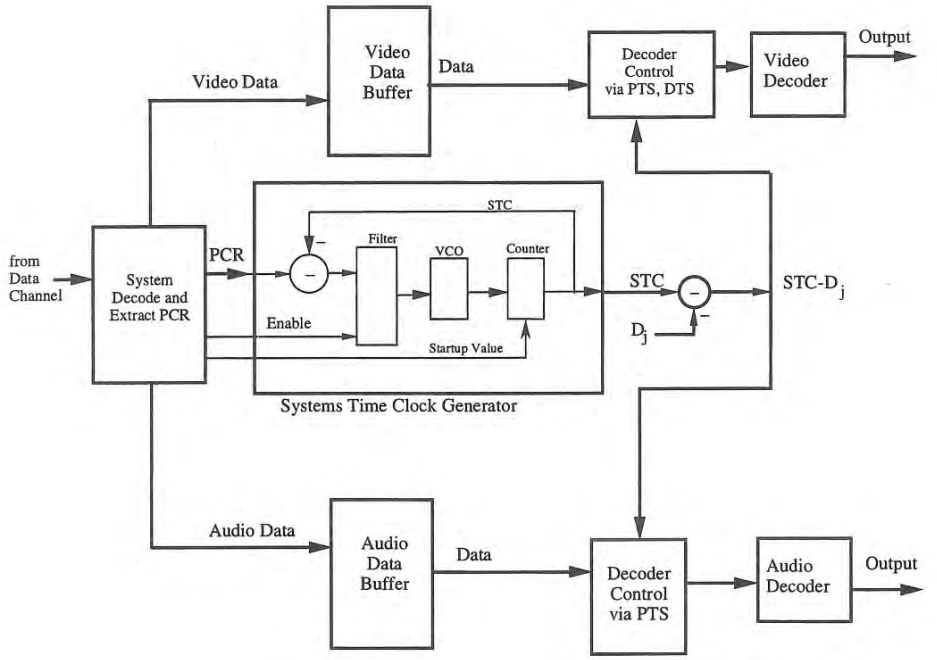


Fig. 3.15 Timing recovery in an MPEG-2 Systems Decoder.

transmission errors and packet losses, which the PES decoders must deal with, the most debilitating transmission effect for System decoders is probably due to transmission delay variation, also called jitter. This causes the received SCRs and PCRs to be in error by an amount dependent on the delay variation.

To smooth the effects of the PCR errors, practical decoders commonly utilize Phase Locked Loops (PLL) to recover stable STC timing. A typical timing recovery system is shown in Fig. 3.15. Here, the differences between the decoder STCs and the received PCRs are averaged (filtered) and the result is used to either speed up or slow down a Voltage Controlled Oscillator (VCO) that clocks the STC counter.

This design can also deal with small as well as large transmission delay jitter by adapting to whatever jitter is present in the received data stream. In this system, the received jitter is estimated as the peak positive value of the difference (STC − PCR). Extra delay is then introduced to accommodate for the jitter by setting Dj equal to the peak jitter value. Increased buffer sizes are also needed in the presence of jitter.

3.6 PRACTICAL SYSTEMS ENCODERS

Practical Systems multiplexers must ensure that conforming MPEG-2 decoders are able to decode the multiplexed data streams. One mechanism for achieving this is shown in Fig. 3.16. Here the Systems Multiplex Controller keeps track of the fullness of the decoder and encoder buffers[2] and controls the elementary stream encoders to ensure that buffers do not overflow or unexpectedly underflow. In some implementations the decoder feeds back an estimate of the jitter that is encountered during transmission. The encoder can then adapt its

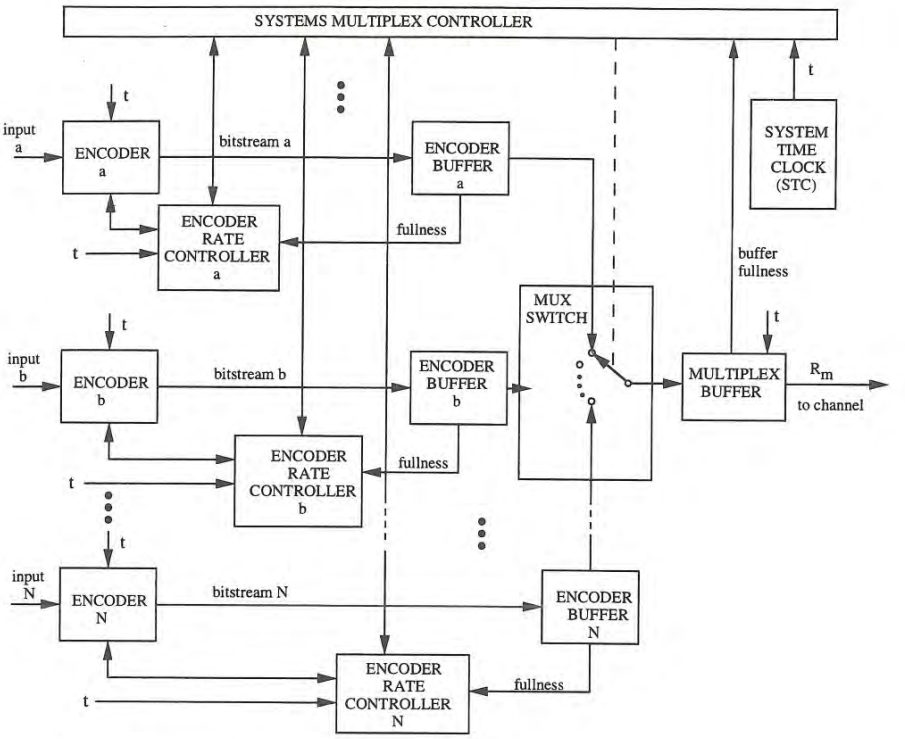


Fig. 3.16 Example of a Systems multiplexer.

buffer control to minimize the occurrence of overflow or underflow due to jitter in transmission.

3.7 CONCLUSION

In this chapter we have seen that a variety of mechanisms are provided in MPEG-2 for multiplexing Elementary Streams from one or more programs. Through the Program Map Tables, Program Descriptors, and Elementary Stream Descriptors, a decoder can discern the interrelationships between the elementary treams of a program to develop a strategy for appropriate display.

REFERENCES

1. A. G. MacInnis, "The MPEG Systems Coding Specification," *Signal Processing: Image Commun.* pp. 153–159 (1992).
2. B. G. Haskell and A. R. Reibman, "Multiplexing of Variable Rate Encoded Streams," *IEEE Trans. Circuits and Systems for Video Technology*, 4(4): 417–424 (August 1994).

4

Audio

4.1 INTRODUCTION

Although this book deals mainly with video issues, we present this chapter to acquaint readers with the fascinating technology of audio coding. The methods employed here exploit the properties of human hearing to a far greater extent than video coding researchers have been able to utilize the characteristics of human vision.

The ITU has a plethora of speech compression and coding standards,[1] as shown in Table 4.1, reflecting its various needs for efficient speech transmission over a variety of telecommunication networks. For example, ITU-T G.711 is simple 8-bit PCM, 64 kbits/s coding of telephone quality audio with a nonlinearity introduced prior to A/D and following D/A. ITU-T G.722 utilizes DPCM coding of monaural AM radio quality audio to compress the data again into 64 kbits/s. ITU-T G.728 employs even more compression technology, specifically Linear Predictive Coding and Vector Quantization, to squeeze telephone speech into 16 kbits/s. And the now finished G.723 is able to obtain reasonable quality speech at rates as low as 5.3 kbits/s.

ISO has relatively few audio compression standards, compared with ITU-T. ISO MPEG-1 audio,[2] officially known as ISO 11172-3, is basically for two-channel stereo. ISO MPEG-2 audio,[3] officially known as ISO 13818-3, extends this capability to five-channel surround sound, with the option of a sixth low-frequency channel. MPEG-2 audio comes in two flavors, one that is backward compatible* with MPEG-1, and one that is not.

Wideband audio has a considerably larger bandwidth than telephone speech. This causes the

Table 4.1 Commonly used ITU-T Monaural Speech Coding Standards.

ITU-T Designation	Bandwidth (Hz)	Sampling Rate (kHz)	Bit Rate (kbits/s)
G.711	200 to 3200	8	64
G.722	50 to 7000	16	64
G.721	200 to 3200	8	32
G.728	200 to 3200	8	16
G.723	200 to 3200	8	5.3 and 6.3

*Backward compatible means MPEG-1 can play MPEG-2-BC audio streams. This feature necessarily requires a tradeoff between MPEG-1 quality and MPEG-2 quality.

Table 4.2 Audio bitrates and total bitrates for Compact Disk (CD), Digital Audio Tape (DAT), Digital Compact Cassette[13] (DCC), and MiniDisk[14] (MD).

Digital Storage Medium	Audio Bandwidth (kHz)	Sampling Rate (kHz)	Stereo Audio Bitrate (kbits/s)	Coding
CD	20	44.1	1411.2	PCM
DAT	16	32.0	1024.0	PCM
ProDAT	16	44.1	1411.2	PCM
DAT	16	48.0	1536.0	PCM
DCC	20	44.1	384	PASC
MD	22	44.1	292	ATRAC

uncompressed data rates to be also much larger, as shown in Table 4.2. Uncompressed audio typically comprises 16 bits per sample and two channels of stereo for a total bitrate of 32 times the sampling rate. However, optical and magnetic storage devices are fairly error prone and require 100% to 200% of extra bits for error correcting redundancy. Thus, the total bitrate is two to three times the audio bitrate.

4.2 AUDIO COMPRESSION FUNDAMENTALS

Most wideband audio encoding, and MPEG in particular, heavily exploit the subjective *masking effects* of the human ear to make any coding noise as inaudible as possible. More specifically, if a tone of a certain frequency and amplitude is present, then other tones or noise of similar frequency, but of much lower amplitude, cannot be heard by the human ear. Thus, the louder tone *masks* the softer tone, and there is no need to transmit the softer tone.

The maximum nonperceptible amplitude level of the softer tone is called the *Masking Threshold*. It depends on the frequency, amplitude, and time history of the louder (masker) tone, and is usually plotted as a function of the softer (maskee) noise frequency, as shown in Fig. 4.1.

In addition to the frequency masking described above, the human ear is also affected by *Temporal*

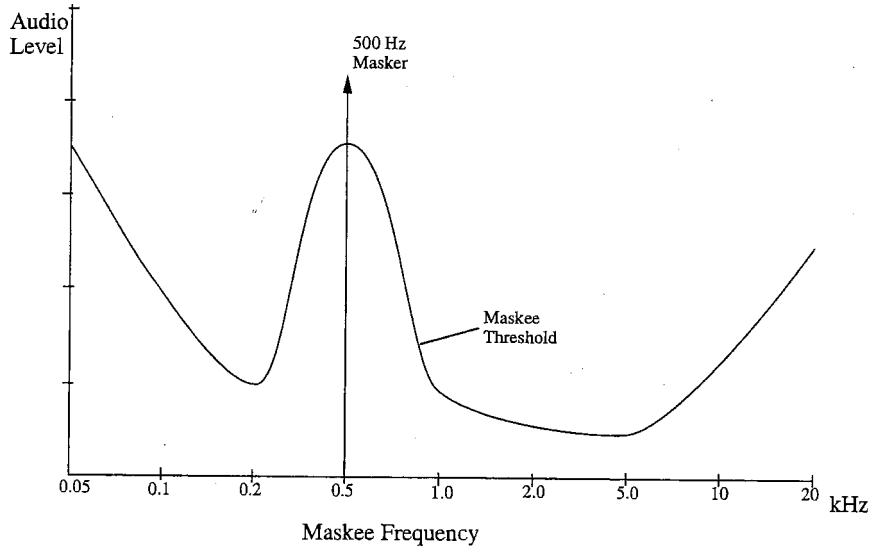


Fig. 4.1 With Frequency Masking, a louder tone masks softer tones of nearby frequency. The *Masking Threshold* is plotted as a function of the softer (Maskee) noise frequency. If the maskee amplitude is less than the Masking Threshold, the maskee is inaudible.

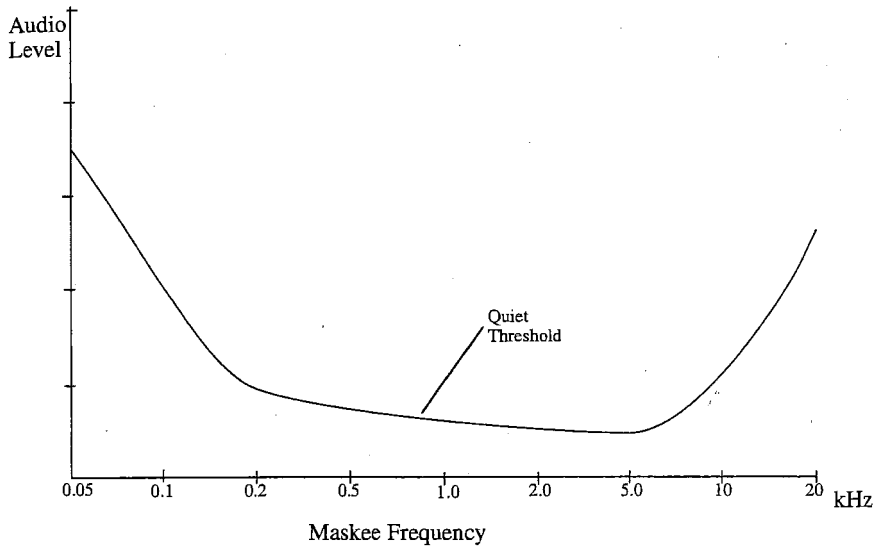


Fig. 4.2 The *Absolute Threshold.* A single tone or noise by itself is inaudable if it is below the Absolute Threshold.

Masking. That is, a loud tone of finite duration will mask a softer tone that follows it or very closely preceeds it in time.* Thus in Fig. 4.1, if the masker tone suddenly disappears, the masking Threshold does not simultaneously disappear. Instead, it dies away over a period of up to 200 ms, depending on masker amplitude and frequency.†

If the masker tone decreases in amplitude, then generally the Masking Threshold also decreases. However, there is a lower limit below which the Masking Threshold will not fall no matter how small the masker. This limit, shown in Fig. 4.2, is called the *Absolute Threshold.* A single tone or noise even with no masker is inaudible if it is below the Absolute Threshold.

With normal audio signals, a multitude of maskers are usually present at a variety of frequencies. In this case, by taking into account the Frequency Masking, Temporal Masking, and Absolute Threshold for all the maskers, the overall net *Masking Threshold* can be derived. Thus, the Masking Threshold is a time varying function of frequency that indicates the maximum inaudible noise at each frequency at a given time, as illustrated in Fig. 4.3.

The Masking Threshold is used by the audio encoder to determine the maximum allowable quantization noise at each frequency to minimize noise perceptibility.

Another technology used by wideband audio coders is called *Subband Coding.*[6] With this approach, shown in Fig. 4.4, the incoming digital PCM signal is first decomposed into N digital bandpass signals by a parallel bank of N digital bandpass *Analysis* filters. Each bandpass signal is then subsampled (or *decimated*) by a factor of N so that the total number of samples in all the subbands is the same as the PCM input. After decimation, each of the bandpass signals becomes, in fact, a low-frequency baseband signal that can be coded and compressed using any baseband compression algorithm.

Reconstruction of the original PCM signal is accomplished by first inserting $N - 1$ zero samples between each sample of the decimated subband signals, which increases the sampling rate of each bandpass signal to its original value. The *upsampled* bandpass signals are then fed to their respective bandpass *Synthesis* filters and finally summed to obtain the full bandwidth PCM output signal.

*Maskee tones that follow the masker are masked significantly more than maskee tones that precede the masker.

†Most of the decay is finished by about 30 ms.

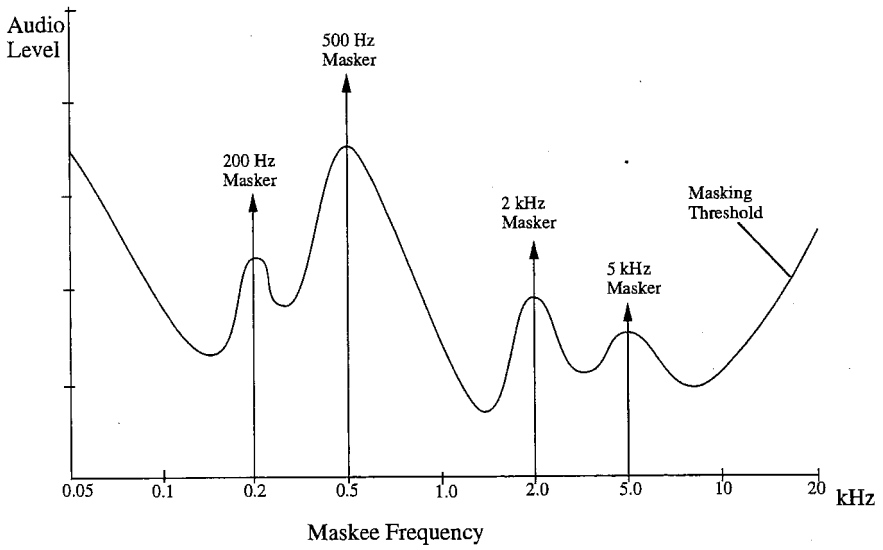


Fig. 4.3 With normal audio signals containing many maskers at a variety of frequencies, the overall net Masking Threshold is calculated from the Frequency Masking, Temporal Masking and Absolute Threshold for all the maskers. The Masking Threshold is a time varying function of frequency that indicates the maximum inaudible noise at each frequency.

Since practical filters can only approximate the ideal bandpass characteristic, each subsampled bandpass signal will contain some alias components that would normally prevent a distortion free reconstruction of the original PCM signal. It turns out, theoretically at least, that by using specially designed bandpass filters, the aliasing in the bandpass signals can be made to cancel when they are summed at the final output. One such set of filters is called *Quadrature Mirror Filters* (QMFs). A particularly efficient QMF filter bank implementation is possible through the use of *Polyphase Networks.*[7] Another such set of filters are called Lapped Orthogonal Transforms (LOTs) or Extended Lapped Transforms (ELTs).[8] These are basically block transforms with overlapping blocks.

Even with QMFs and ELTs, certain approximations and compromises are usually necessary, with the result that performance of the bandpass filters is never completely ideal. Also, with lossy compression coding, the bandpass signals themselves are rarely recovered with full accuracy at the decoder, which further contributes to non-ideal operation.

4.3 MPEG-1

In the first phase of MPEG, a compression standard for two-channel stereo audio was developed that gives near Compact Disk (CD) quality at a bitrate of 256 kbits/s. MPEG-1 is officially known as ISO/IEC 11172–3.

In principle, the standard does not specify the encoder, thus allowing for an evolution as technology progresses and costs decline. MPEG-1 only specifies the decoder, and even here the specification ignores post processing and error recovery.

The algorithm has three possible *Layers** of operation, namely I, II, or III. In principle, the encoder chooses the Layer depending on how much quality is needed or equivalently how much compression is desired. However, in some applica-

*Although MPEG uses the term Layers, they are in no way hierarchical. Audio Layers I, II, and III are three different coding algorithms.

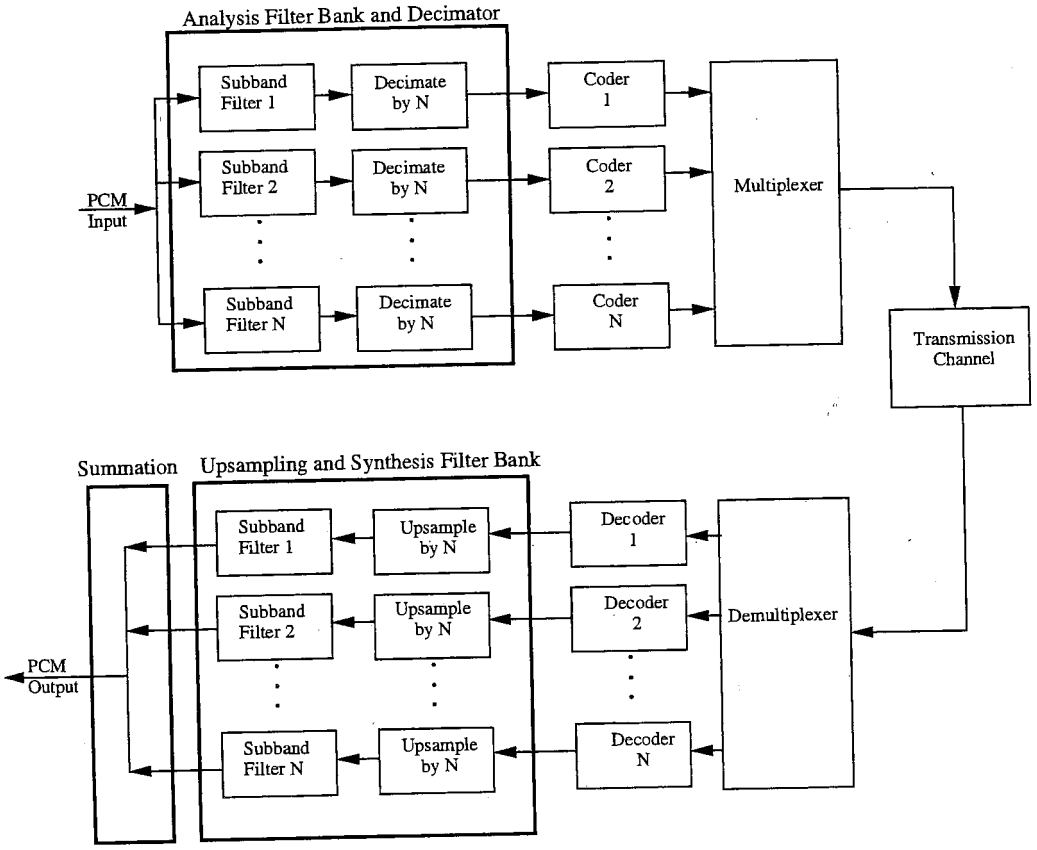


Fig. 4.4 Subband Audio Coder (single channel). The incoming signal is fed to a parallel bank of N digital bandpass *Analysis* filters to produce N digital bandpass signals. These are then decimated by a factor of N, coded and transmitted. Each coded bandpass signal thus consists of one sample for every N input PCM samples. At the decoder the bandpass signals are upsampled by a factor of N by the insertion of zeros, then fed to a bank of digital bandpass *Synthesis* filters and finally summed to produce the final PCM output.

tions, decoders only implement the first one or two layers, which restricts the encoder choices.

MPEG-1 can code a single channel, a stereo pair, or two independent (perhaps multilingual) channels. With a stereo pair, the two audio channels can be coded either separately or jointly. In the latter case, MPEG-1 provides rudimentary methods for exploiting the redundancies between the left and right channels for a modest bitrate reduction.[9]

MPEG-1 audio allows for initial sampling rates of 32, 44.1, and 48 kHz. A variety of fixed bitrates can be specified, ranging from 32 kbits/s up to 448 kbits/s depending on the Layer, or alternatively a fixed unspecified bitrate is also possible. Pseudo-variable-bitrate operation is possible in Layer III.

An MPEG-1 audio encoder and decoder example for Layers I and II is shown in Fig. 4.5. A 16-bit PCM input signal is first fed to a polyphase, approximately ELT, analysis filter bank consisting of 32 equally spaced bandpass filters. Thus, each band has a width of $1/64$ of the PCM sampling rate.

The bandpass analysis filters are of order 511 and have fairly steep rolloff, providing more than 96 dB of alias suppression. As mentioned previously, ELTs with no distortion would ordinarily cancel the alias components at the decoder. However, with coding and compression, the subbands suffer

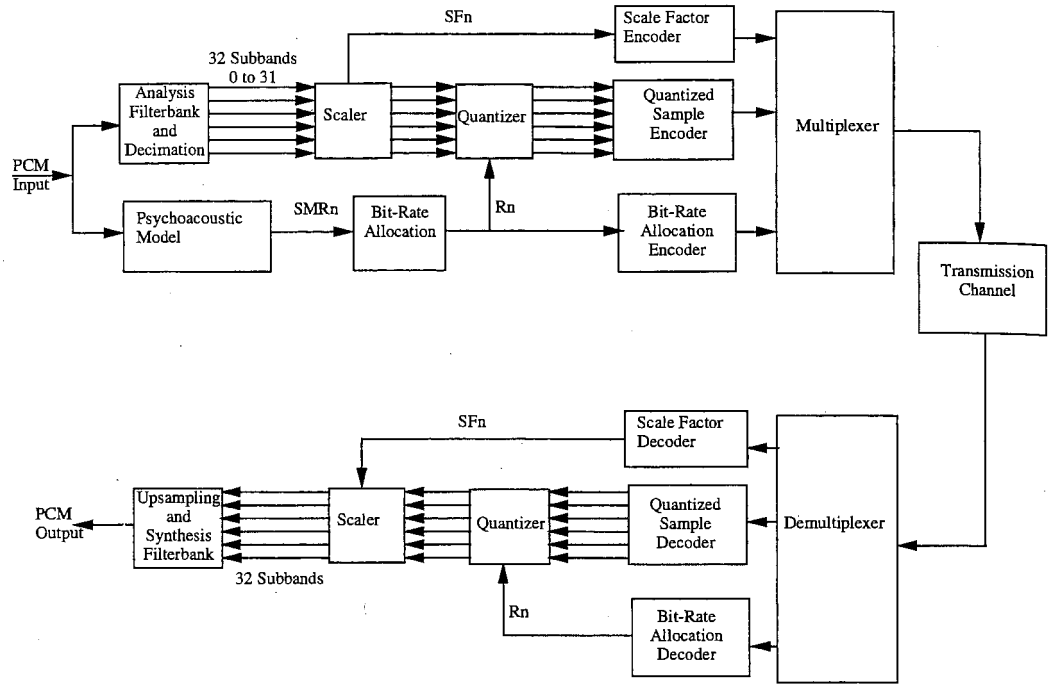


Fig. 4.5 MPEG-1 audio encoder and decoder for Layers I and II (single channel). The standard does not specify the encoder in order to allow for evolution over time. Thus, this is only an example.

from a certain level of quantization noise, and this can produce unwanted alias components if they are not attenuated as much as possible in the analysis filter bank. Following analysis, each of the 32 bandpass outputs is subsampled (decimated) by a factor of 32.

For each subband n (n = 0 to 31), the decimated samples are assembled into blocks of 12 consecutive samples, giving 32 parallel blocks Bn that correspond to $32 \times 12 = 384$ input samples.

For each block Bn, a *Scale_Factor* SFn is then chosen from a table of 63 allowable values, such that when the samples of the block are divided (i.e., normalized) by that Scale_Factor, they all have absolute value less than 1. This is typically done by first finding the sample with the largest absolute value, and then choosing the next higher value in the table for SFn.

4.3.1 Psychoacoustic Model

In parallel with these operations, the Masking Threshold is computed for each group of 384 input samples, that is, each group of 32 blocks Bn. Computation of the Masking Threshold, which is referred to as the *Psychoacoustic Model*, is an encoder-only operation and is not specified by the standard. However, MPEG does provide two examples of Psychoacoustic Models, the first of which we will now describe.* It consists of nine steps, as shown in Fig. 4.6:

1. Calculation of the Frequency Spectrum via a windowed FFT.† This generally uses a 512-point (Layer I) or 1024-point (Layer II) FFT with a raised cosine Hann window. The resulting spectral resolution is much finer than the subbands, giving 8 points per subband for Layer I and 16 points per

*Psychoacoustic Model 1 is the simpler of the two. Psychoacoustic Model 2 performs better for Layers II and III.
†Fast Fourier Transform.

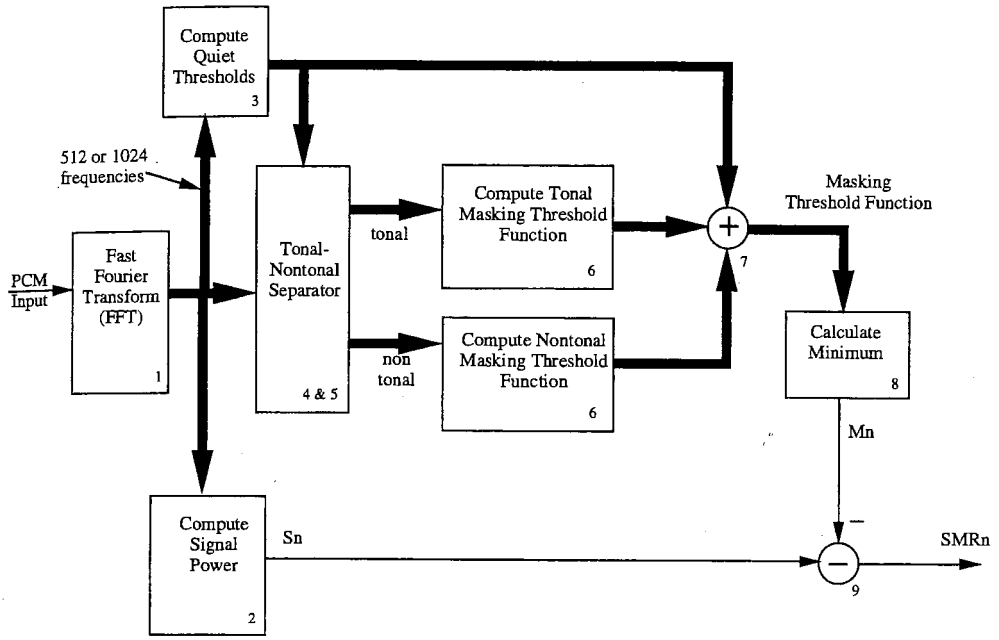


Fig. 4.6 Psychoacoustic Model suitable for MPEG-1 Layers I and II. The standard does not specify the Psychoacoustic Model. Thus, this is only an example.

subband for Layer II. This increased resolution is needed to accurately estimate masking, especially at the lower frequencies.

2. Computation of *Sound Pressure Level* (Sn) within each block Bn. Either the maximum FFT value in the block subband is used or a normalized value of Scale_Factor SFn is used, whichever is larger. For pure tones within the block subband the FFT value will be larger, whereas for noiselike sounds the normalized SFn may be larger.

3. Determination of the Absolute Threshold for each FFT component. These values are simply taken from empirically derived tables. For higher quality coding, that is, bitrates ≥96 kbits/s, the value is reduced by 12 dB.

4. Detecting the tonal (more sinusoidlike) and nontonal (more noiselike) components. A tone is basically an FFT component that is a local maximum surrounded by at most one or two other significant components within a predefined bandwidth of nearby frequencies.* The predefined bandwidths are smaller than a subband for low-frequency tones and larger than a subband for high-frequency tones. Tonal components are identified, their frequencies noted, and their signal powers measured.

Each tonal component and its nearby frequencies in the predefined bandwidths are then removed from the set of FFT components to obtain the nontonal or noiselike components. These are then characterized by measuring the remaining signal power within each band of a contiguous set of *Critical Bands*, which are specified in empirically derived tables. Nontonal components are then assigned a representative frequency within each Critical Band.

5. Detecting only the relevant tonal and nontonal maskers. First, tonal or nontonal components less than the Absolute Threshold are discarded. Next, if two tonal components are closer in frequency than a specified amount, then the smaller component is discarded.

6. Computing the thresholds for each masker. For each relevant tonal masker, a masking threshold

*This is a signal processing definition of tones. It is not necessarily a good definition of tones as perceived by the human ear.

function is computed that is an empirically derived nonlinear function of masker signal power and masker frequency. For each relevant nontonal masker, a masking threshold function is also computed that is a different nonlinear function of masker power and frequency. A given amount of tonal masker power generally provides less masking than the same amount of nontonal masker power.

7. The Masking Threshold at each maskee frequency is then computed as a sum in the signal power domain* of the Absolute Threshold, the tonal masker thresholds, and the nontonal masker thresholds. Typically, Masking Thresholds are computed separately for each channel of a stereo pair. However, in some applications it may be possible to exploit further masking of one stereo channel by the other.

8. Calculating the minimum masking threshold value Mn in each block Bn. The minimum value of the Masking Threshold is computed for frequencies within the block Bn subband. Mn is then set to this value.

9. Computing the signal-to-minimum_mask ratio in each block.

$$\text{SMRn} = \text{Sn} - \text{Mn (in dB)}.$$

Following the calculation of the SMR values, we then carry out bit assignments, quantization, and coding for each block Bn. In the case of stereo, this is done simultaneously for both channels, which allows for adaptive bit assignment between channels. Subband samples of each block Bn are then quantized by a uniform midstep quantizer† having the assigned number of levels. However, bit assignment and choice of the number of quantization levels in each block is done differently in Layers I and II, as we shall see later.

4.3.2 MPEG-1 Syntax and Semantics

Data from all blocks from both channels, including bit assignments Rn, Scale_Factors SFn, coded samples, and so forth are then assembled into a *Frame* of data,‡ preceded by a 32-bit Frame Header. The syntax of the MPEG-1 Frame and Header is shown in Fig. 4.7.

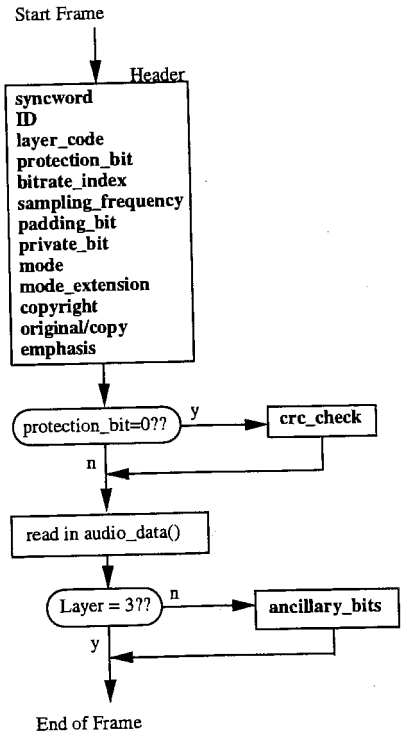


Fig. 4.7 Syntax of an MPEG-1 Audio Frame and Header. Two channels (one if single_channel) are carried in one Frame. A boldface variable by itself means simply read in the value of the variable.

The semantics of the MPEG-1 Frame and Header are as follows:

syncword 12 bits, all 1 s.

ID 1 bit, indicates the algorithm. **ID** = 1 for MPEG-1 audio. **ID** = 0 for low rate extension of MPEG-1 as defined in MPEG-2.

layer_code 2 bits, indicates the Layer. **layer_code** = 4 − Layer, where Layer = 1, 2, or 3.

protection_bit 1 bit. 0 indicates **crc_check** is present. 1 indicates it is not.

*Individual masker and Absolute Thresholds are usually specified in dB = 10 log(power).

†A uniform midstep quantizer has an odd number N of levels and contains the zero level. The set of levels is $\{i \times Q$, for $i = 0, \pm 1, \pm 2, \pm 3, \ldots \pm (N-1)/2\}$, where Q is the step size. If samples to be quantized have a range of −1 to +1, then $Q = 2/N$.

‡Not to be confused with a video frame.

Table 4.3 Modes of audio for MPEG-1.

mode	channel assignment
00	stereo
01	joint_stereo (intensity_stereo for Layers I and II, ms_stereo for Layer III)
10	dual_channel
11	single_channel monophonic

bit-rate_index 4 bits, indicates the bitrate. 0 indicates a nonstandard bitrate. 14 standard values depend on Layer and ID.

sampling_frequency 2 bits, indicates the sampling frequency, which depends on ID.

padding_bit 1 bit; 1 indicates the frame contains additional bytes to control the mean bitrate.

private_bit bit for private use. This bit will not be used in the future by ISO.

mode 2 bits, according to Table 4.3. In stereo modes, channel 0 is the left channel.

mode_extension 2 bits, used in joint_stereo mode. In Layers I and II they indicate which subbands are in intensity_stereo and which are coded in normal stereo. In Layer III they indicate which type of joint stereo coding method is applied.

copyright 1 bit, 1 indicates copyright protected.

original/copy 1 bit; 1 indicates an original.

emphasis 2 bits, indicates the type of deemphasis.

crc_check 16-bit Cyclic Redundancy Code computed for Header. Used for optional error detection.

audio_data( ) Contains coded data for audio samples.

ancillary_data Contains ancillary data, if any.

4.3.3 Layer I Coding

Layer I is capable of compressing stereo audio to approximately 384 kbits/s with near CD quality. It utilizes 15 possible quantizers, with a new quantizer being selected for each block. Layer I assigns Rn bits per sample to be assigned to each block Bn, where either Rn = 0 or Rn = 2 to 15. If Rn > 0 bits per sample are assigned to a block, then the samples of that block (normalized by SFn) are quantized by a uniform midstep quantizer having $2^{Rn} - 1$ levels and step size Qn = $2/(2^{Rn} - 1)$. If zero bits are assigned, all samples are set to zero.

For the optimization of subjective quality versus bitrate, we first note that, to a very good approximation, the ratio of signal power to quantization noise (SNR) depends only on the number of quantization levels.* In fact, MPEG provides tables of assumed SNR versus number of quantization levels for audio waveforms. For zero bits per sample (all samples set to the zero level), SNR = 0 dB, whereas for 15 bits per sample (32,767 levels), SNR = 92 dB.

For optimum quality within the constraints of available bitrate, we would like the ratio of noise power to masking_threshold (NMRn) to be fairly uniform over the subbands. If the bitrate is high enough to allow for NMRn < 0 dB for all blocks, then the quantization noise will be imperceptible. However, this cannot always be achieved, especially for low bitrates.

The quantization bit assignment Rn for each block may be accomplished through an iterative procedure that starts out with zero bits per block, that is, Rn = 0 and SNRn = 0. For each block Bn, we compute NMRn from

$$\text{NMRn} = \text{SMRn} - \text{SNRn dB}$$

We then look for the block having maximum NMRn, and increase its bit assignment Rn. If the old Rn was zero, then 6 bits for the Scale_Factor SFn must also be take into account. We then recompute NMRn and repeat the process iteratively until all bits available for coding are used up. The normalized samples of all blocks assigned Rn > 0 are then quantized and coded using a fixed length code† of Rn bits per sample.

The Layer I decoder is much simpler than the encoder, since it does not need the Psychoacoustic Mode or the bit assignment decision process. The decoder basically processes the received stream according to the syntax below, and then reproduces subband samples and PCM audio samples as described previously.

*For Rn > 2, SNR ≈ Rn × 6 dB is not a bad approximation. For Rn = 2, SNR ≈ 7 dB.

†Binary values 0 to $2^{Rn} - 2$ are used. The value $2^{Rn} - 1$ is reserved for synchronization.

4.3.4 Intensity_Stereo Mode

With the Stereo mode, the left and right channels are normally coded separately. However, in case there are not enough bits to achieve high quality in certain Frames, the Intensity_Stereo mode may be used. This mode exploits a characteristic of human stereo perception in which the stereo effect of middle and upper frequencies depends not so much on the different *content* of the left and right channels, but instead on the differing *amplitudes* of the two channels. Thus, to save bits in Intensity_Stereo mode, the middle and upper subbands of the left and right channel are added together, and only the resulting summed samples are quantized, coded, and sent as a single channel. However, to maintain the stereo effect, Scale_Factors are sent for both channels so that amplitudes may be controlled independently during playback.* The subband number sb_bound at which Intensity_Stereo coding begins is determined by the Frame Header **mode_extension** variable.

The syntax of the MPEG-1 Layer I audio_data( ) is shown in Fig. 4.8.

The semantics of the MPEG-1 Layer I audio_data( ) are as follows:

nch Specifies the number of channels. nch = 1 for monophonic mode, and 2 for all other modes.

ch Channel number, which ranges from 0 to nch-1.

sb_bound Starting subband number for intensity_stereo.

allocation[ch][sb] (4 bits) Indicates the number of bits used to quantize and code the samples in subband sb of channel ch. For subbands in intensity_stereo mode the bitstream contains only one allocation per subband.

scalefactor[ch][sb] (6 bits) Indicates the Scale_Factor of subband sb of channel ch. Actual values are found from tables.

sample[ch][sb][s] Coded data for samples in subband sb of channel ch.

4.3.5 Layer II Coding

Layer II is capable of compressing audio to 256 kbits/s with near CD quality. It utilizes all the compression methods of Layer I, but further reduce overhead redundancy by exploiting similarities i Scale_Factors of successive blocks in a subband, a well as by reducing the number of quantizer choic es for higher frequency subbands. Layer II is some what more complex and has more delay than Lay er I, however.

Unlike Layer I, which codes 32 blocks at a tim (one block from each analysis filter), Layer II code 96 blocks at a time (three blocks from each analys filter). The first step is to check the three consecu tive Scale_Factors that occur in the three blocks (= 0, 1, 2) of each subband. If all three Scale_Fac tors are nearly identical, then only one value sent. If two out of the three Scale_Factors are sim ilar in value, then two Scale_Factor values are sen Also, if large temporal masking is present, the either one or two Scale_Factor values may be sen Otherwise, all three Scale_Factors must be sen Scale_Factor selection side-information (**scfs** must also be sent to tell the decoder whic Scale_Factors were sent.

The next redundancy to be reduced lies in th number of quantization levels that are allowed i the middle and upper subbands. Since the sampl values in these bands are almost always small, coar er quantization rarely has an audible effect, and saving in bit allocation side information is possible only a few choices of quantization are allowed. I case a high level tone occurs, 16-bit quantization always an option (7 bits for the lower bitrates). Th bit allocation and quantization apply to all thre blocks (36 samples) of each subband.

The next compression feature of Layer II is t allow two new quantizations, namely 5-level and 9 level. In addition, for these new quantizations pl the former 3-level quantization, *sample grouped codi* is used. With sample grouping, three consecutiv quantized samples in a block form one *triplet* an are coded together using one code word. For 3-, 5 and 9-level quantization, a triplet† is coded using 5-, 7-, or 10-bit code word, respectively.

The quantization bit assignment for Layer may be derived iteratively as with Layer I. Howe er, for Layer II we also need to take into accou

*The rate at which Scale_Factors are sent (once per block) is not high enough to reproduce excellent quality stereo. Thus, inte sity_stereo is useful only for medium quality audio, for example, at bitrates well below 384 kbits/s.

†MPEG calls this triplet a granule. However, granule has another meaning in Layer III. Thus, we avoid its usage here.

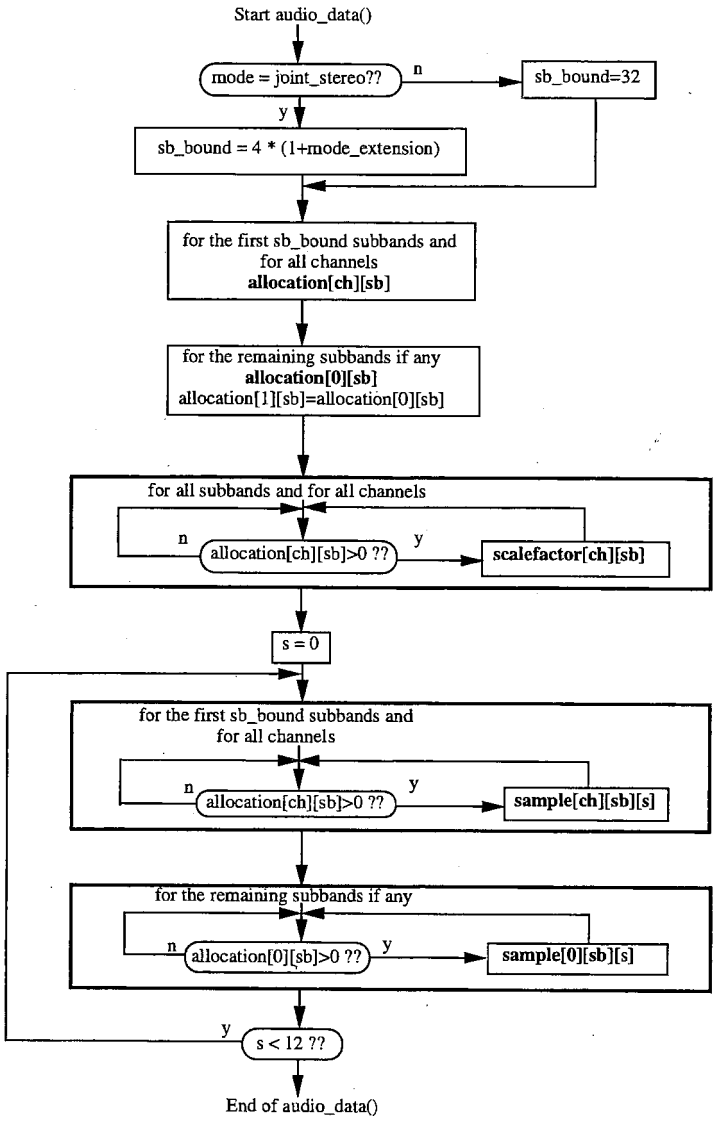


Fig. 4.8 Syntax for MPEG-1 audio_data( ) for Layer I. Each Frame represents 384 samples of input PCM audio.

the variable number of bits due to Scale_Factor coding, bit allocation, side information, and sample grouping.

The Layer II decoder is much simpler than the encoder. The decoder basically processes the received stream according to the syntax below, and then reproduces subband samples and PCM audio samples as described previously.

The syntax of the MPEG-1 Layer II audio_data( ) is shown in Fig. 4.9.

The semantics of the MPEG-1 Layer II audio_data( ) are as follows:

nch Specifies the number of channels. nch = 1 for monophonic mode, and 2 for all other modes.

ch Channel number, which ranges from 0 to nch-1.

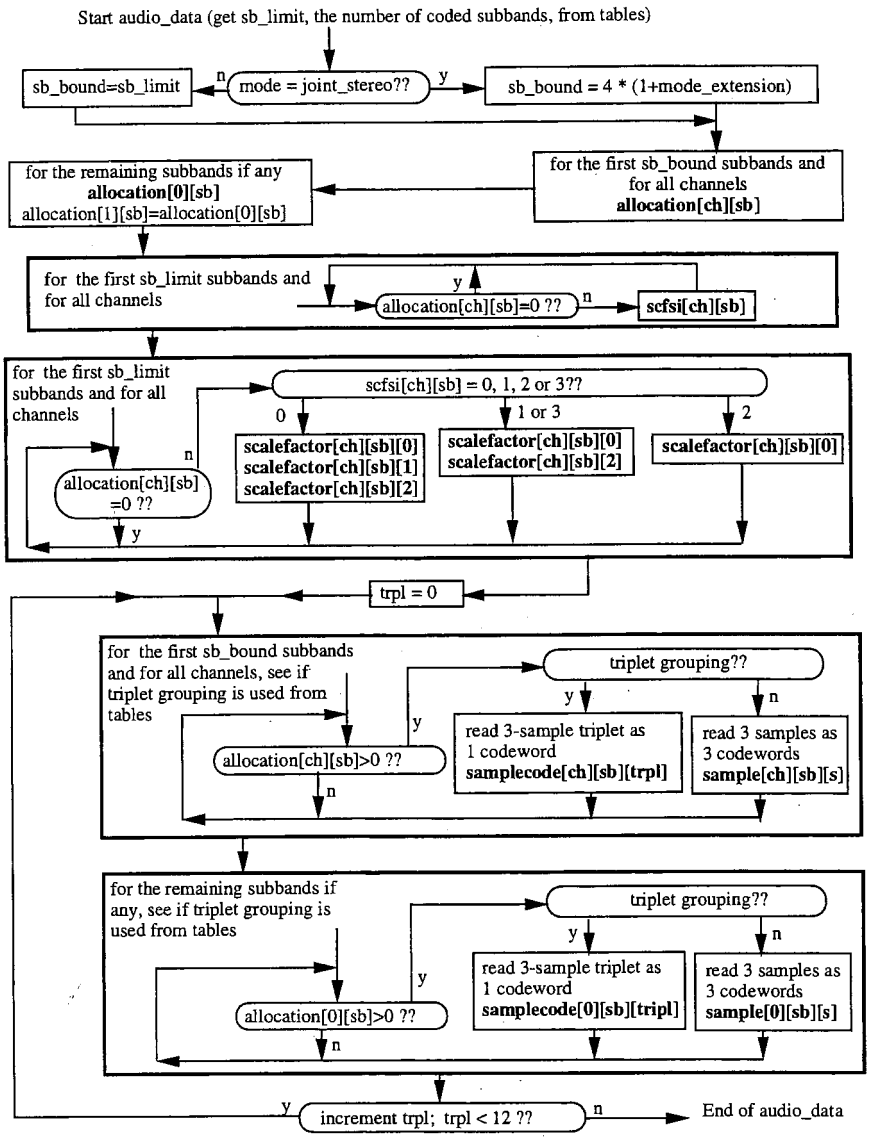


Fig. 4.9 Syntax for MPEG-1 audio_data( ) for Layer II. Each Frame represents 1152 samples of input PCM audio. Boldfac means read in the data from the stream.

sb_limit Number of subbands to be coded. Depends on bit and sampling rates.

sb_bound Starting subband number for intensity_stereo.

trpl 3-sample triplet number.

allocation[ch][sb] 2 to 4 bits (from tables) Specifies the quantizers used in subband sb in channel ch, whether three consecutive samples have been grouped, and the number of bits per sample. A values are taken from tables.

scfsi[ch][sb] (2 bits) Specifies the numbe Scale_Factors for subband sb in channel ch ar which blocks they are valid.

scalefactor[ch][sb][b] (6 bits) Indicates Scale_Factor for block b of subband sb of cha ch. Actual values are found from tables.

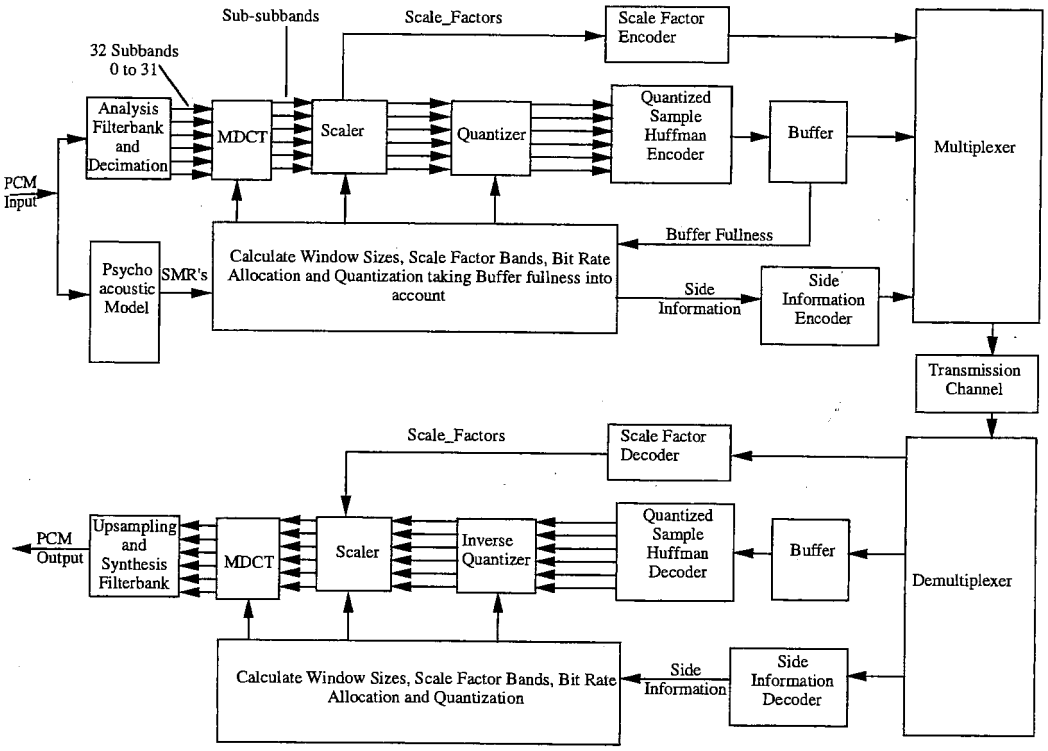


Fig. 4.10 MPEG-1 audio encoder and decoder for Layer III (single channel). The standard does not specify the encoder in order to allow for evolution over time. Thus, this is only an example.

samplecode[ch][sb][trpl] Coded data for three consecutive samples in triplet trpl in subband sb of channel ch.

sample[ch][sb][s] Coded data for sample s in subband sb of channel ch.

4.3.6 Layer III

Layer III is considerably more complex and utilizes many more features than Layers I and II. It is capable of compressing stereo audio to 128 kbits/s with adequate entertainment quality, but usually not CD quality. An example of an MPEG-1 Layer III coder and decoder is shown in Fig. 4.10.

The first improvement over Layer II is to increase the frequency resolution. This is achieved by further processing each subband analysis filter output using a Modified Discrete Cosine Transform (MDCT). We call the MDCT outputs sub-subbands. The increased sub-subband resolution allows for better perceptual coding, especially at lower frequencies where the Masking Threshold may have considerable variation within a subband. It also allows for some cancellation of aliasing caused by the polyphase analysis subband filters within a subband.

The MDCTs can be switched to generate for each subband either 6 sub-subbands (called *short-window* MDCTs) or 18 sub-subbands* (called *long-window*† MDCTs). Long-windows give better frequency resolution, while short-windows give better time resolution. Also, during large transients, the

*The MDCT is a 50% overlapped transform. Thus, it is actually a 12-point or a 36-point transform, respectively.

†In MPEG Layer III, the terms *window* and *block* are used interchangeably. To avoid confusion with earlier terminology, we use only the term *window*.

lower two subband MDCTs can be switched to long-windows, while the upper 30 subband MDCTs are switched to short-windows. This assists dynamic adaptation of the coding while maintaining the higher frequency resolution for the low frequencies.

Adaptation to the transition of an MDCT between long- and short-windows is also possible. This is implemented by two special long-window MDCTs, one for the long-to-short transition and one for the short-to-long transition.

After formation of the sub-subbands, the samples are then assembled in frequency order into *Granules.* A Granule consists of one sample from each long-window sub-subband and three successive time samples from each short-window sub-subband, for a total of 576 samples. Because of the frequency ordering, the quantized sample amplitudes on the average decrease from beginning to end of the Granule. Two Granules make up each audio Frame, which corresponds to 1152 input PCM samples.

There are not enough bits to allow for a separate Scale_Factor for each sub-subband. Thus, the samples of each Granule are adaptively grouped into *Scale_Factor_Bands,* and one Scale_Factor is computed for all the samples in the associated Scale_Factor_Band. If for a particular Scale_Factor_Band, the Scale_Factors in both Granules of a Frame are identical, then the second Scale_Factor need not be transmitted. In this case, the corresponding Scale_Factor of the first Granule is used for both Granules. A further partitioning of scale factor bands into four *Classes* facilitates the transmission of this latter side information.

The Granule samples are then nonuniformly quantized. This is implemented by raising the absolute value of the samples to the power 0.75 before passing to a uniform quantizer. The quantized samples are then coded using variable-length codewords. For this purpose the samples are partitioned into five variable sized *Regions,* each Region using its own set of Huffman code tables especially designed for the statistics expected in that Region. The first three Regions are called *Big Values* Regions. Within these Regions the quantized samples are coded in pairs, that is, two samples per Huffman codeword. The fourth Region of quantized samples (called the *Small Values** Region) consists entirely of zeros and ± 1s. Samples in this Region are coded four at a time, that is, four samples per codeword. The fifth Region, corresponding to the highest frequencies, consists entirely of zeros and is not transmitted.

The Huffman coding process may generate a variable number of bits per Frame, thus requiring a buffer† to connect to a constant rate digital channel. In this case, feedback from the buffer is used to control the quantization to avoid emptying or overflowing the buffer. However, a more important use of the buffer concerns large temporal changes in the audio signal. In case of a sudden increase in audio amplitude, the buffer allows for a higher quality of coding just *before* the transient when noise is more audible, at the expense of lower quality coding *during* the transient when noise is less audible. This is called *Pre-echo* correction and results in a significant improvement in overall coding quality.

To improve error resilience, the Frame Header and some of the side information of Layer III are not necessarily placed at the beginning of the variable length audio data of each Frame. Instead the Header and partial side information may occur later. Specifically, they are placed exactly at the locations they would appear if the Frames were all of constant length. For this to work correctly, a pointer called **main_data_begin** is included after each Frame Header to indicate where the audio main_data for the corresponding Frame actually begins.

The Layer III decoder is much simpler than the encoder. In fact, the Layer III decoder is only slightly more complex that a Layer II decoder. The main increase in complexity is due to the overlapped MDCTs. Because of the overlap, samples from two successive inverse transforms must be added together to produce samples for a subband.

The decoder basically processes the received stream according to the syntax below, and then

*MPEG calls this the count1 region. Don't ask.

†Alternatively, a rate controller may be used to achieve a constant number of bits per frame.

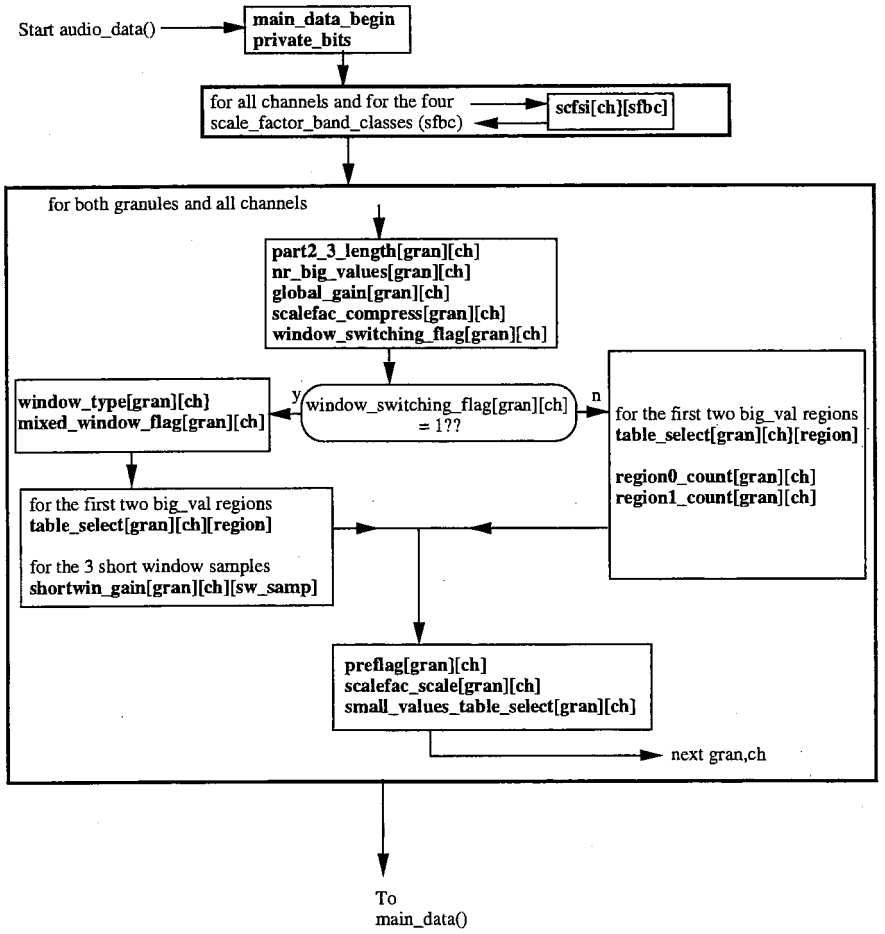


Fig. 4.11a Syntax for MPEG-1 audio_data( ). Each Frame represents 1152 samples of input PCM audio.

reproduces subband samples and PCM audio samples as described previously.

4.3.7 MS_Stereo mode

In addition to Intensity_Stereo Mode, Layer III supports an additional stereo mode called Middle-Side (MS) stereo. With this technique, the frequency ranges that would normally be coded as left and right are instead coded as Middle (left + right) and Side (left − right). This method exploits the fact that the Side channel can be coded with fewer bits, thus reducing the overall bitrate. If MS_Stereo and Intensity_Stereo are both used, then MS_Stereo applies only to the lower bands where Intensity_Stereo is not used.

The syntax of the MPEG-1 Layer III audio_data( ) is shown in Fig. 4.11.

The semantics of the MPEG-1 Layer III audio_data( ) are as follows:

nch Specifies the number of channels. nch = 1 for monophonic mode, and 2 for all other modes.

ch Channel number, which ranges from 0 to nch-1.

sfb Scale_Factor_Band. Ranges from 0 to 20.

sfbc Scale Factor Band Class. The first 6 scale factor bands are in class 0. Classes 1, 2, 3 contain five scale factor bands each.

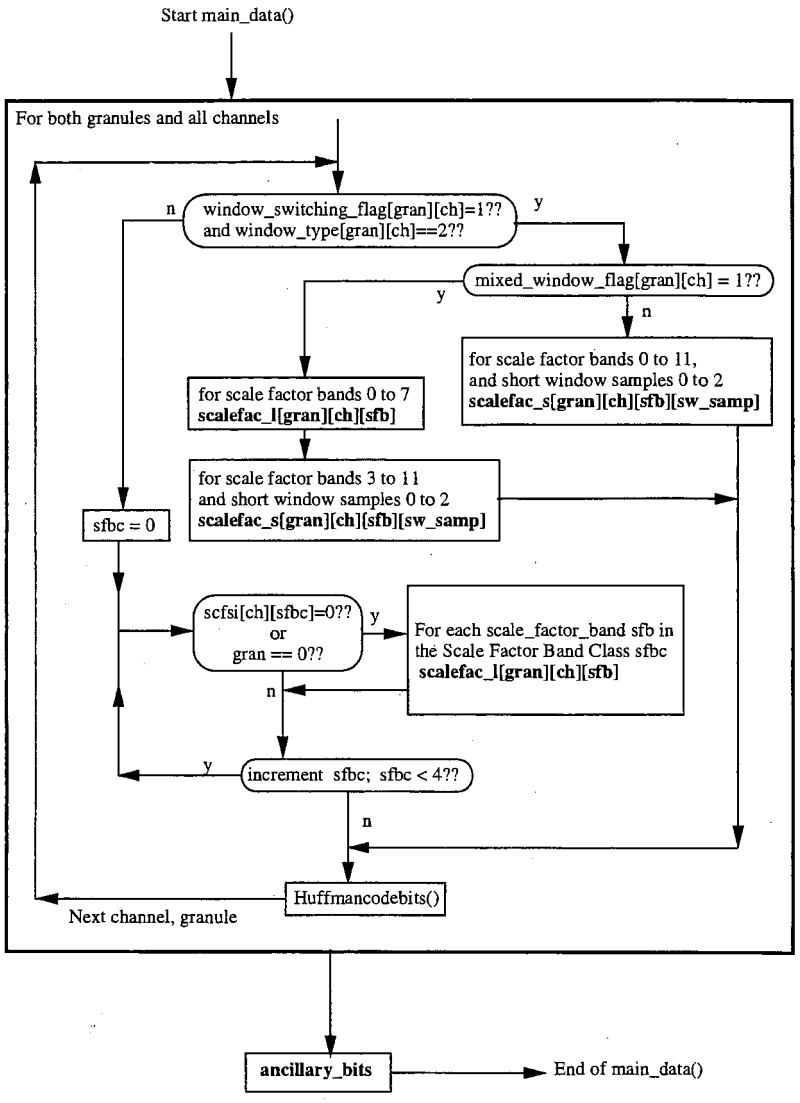


Fig. 4.11b Syntax for MPEG-1 main_data( ) for Layer III.

gran Granule number 0 or 1.

main_data_begin (9 bits) Specifies the location of the audio main_data of a frame as a negative offset in bytes from the first byte of the audio sync word. 0 indicates the main data starts after the Header and side-information.

private_bits 3 or 5 bits depending on mode, for private use.

scfsi[ch][sfbc] (1 bit) Scalefactor selection information, as in Layers I and II. But here it is for classes of Scale_Factor_Bands, instead of subbands.

part2_3_length[gran][ch] (12 bits) Specifies the number of main_data bits used for Scale_Factors and Huffman code data.

nr_big_values[gran][ch] (9 bits) The number of sample pairs in the lowest frequency region (big values region).

nr_small_values The number of sample quadruples in the mid frequency region (Small Values region). It is equal to the remaining samples in the granule, and is found from **part2_3_length** and **nr_big_values.**

global_gain[gran][ch] (8 bits) Specifies quantizer step size for the granule and channel.

scalefac_compress[gran][ch] (4 bits) Specifies the number of bits used for Scale_Factors. Actual values are taken from a Table.

window_switching_flag[gran][ch] (1 bit) Specifies a window other than the normal long window.

table_select[gran][ch][region] (5 bits) Specifies Huffman code tables for the first two big values regions.

region0_count (4 bits) Specifies number of samples in subregion 0 of the big_values region. Depends on **window_type** and **mixed_window_flag.**

region1_count (3 bits) Specifies number of samples in subregion 1 of the big_values region. Depends on **window_type** and **mixed_window_flag.**

region2_count The remaining samples of the big_values region. It is found from **nr_big_values, region0_count,** and **region1count.**

window_type[gran][ch] (2 bits) Specifies window type for the granule.

mixed_window_flag[gran][ch] (1 bit) Specifies that the two lowest subbands have a long window type, which may be different than the higher 30 subbands.

shortwin_gain[gran][ch][sw_samp] (3 bits) Specifies a gain offset from the global_gain for short window samples (sw_samp = 0 to 2) of short windows.

preflag[gran][ch] (1 bit) Specifies additional high frequency amplification.

scalefac_scale[gran][ch] (1 bit) Specifies quantization of Scale_Factors.

small_values_table_select[gran][ch] (1 bit) Specifies Huffman code tables for the Small Values region.

scalefac_l[gran][ch][sfb] (0 to 4 bits) Scale_Factors for long windows.

scalefac_s[gran][ch][sfb][sw_samp] (0 to 4 bits) Scale_Factors for each sample (sw_samp = 0 to 2) of a short window.

huffmancodebits( ) Huffman encoded data

4.4 MPEG-2 AUDIO

For two-channel stereo, the differences between MPEG-1 and MPEG-2 are minor. The initial PCM sampling rate choices are extended downward to include 16, 22.05, and 24 kHz. Also, the preassigned bitrates are extended to as low as 8 kbits/s. The lower sampling and bitrates are signalled by setting **ID** = 0 in the Header.

For the lower rates, MPEG-2 provides better quantization tables. Also, in Layer III, the coding of Scale_factors and intensity_mode stereo are improved.

4.4.1 MPEG-2 Backward Compatible (BC)

The most substantial change in MPEG-2 is the definition of a compression standard for five-channel surround-sound that is backward compatible (BC) with MPEG-1 audio. Thus, MPEG-1 audio players can receive and decode two channels of MPEG-2-BC audio, namely the left (L) and right (R) front speaker signals. However, MPEG-2-BC allows for up to four additional channels, namely front center (C), side/rear left surround (LS), side/rear right surround (RS), plus an optional low-frequency enhancement (LFE) channel.* MPEG-2-BC achieves Backward Compatibility by coding the L and R channels as MPEG-1, while the additional channels are coded as ancillary data in the MPEG-1 audio stream.

The L, C, R, LS, RS arrangement is often referred to as $^3/_2$ stereo, that is, 3 front and 2 rear channels. If LFE is included, it is called 5.1 channel stereo. The compressed data for these channels is called the *Multichannel* or mc coded data.

Other possible channel arrangements are $^3/_1$, $^3/_0$, $^2/_2$, and $^2/_1$, in addition to $^2/_0$ (MPEG-1 stereo) and

*The LFE frequencies typically range from 15 Hz to 120 Hz. One or more loudspeakers are positioned for LFE, which usually reproduces loud sound effects.

1/0 (MPEG-1 monaural). For the cases having three or less channels, it is possible to define a *Second Stereo Program,* perhaps for bilingual, consisting of a second left-right pair (L2, R2) that is carried on two of the unused channels. For example, 3/0 + 2/0 and 2/0 + 2/0 designate dual (perhaps bilingual) stereo programming on one MPEG-2-BC audio stream.

In the following, we will concentrate mainly on Layers I and II. Many of the concepts also carry over to Layer III, where they are coded more efficiently.

4.4.2 Subband Groups

Some of the adaptive and dynamic features of MPEG-2 are specified for Subband Groups instead of individual subbands or sub-subbands. Subband Groups for Layers I and II are defined in Table 4.4. Layer III has a similar definition, except that in Layer III they are called Scalefactorband_Groups.

4.4.2.1 Matrixing for Backward Compatibility and Downward Compatibility*

In many cases, it is advantageous to transmit linear combinations of the five input audio channels instead of the input signals themselves. This is called *Matrixing,*[10] as shown in the matrix equation

$$[\text{Lo Ro T2 T3 T4}] = [\text{L C R LS RS}] \times \mathbf{M}$$

where **M** is a 5 × 5 numerical matrix. The matrix converts the *Audio Multichannels* (L C R LS RS) into *Transmission Channels* (Lo Ro T2 T3 T4). Lo and Ro† are sent as normal MPEG-1 stereo channels, while T2, T3, T4, and LFE,‡ if present, are sent as ancilliary data in the MPEG-1 stream. Matrixing can produce Lo and Ro that are better renditions of two-channel stereo, and therefore give better quality with MPEG-1 decoders. Also, transmission channels T2, T3, and T4 are often easier to compress by exploiting the subjective redundancies and correlations with Lo and Ro. However, better MPEG-1 quality can sometimes mean worse MPEG-2-BC quality when five-channel audio must be compressed.

For downward compatibility or other reasons, we may wish to code fewer than five transmission channels. For example, 2/2 stereo could be produced by the matrixing:

$$[\text{Lo Ro T2 T3}] = [\text{L C R LS RS}] \times \mathbf{M}$$

where M is a 5 × 4 numerical matrix.

4.4.3 Dynamic Transmission Channel Switching

The matrix may be defined for all subbands, or individually for each subband group. The matrix definitions may also be changed dynamically for each new Frame.

4.4.4 Dynamic Crosstalk

Dynamic Crosstalk is the MPEG-2 implementation of intensity_stereo. It allows for dynamic deletion of sample bits in specified subband groups of specified Transmission Channels. As in MPEG-1, the missing samples are copied from another specified Transmission Channel. However, scale Factors are transmitted for the missing samples to

Table 4.4 Subband Groups for MPEG-2 Layers I and II.

Subband Group for Layers I and II	Subbands in Subband Group
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8 to 9
9	10 to 11
10	12 to 15
11	16 to 31

*Downward Compatibility is for decoders that cannot handle the full 3/2 audio channel.
†MPEG sometimes uses T0 and T1 instead of Lo and Ro. However, the meaning is the same.
‡The optional LFE channel is never matrixed with other channels.

allow for independent amplitude control, as in MPEG-1 intensity_stereo.

4.4.5 Adaptive Multichannel Prediction

In Layers I and II, further compression of channels T2, T3, and T4 is allowed using predictive coding. The first eight subband groups may be coded predictively using up to three samples from T0 plus up to three samples from T1. Up to six predictors may be specified, depending on channel assignment and dynamic crosstalk modes. Layer III allows predictive coding of more channels using more samples.

4.4.6 Phantom Coding of the Center Channel

This refers to a compression method for the center channel, wherein the subbands above subband 11 are not transmitted. These are set to zero at the decoder, and the listener perceives high frequencies only from the other channels. In this case, the encoder may choose to matrix some of the center channel into the side channels prior to coding.

4.4.7 Multilingual Channels

MPEG-2-BC allows for up to seven so called *Multilingual* or ml channels. These are monophonic and would typically contain speech in different languages, commentary for the visually impaired, clean speech for the hearing impaired, educational side comments, and so forth. A multilingual channel might be self contained, or it might contain speech or singing that is to be added to the multichannel program signal. For example, the multichannel signal might be in 2/2 format, while one of the multilingual channels is passed to the center channel speaker.

A maximum of seven multilingual channels can be carried on a single MPEG-2-BC stream. In addition, MPEG Systems allows up to 32 audio streams per program. Thus, in principle up to 224 multilingual channels could be carried in an MPEG-2 program.

4.4.8 Syntax and Semantics of MPEG-2-BC Audio

Space does not permit the inclusion of all details of the MPEG-2-BC audio stream. Thus, we will only describe the main structure and arrangement of data for the three Layers. As stated above, the Lo and Ro channels are sent as normal MPEG-1 data. The additional channels plus descriptive data, which is called the **mcml_extension( ),** replaces the ancillary data in the MPEG-1 stream. If there is insufficient capacity in the MPEG-1 stream for all of the **mcml_extension( ),** then the remainder may be carried in an *external stream** having a simple header and packet structure defined by MPEG-2. The external stream is treated by MPEG Systems as a separate audio stream, and normally requires the hierarchy descriptor in the Systems PSI.

In Layer I, three MPEG-1 Frames plus a packet from the external stream, if present, are required to carry the *mcml_extension( )*. In Layers II and III, only one MPEG-1 Frame is needed, plus a packet from the external stream, if present. The syntax of the MPEG-2-BC **mcml_extension ( )** for Layers I, II and III is shown in Fig. 4.12.

The semantics of the MPEG-2-BC **mcml_extension( )** for Layers I, II and III are as follows:

mpeg-1_ancillary_data( ) Data for MPEG-1 decoders, which may be defined in the future.

ext_stream_present (1 bit) 1 indicates that an external stream exists, which contains the remainder of the data in case it does not all fit in one MPEG-1 compatible stream. This bit is redundant if Systems PSI is present.

n_ad_bytes 8 bits, indicating how many bytes are in the MPEG-1 compatible ancillary data in the external stream.

center 2 bits, indicating if a center channel exists and if phantom center coding is used.

center_limited[mch][sb] 1 indicates center channel subbands 12 and above are not sent.

*MPEG calls this the *extension* stream, which we believe is confusing.

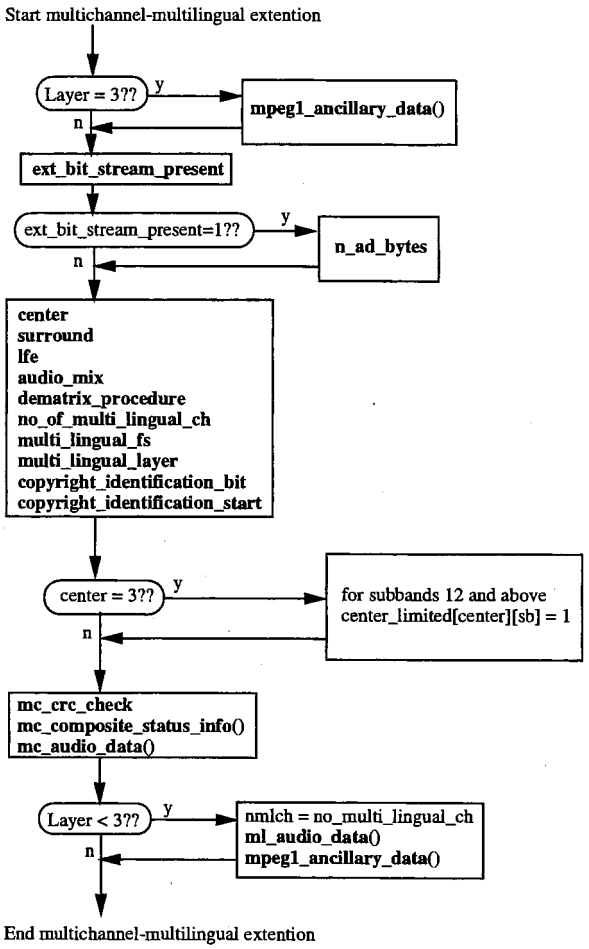


Fig. 4.12 Syntax of the MPEG-2-BC **mcml_extension( ).** Two channels Lo and Ro (one if single_channel) are carried in the MPEG-1 portion of the Frame. The additional channels plus descriptive data is called the **mcml_extension( )** and replaces the ancillary data in the MPEG-1 stream. An external stream may be required to carry excess data. A boldface variable by itself means simply read in the value of the variable

surround 2 bits, indicating the presence of surround channels or the second stereo program.

lfe (1 bit) 1 indicates a low frequency enhancement channel is present.

audio_mix (1 bit) Indicates whether mixing for a large listening room, like a theatre, or for a small listening room. 1 indicates a small listening room.

dematrix_procedure 2 bits, indicating the dematrix procedure to be applied in the decoder.

no_of_multi_lingual_ch 3 bits, indicating the number of multilingual or commentary channels.

multi-lingual_fs (1 bit) 1 indicates the sampling frequency of the multilingual channels is half that of the main audio channels. 0 indicates both sampling frequencies are the same.

multi_lingual_layer (1 bit) Indicates whether Layer II multilingual or Layer III multilingual is used.

copyright_identification_start (1 bit) 1 indicates that a 72-bit copyright identification field starts in this frame to identify copyrighted material.

mc_crc_check (16 bits) Check word for error detection.

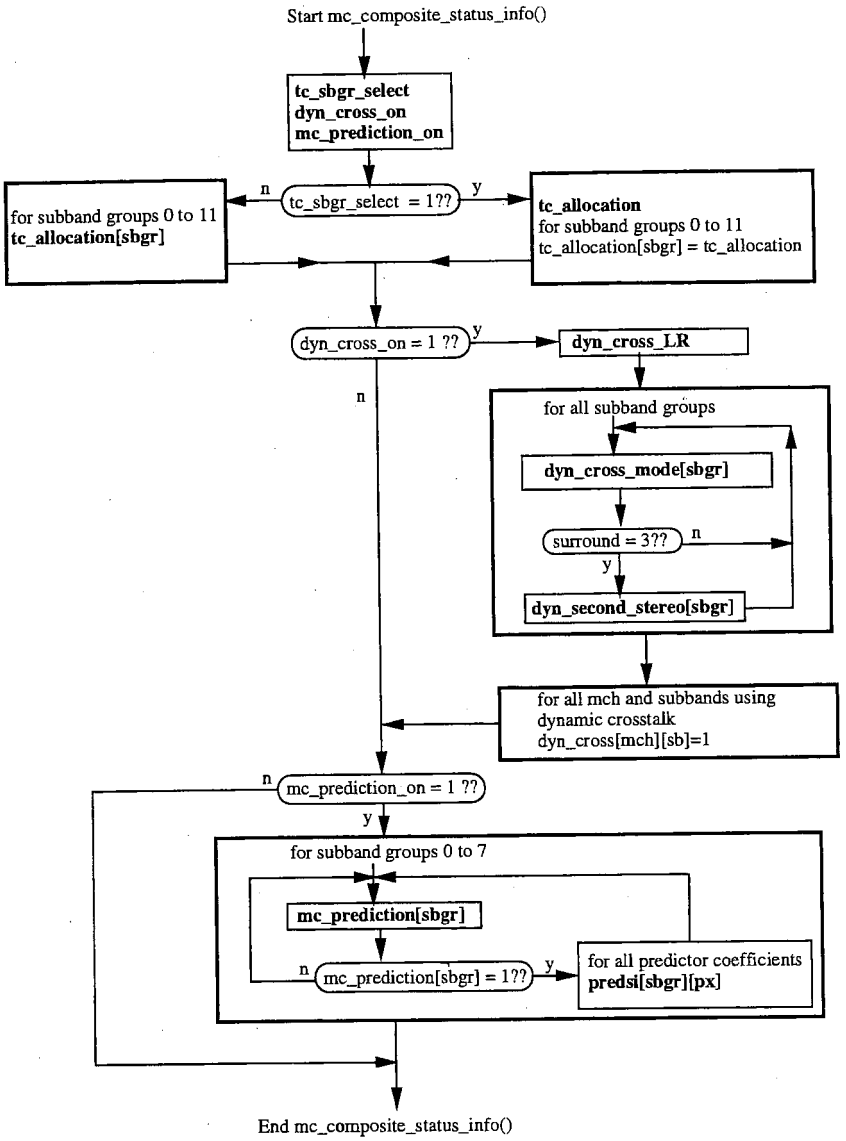


Fig. 4.13 Syntax for MPEG-2-BC **mc_composite_status_info( )** for Layers I and II. Each Frame represents 1152 samples of input PCM audio.

mc_composite_status_info ( ) Specifies information on the multichannel coding mode.

mc_audio_data( ) Coded audio data for multichannel audio.

ml_audio_data( ) Coded audio data for multilingual audio.

The syntax of the MPEG-2-BC Multichannel Layers I and II **mc_composite_status_info( )** is shown in Fig. 4.13.

The semantics of the MPEG-2-BC Multichannel Layers I and II **mc_composite_status_info( )** are as follows:

tc transmission channel.

sbgr subband group number.

dyn_cross_on 1 bit, indicating whether dynamic crosstalk is used.

mc_prediction_on 1 bit, indicating whether mc_prediction is used.

tc_sbgr_select 1 bit, to indicate if the tc_allocation below is valid for all subbands or for individual subband groups.

tc_allocation, tc_allocation[sbgr] (1 bit) Contains information on the multichannel allocation of the additional channels that are not in the MPEG-1 portion of the stream.

dyn_cross_LR 1 bit, indicating whether dynamic crosstalk copying shall be from Lo or Ro.

dyn_cross_mode[sbgr] (1 to 4 bits) Indicates between which transmission channels dynamic crosstalk is active.

dyn_cross[mch][sb] 1 indicates no samples sent for multichannel mch and subband sb.

dyn_second_stereo[sbgr] (1 bit) Indicates whether dynamic crosstalk is used in the second stereo program.

mc_prediction[sbgr] (1 bit) Indicates whether multichannel prediction is used in subband group sbgr.

predsi[sbgr][px] 2 bits, predictor select information. Indicates whether the predictor indexed by px in subband group sbgr is used, and if so, how many predictor coefficients are transferred.

The syntax of the MPEG-2-BC Multichannel Layers I and II **mc_audio_data( )** is shown in Fig. 4.14.

The semantics of the MPEG-2-BC Multichannel Layers I and II **mc_audio_data( )** are as follows:

mch Multichannel (mc) number.

trpl Three-sample triplet number.

lfe_allocation (4 bits) Specifies the quantiser used for the low-frequency enhancement channel.

allocation[mch][sb] (2 to 4 bits) Specifies the quantiser used for the samples in subband sb of the multichannel mch.

scfsi[mch][sb] (2 bits) Scale_Factor select infc mation. Indicates the number of Scale_Factors trar ferred for subband sb of the multichannel mch. As MPEG-1 Layer II, the frame is divided into thr equal blocks (b) of 12 samples each.

delay_comp[sbgr][px] (3 bits) Specifies a sh of subband samples for delay compensation in su band group sbgr and predictor index px.

pred_coef[sbgr][px][pci] (8 bits) Coefficient predictor with up to second order in subband grou sbgr and predictor index px.

Lf_Scale_Factor (6 bits) Scale_Factor for the lo frequency enhancement channel.

Scale_Factor[mch][sb][b] (6 bits) Scale_Fact for block b of subband sb of multichannel mch.

Lf_sample[trpl] (2 to 15 bits) Coded single sar ple of the low frequency enhancement chann There are three normal samples for each LF sample.

samplecode[mch][sb][trpl] (5 to 10 bits) Co ed three consecutive samples in triplet trpl of su band sb of multichannel extension channel mch.

sample[mch][sb][s] (2 to 16 bits) Coded sar ple s of subband sb of multichannel extension cha nel mch.

The syntax of the MPEG-2-BC Multichann Layers I and II **ml_audio_data( )** is shown Fig. 4.15.

The semantics of the MPEG-2-BC Multicha nel Layers I and II **ml_audio_data** are as fc lows:

ntrip Number of sample triplets. It is 12 or depending on whether sampling is at full rate (mu ti_lingual_fs = 0) or half rate (multi_lingual_fs = 1 respectively.

mlch Multilingual (ml) channel number.

allocation[mlch][sb] (2 to 4 bits) Specifies th quantiser used for subband sb of multilingual cha nel mlch. The number of levels is taken from table depending on sampling and bitrates.

scfsi[mlch][sb] (2 bits) Specifies the number scalefactors for subband sb of multilingual chann mlch. As in MPEG-1 Layer II, the frame is divided in three equal blocks (b) of 12 or 6 samples each depen ing on whether sampling is at full or half rate, respe tively. One, two, or three scalefactors may be sent.

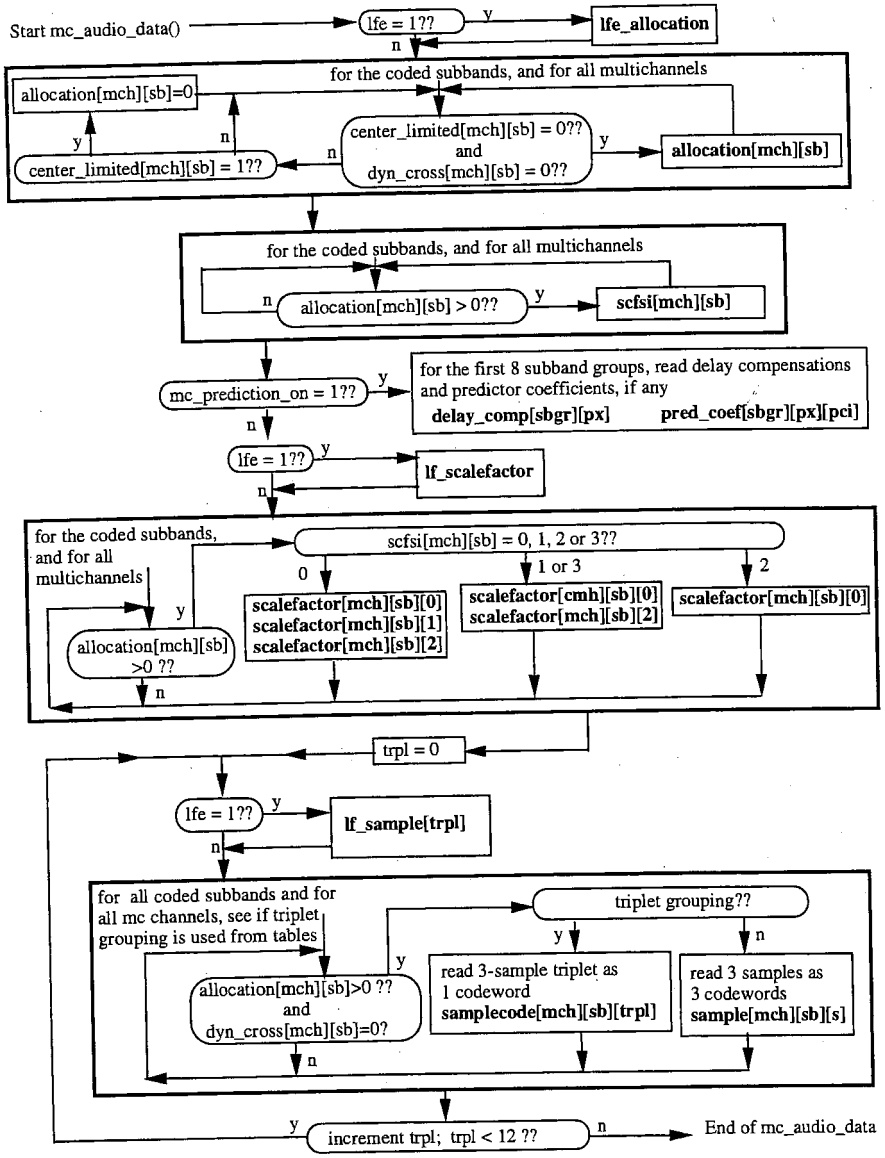


Fig. 4.14 Syntax for MPEG-2-BC **mc_audio_data( )** for Layers I and II. Each Frame represents 1152 samples of input PCM audio.

scalefactor[mlch][sb][b] (6 bits) Scale factor index for block b of subband sb of multilingual channel mlch. The actual scale factor is taken from tables.

samplecode[mlch][sb][trpl] (5 to 10 bits) Code word for three samples in triplet trpl of subband sb of multilingual channel mlch.

sample[mlch][sb][s] (2 to 16 bits) Coded representation of the sample s of subband sb of multilingual extension channel mlch.

This ends the description of MPEG-2 Layers I and II.

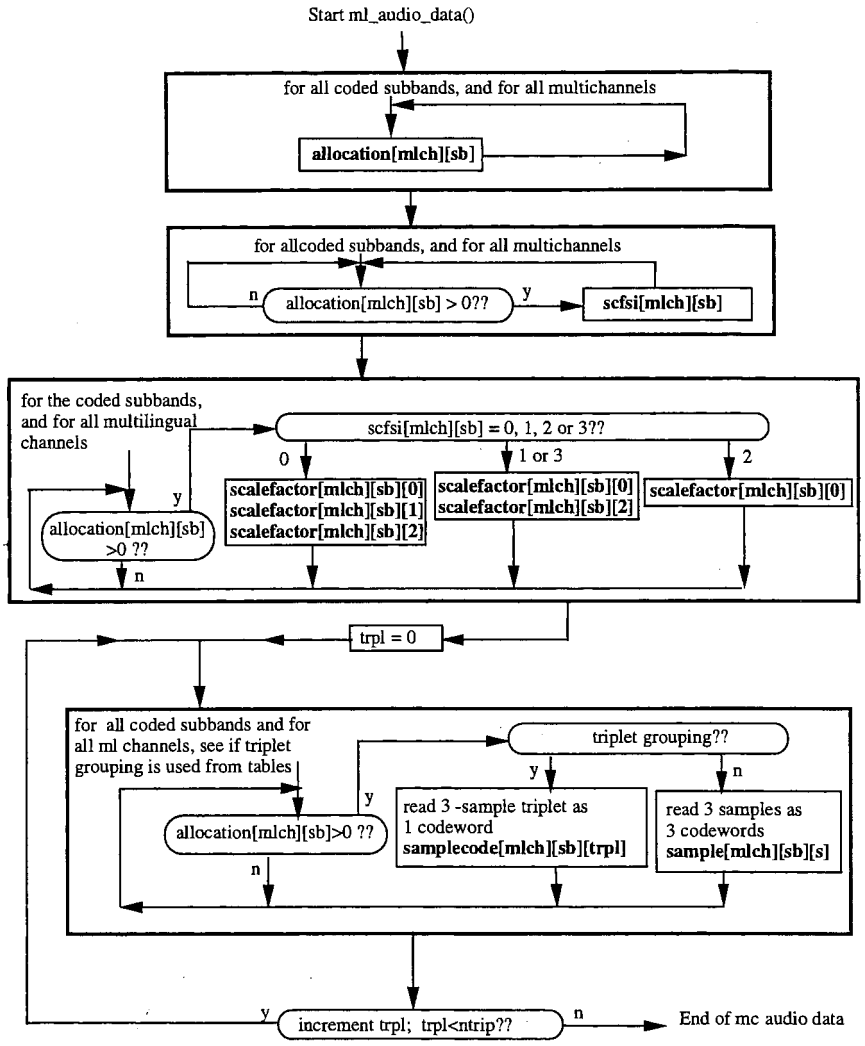


Fig. 4.15 Syntax for MPEG-2-BC **ml_audio_data( )** for Layers I and II. Each Frame represents 1152 samples of input PCM audio.

The syntax and semantics of Layer III is more complex, reflecting its greater coding efficiency. Space does not permit its description here.

4.4.9 MPEG-2 Non Backward Compatible (NBC)

Providing backward compatibility with MPEG-1 necessarily places a constraint on the quality that can be achieved in the compression of five-channel audio. Moreover, the quality may not be easily improved by increasing the bitrate. For this reason, there is currently underway an effort to standardize a Non Backward Compatible audio compression algorithm that can achieve state-of-the-art audio quality with five channels. We designate it as MPEG-2-NBC audio.

Two algorithms have already been found to have better quality than MPEG-2-BC. They are

the AC-3 system[11] from Dolby and the PAC algorithm[12] from AT&T.

REFERENCES

1. P. Noll, "Wideband Speech and Audio Coding," IEEE Commun. pp. 34–44. (November 1993). See also P. Noll, "Digital Audio Coding for Visual Communications," Proc. IEEE, pp. 925–943 (June 1995).
2. ISO/IEC 11172–3, "Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbits/s-Part 3: Audio. ISO/IEC JTC 1/SC 29. See also D. Pan, "A Tutorial on MPEG/Audio Compression," IEEE Multimedia, pp. 60–74 (Summer 1995).
3. ISO/IEC 13818–3, "Generic Coding of Moving Pictures and Associated Audio Information—Part 3: Audio." ISO/IEC JTC 1/SC 29.
4. A. Hoogendorn, "Digital Compact Cassette," Proc. IEEE 82:554–563 (April 1994).
5. T. Yoshida, "The Rewritable MiniDisc System," Proc. IEEE 82:1492–1500 (October 1994).
6. N. S. Jayant and P. Noll, Digital Coding of Waveforms, Prentice Hall, Englewood Cliffs, NJ (1984).
7. P. P. Vaidyanathan, Multirate Systems and Filter Banks, Prentice Hall, Englewood Cliffs, NJ (1993). See also, "Polyphase Quadrature Filters—A New Subband Coding Technique," Proc. Int'l Conf. IEEE ASSP, IEEE Press, New York, pp. 1280–1283 (1984).
8. H. S. Malvar, Signal Processing with Lapped Transforms, Artech House, Norwood, MA (1992).
9. J. D. Johnston and A. J. Ferreira, "Sum-Difference Stereo Transform Coding," Proc. IEEE ICASSP '92, II:569–572.
10. "Multichannel Stereophonic Audio System With and Without Accompanying Picture," ITU-R Recommendation 775, (November 1992).
11. "Digital Audio Compression (AC-3) Standard," published by the United States Advanced Television Systems Committee, April 12, 1995. Also, C. Todd et al., "AC-3: Flexible Perceptual Coding for Audio Transmission and Storage," AES 96th Convention (February 1994).
12. J. Johnston et al., "The AT&T Perceptual Audio Coder (PAC)," Proceedings of the American Acoustic Society Conference, New York (October 1995).
13. A. Hoogendorn, "Digital Compact Cassette," Proc. IEEE 82:554–563 (April 1994).
14. T. Yoshida, "The Rewritable MiniDisc System," Proc. IEEE 82:1492–1500 (October 1994).

5

Video Basics

Before we actually discuss video compression, and in particular, MPEG video compression, we need to understand the various aspects of the video signal itself and several related issues. Simply speaking, to understand how to compress video, we must have some understanding of how a video signal is generated, how the human visual system processes visual information, how a video signal is represented for storage[8] or transmission,[6] and how it can be converted between the various representations. We will certainly provide answers to these questions and much more. Before you begin wondering what this has to do with MPEG, remember that MPEG video compression is performed on color video signals represented in digital format. As explained in earlier chapters, digital format allows case of manipulation as well as robust storage[8,9] and transmission. Also, depending on the needs of an application and the bandwidth available, one of many video representations may be used in MPEG compression, which may be different from the representations made available by the camera used for imaging the scene. Furthermore, the video compressed by MPEG is eventually decoded for display, and if the decoded video uses a different representation than the display can handle, then again conversion is needed.

First we discuss the process of video imaging with emphasis on video scanning methods used for TV and film; image aspect ratio (IAR), which controls display shape; and gamma, which controls the relationship between voltage and intensity. Second, we discuss the human visual system, properties of colored light such as hue and saturation, the coordinate system used for representation of color video signals, and the composite and component systems used in various TV systems. Third, since MPEG compression uses digital video signals, we briefly discuss the generation of digital video from analog video signals. Fourth, we discuss specific digital component formats that MPEG coding uses. Fifth and sixth, we describe the various conversions[10,11] between the video formats used by MPEG; typically these conversions are needed in preprocessing prior to MPEG encoding or postprocessing after MPEG decoding. The last section discusses a unique form of conversion called mixing of video, which has potential for use in MPEG-1 and MPEG-2 applications involving overlaid text and graphics, as well as future MPEG based coding.

5.1 VIDEO IMAGING

To begin with, imaging is a process by which a representation of an actual scene is generated. Imaging can be of various types, such as normal photography, X-rays, electronic documents, electronic still pictures, motion pictures, and TV. We are obviously interested in the kind of imaging that is

time varying as it can adequately represent the motion in a scene. Video can be thought of as comprised of a sequence of still pictures of a scene taken at various subsequent intervals in time. Each still picture represents the distribution of light energy and wavelengths over a finite size area and is expected to be seen by a human viewer. Imaging devices such as cameras allow the viewer to look at the scene through a finite size rectangular window. For every point on this rectangular window, the light energy has a value that a viewer perceives which is called its intensity. This perceived intensity of light is a weighted sum of energies at different wavelengths it contains.

The lens of a camera focuses the image of a scene onto a photosensitive surface of the imager, which converts optical signals into electrical signals. There are basically two types of video imagers: (1) tube-based imagers such as vidicons, plumbicons, or orthicons, and (2) the more recent, solid-state sensors such as charge-coupled devices (CCD). The photosensitive surface of the tube imager is typically scanned with an electron beam or other electronic methods, and the two-dimensional optical signal is converted into an electrical signal representing variations of brightness as variations in voltage. With CCDs the signal is read out directly.

5.1.1 Raster Scan

Scanning is a form of sampling of a continuously varying two-dimensional signal. Raster scanning[1–3] is the most commonly used spatial sampling representation. It converts a two-dimensional image intensity into a one-dimensional waveform. The optical image of a scene on the camera imager is scanned one line at a time from left to right and from top to bottom. The imager basically converts the brightness variations along each line to an electrical signal. After a line has been scanned from left to right, there is a retracing period and scanning of next line again starts from left to right. This is shown in Fig. 5.1.

In a television camera, typically an electron beam scans across a photosensitive target upon which light from a scene is focused. The scanning spot itself is responsible for local spatial averaging of the image since the measured intensity at a time

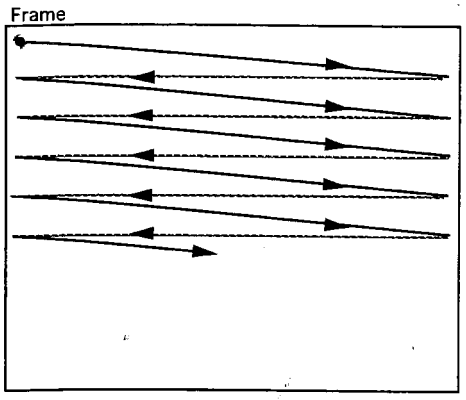


Fig. 5.1 Raster scan of an image

is actually a weighted average over the area of the spot. With the use of electron beam scanning, however, it is very hard to control the scanning spot filter function, with the result that the image is sometimes not sufficiently prefiltered. Optical scanning devices that use a beam of light are significantly better in this regard. Prefiltering can also be done in most cameras by a slight defocusing of the optical lens. With raster scanned display devices, filtering in the vertical direction is done by the viewer's eye or by adjustment of the scanning beam or optical components.

The smallest detail that can be reproduced in a picture is about the size of a picture element, typically referred to as a *pixel* (in computer terminology) or a *pel* (in video terminology). A sampled TV raster, for example, can be described as a grid of pels as shown in Fig. 1.2. The detail that can be represented in the vertical direction is limited by the number of scan lines. Thus, some of the detail in vertical resolution is lost as the result of raster scanning fall between the scan lines and cannot be represented accurately. Measurements indicate that only about 70% of the vertical detail is actually represented by scan lines. Similarly, some of the detail in the horizontal direction is lost owing to sampling of each scan line to form pels.

5.1.2 Interlace Raster Scan

The choice of number of scan lines involves a tradeoff among contradictory requirements of

bandwidth, flicker, and resolution. Interlaced[1–3,5–6] scanning tries to achieve these tradeoffs by using frames that are composed of two *fields* sampled at different times, with lines of the two fields interleaved, such that two consecutive lines of a frame belong to alternate fields. This represents a vertical–temporal tradeoff in spatial and temporal resolution. For instance, this allows slow-moving objects to be perceived with higher vertical detail while fast-moving objects are perceived at a higher temporal rate, albeit at close to half vertical resolution. Thus the characteristics of the eye are well matched because, with slower motion, the human visual system is able to perceive the spatial details, whereas with faster motion spatial details are not as easily perceived. This interlace raster scan is shown in Fig. 5.2.

From a television transmission and display aspect, the interlace raster scan represents a reasonable tradeoff in the bandwidth necessary and the amount of flicker generated. It is worth noting that the tradeoffs that may have been satisfactory for television may not be satisfactory for computer displays, owing to the closeness of the screen to the viewer and the type of material typically displayed, that is, text and graphics. Thus, if interlaced TV displays were to be used with computers, the result would be an annoying large area flicker, interline flicker, line crawling, and other problems. To avoid these problems, computer displays use noninterlaced (also called progressive or sequential) displays with refresh rates of higher than 60 frames/s, typically 72 frames/s.

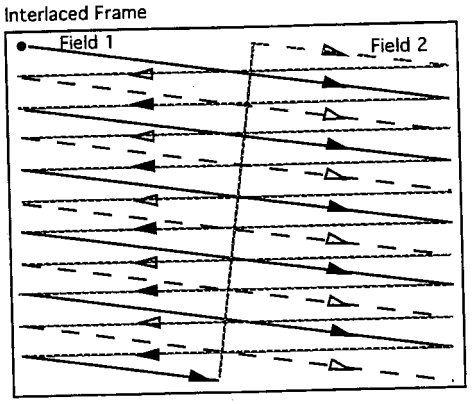


Fig. 5.2 Interlaced raster scan of an image

Table 5.1 Equivalent Line Number of Film and TV Image Formats

Image Formats	Equivalent Line Number Range
NTSC TV	175 to 310
16 mm Film	200 to 300
35 mm Film	350 to 450
HDTV	300 to 515

5.1.3 Film versus Television

Originally, television was designed to capture the same picture definition as that of 16-mm film. Actually, the resolution of 16-mm film is significantly higher than that of normal TV, but the perceived sharpness is about the same owing to a better temporal response of normal TV. To compare the performance of quite different display systems, the concept of equivalent line number is useful. In Table 5.1 we show the equivalent line numbers for film and television.

Based on what we have said so far, it may appear that a television system can achieve the performance of film. In practice, however, there are too many ways in which the performance of a TV system can become degraded. In this context, it is worth noting that *frequency response,* which is the variation in peak-to-peak amplitude of a pattern of alternate dark and bright lines of varying widths, can be used to establish the performance of an imaging system. In reality, TV and film images never really appear the same owing to the difference in shape of the overall frequency response. For instance, the frequency response of TV drops much faster, whereas the frequency response of film decays more gradually.

The response of the human eye over time is generally continuous,[1–3] that is, nerve impulses travel from the eye to the brain at the rate of about 1000 times per second. Fortunately, television does not need to provide new images of a scene at this rate as it can exploit the property of persistence of vision to retain the sensation of an image for a time interval even though the actual image may have been removed. Both TV and motion pictures employ the same principle—a scene is portrayed by showing a sequence of slightly different images (frames) taken over time to replicate the motion in a scene. For motion in

cene to appear realistic, the rate of imaging of rames should be high enough to capture the notion without jerkiness, and the rate of display hould be high enough so that the captured notion can be replicated smoothly. The display ate must also be high enough so that persistence f vision works properly. One of the nuances of uman vision is that if the images being presented o it are bright, the persistence of vision is actually horter, thus requiring a higher repetition rate for lisplay.

In the early days of motion pictures,[1–3] it was ound that for smooth motion rendition, a mini- ıum frame rate of 15 per second (15 frames/s) /as necessary. Thus old movie equipment, includ- ıg home movie systems, use 16 frames/s. Howev- r, because of a difficulty in representing action in owboy movies of those days, the motion picture ıdustry eventually settled on the rate of 24 ·ames/s, which is used to this day. Note that lthough 24 frames/s movies use 50% more film ıan 16 frames/s movies, the former was adopted s a standard as it was considered worthwhile. As ıovie theater projection systems became more owerful, the nuance of eye mentioned earlier, hich requires a higher frame rate, began to come ıto play, resulting in the appearance of flicker. A ɔlution that did not require a change in the cam- :a frame rates of movies was to use a 2 blade ɔtating shutter in the projector, synchronized to ıe film advance, which could briefly cut the light ıd result in double the displayed frame rate to 48 ames/s. Today, even a 3 blade shutter to effec- vely generate a display rate of 72 frames/s is ımetimes used since projection systems have got- n even more brighter.

The frame rate used for monochrome TV in urope was in fact chosen to be slightly higher .an that of film at 25 frames/s, in accordance ith electrical power there, which uses 50 cycles/s Iz) rate. Television in Europe uses systems such as AL and SECAM, each employing 25 frames/s. It common practice in Europe to run 24 frames/s ovies a little faster at 25 frames/s. The PAL and ECAM based TV systems are currently in wide- read use in many other parts of the world as ell.

The frame rate of monochrome TV in the United States was chosen as 30 frames/s, primarily to avoid interference to circuits used for scanning and signal processing by 60-Hz electrical power. In 1953, color was added to monochrome by migration to an NTSC system in which the 30 frames/s rate of monochrome TV was changed by 0.1 percent to 29.97 frames/s owing to the requirement of precise separation of the video signal carrier from the audio signal carrier. Incidentally Japan and many other countries also employ the NTSC TV system.

For reasons other than those originally planned, the frame rate chosen in the NTSC TV system turned out to be a good choice since there is a basic difference in the way TV is viewed as compared to film. While movies are generally viewed in dark surroundings that reduce the human eye's sensitivity to flicker and the smoothness of motion, TV is viewed under condition of surround lighting, in which case the human visual system is more sensitive to flicker and the smoothness of motion. Thus, TV requires a higher frame rate than movies to prevent the appearance of flicker and motion-related artifacts.

One very important point is that unlike TV, which uses interlace, film frames are sequential (also called noninterlaced or progressive).

5.1.4 Image Aspect Ratio (IAR)

The image aspect ratio is generally defined as the ratio of picture width to height* and impacts the overall appearance of the displayed image. For example, the IAR of normal TV images is 4/3, meaning that the image width is 1.33 times the image height. This value was adopted for TV, as this format was already used and found acceptable in the film industry at that time, prior to 1953. However, since then the film industry has migrated to wide-screen formats with IARs of 1.85 or higher. Since subjective tests on viewers show a significant preference for a wider format than that used for normal TV, HDTV uses an IAR of 1.78, which is quite close to that of the wide-screen film format. Currently, the cinemascope film format provides one of the highest IARs of 2.35. Table 5.2

*Some authors, including MPEG, use height/width.

Table 5.2 Image Aspect Ratio (IAR) of Film and TV Image Formats

Image Formats	Aspect Ratio
NTSC, PAL and SECAM TV	1.33
16 mm and 35 mm Film	1.33
HDTV	1.78
Widescreen Film	1.85
70 mm Film	2.10
Cinemascope Film	2.35

compares aspect ratios of a variety of formats used in film and TV.

Sometimes there may be a mismatch such that an image that was intended for display at one IAR may have to be shown on a display of a different IAR. Figure 5.3 shows images of a 4:3 IAR on a 16:9 display and vice versa.

In the discussion of aspect ratio the concept of pel aspect ratio (PAR) is often quite useful. Although pels have no intrinsic shape of their own, under some restrictions, they can be thought of as contiguous rectangles as shown in Fig. 1.2. MPEG-1 video allows specification of PAR through the parameter *pel_aspect_ratio*, a 4-bit code representing several discrete values of PAR.

MPEG-2 video interprets the same 4-bit code as *aspect_ratio_information*, and uses it to either specify that pels in a reconstructed frame are square (PAR = 1) or it specifies one of three values of IAR (see Sec. 10.2.2). When IAR is specified, there are two cases, one in which the reconstructed image is to be shown on the entire display, and another in which, depending on the display size and coded image size, either a portion of the reconstructed image is displayed or a full image is displayed on a portion of the screen.

In the first case, PAR can be calculated as follows:

$$\text{PAR} = \text{IAR} \times \text{vertical\_size}/\text{horizontal\_size}$$

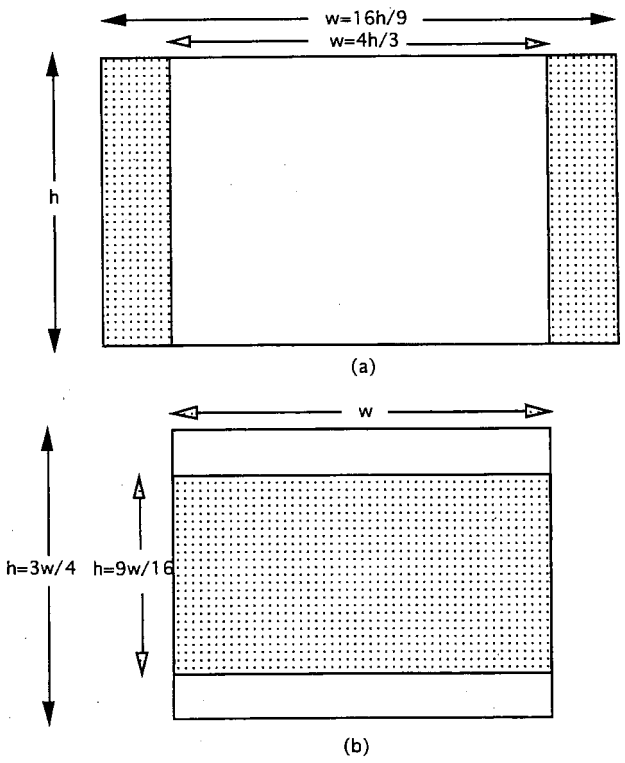


Fig. 5.3 Image aspect ratio (IAR) =w/h.

where horizontal_size is the width of the image in terms of pels and the vertical_size is the height of the image in lines. As an example consider a 4:3 IAR display, and suppose the image width is 720 pels and the image height is 486 lines. Then PAR can be calculated as follows.

$$\text{PAR} = {}^{4}/_{3} \times {}^{486}/_{720} = 0.9$$

In the second case, PAR is calculated as follows:

$$\text{PAR} = \text{IAR} \times \text{display\_vertical\_size}/\text{display\_horizontal\_size}$$

This second case is signalled via *sequence_display_extension( )* which carries information regarding *display_horizontal_size* and *display_vertical_size* which determine the size of the display rectangle. When this display rectangle is smaller than the image size it implies that only a portion of image is to be displayed, and conversely when the display rectangle is larger than the image size it implies that the image is displayed only on a portion of the display screen.

It is worth noting that computer displays use square pels (PAR = 1.0) and if material intended for TV is shown on a computer display, correction of the pel aspect ratio is necessary; otherwise, circles on a TV screen appear as ellipses on a computer display.

5.1.5 Gamma

Many TV cameras and all CRT-based displays have a nonlinear relationship between signal voltage and light intensity. The light intensity input to the camera or output by the display is proportional to the voltage raised to the power gamma and is given by the following generalized relationship:

$$B = cv^{\gamma} + b$$

where B is the light intensity, v is voltage, c is a gain factor, b is either cutoff (camera) or black level (cathode ray tube, CRT) light intensity, and γ typically takes values in the range of 1 to 2.5, although for some picture displays it can be as high as 3.0.

Typically, camera picture tubes or CCD sensors have a gamma of about 1 except for a vidicon type picture tube, which has a gamma of 1.7. The CRT displays use kinescopes, which typically have a gamma of 2.2 to 2.5.

To avoid gamma correction circuitry inside millions of TV receivers, gamma correction is done prior to transmission. For example, assuming the gamma of the camera to be 1 and that of the display to be 2.2, then the camera voltage is raised to a power of ${}^{1}/_{2.2} = 0.45$.

The result of gamma correction is that a gamma-corrected voltage is generated by the camera that can be transmitted and applied directly to the CRT. As a side benefit, the gamma-corrected voltage is also less susceptible to transmission noise.

In normal TV transmission, the signal from the camera is gamma corrected, modulated, and transmitted to the receiver as a radiofrequency (RF) signal, which is received by the receiver's antenna and demodulated and applied to the display. The main components of such a TV transmission system are shown in Fig. 5.4.

In our discussion thus far, we have considered only monochrome TV and have deferred a discussion about color TV so that we could introduce

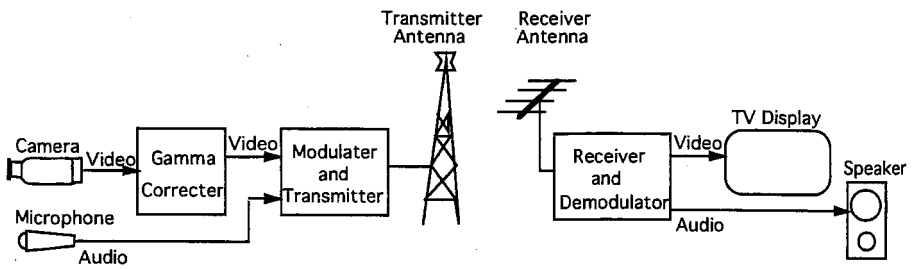


Fig. 5.4 Monochrome television analog transmission system

some aspects of human vision and how they are exploited in the design of a color TV system.

5.2 THE HUMAN VISUAL SYSTEM AND COLOR TV

With the background on video imaging developed thus far, we are now well prepared to delve into the intricacies of the human visual system, the perception of color, the representation of color, and further to specific color TV systems.

5.2.1 The Human Visual System

The human eye is a remarkably intricate system; its anatomy is shown in Fig. 5.5.

Incoming light bouncing off different objects is refracted by the cornea toward the pupil. The pupil is the opening of the iris through which all light enters the eye. The light again is refracted by the lens onto the back of the eyeball, forming an image on the retina. The retina consists of receptors sensitive to light called photoreceptors connected by nerve cells. To reach these photoreceptors, light must first pass through nerve cells. The photoreceptors contain chemical pigments that can absorb light and initiate a neural response. There are two types of photoreceptors, rods and cones. Rods are responsible for low light vision, while cones are responsible for details and color under normal light conditions, as in daylight. Both rods and cones enable vision when the amount of light is between the two extremes. Light absorbed by photoreceptors initiates a chemical reaction that bleaches the pigment, which reduces the sensitivity to light in proportion to amount of pigment bleached. In general, the amount of bleached pigment rises or falls depending on the amount of light. The visual information from the retina is passed via optic nerves to the brain. Beyond the retina, processing of visual information takes place en route to the brain in areas called the lateral geniculate and the visual cortex. Normal human vision is binocular, composed of a left-eye and a right-eye images. The left-eye image is processed by the right half of the brain, and the right-eye image by the left half of brain. We further discuss the intricacies of the human visual system, and in particular as it refers to binocular vision, in Chapter 15.

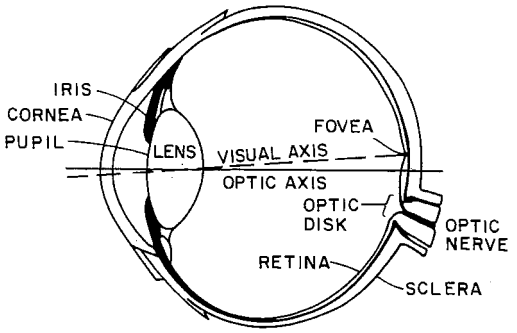


Fig. 5.5 Anatomy of the human eye

Light of various wavelengths produces a sensation called *color.* Different distributions generally result in the perception of different colors. A color is thus visible energy of a given intensity and of a given composition of wavelengths. The normal human visual system can identify differences in sensation caused by different colors and thus colors can be classified. In our earlier discussion of the human visual system we mentioned that the retina of the human eye contains photoreceptors called cones that are responsible for color vision. The human retina contains a color-sensitive area that consists of three sets of cones that are individually sensitive to red, green, and blue light. Thus the sensation of color experienced directly depends on the relative amounts of red, green, and blue light; for instance, if only blue and red cones are excited, the sensation is magenta. When different combinations of red, blue, and green cones are excited, different colors are sensed; a combination of approximately 30% red, 60% green, and 10% blue results in white light.

5.2.2 Hue and Saturation

The sensation known as color produced by visible light is also called *hue.* Visible light is electromagnetic radiation with a spectrum of wavelengths; a hue is determined by its dominant wavelength. Besides hue, *saturation* is another property of interest and represents the degree of

purity of a color. For instance, a pure spectral color having a single wavelength has a saturation of 100%, while the saturation of white or gray light is zero. The three basic properties of light—*brightness, hue,* and *saturation*—can be completely defined by its spectral distribution, a plot of amplitude versus wavelength. Based on our earlier discussion about the sensation of color caused by the response of three types of color-sensitive cones in the retina of human eye, it can be stated that the perceived hue and saturation of two colors is exactly the same if they invoke the same response from the color-sensitive cones in the retina. The trichromatic theory of color vision implies that hue and saturation of nearly any color can be duplicated by an appropriate combination of the three *primary* colors.

5.2.3 Color Primaries

One of the requirements in the choice of primary colors is that they are independent and chosen such as to represent the widest range of hues and saturation by appropriate combinations of the three primaries. There are basically two kinds of color primaries, one composed of subtractive primaries and the other composed of additive primaries. Subtractive primaries are used in printing, photography, and painting, whereas additive primaries are used in color television.

Although a given color can be matched by an additive mixture of three primaries, when different observers match the same color, slight differences in the amount of primary colors needed to match the color arise. In 1931, the CIE defined a standard observer and three standard primaries by averaging the color matching data of a large number of observers with normal vision. The CIE chromaticity diagrams enclose a horseshoe-shaped region in which each point represents a unique color and is identified by its x and y coordinates. Three such points identify primary colors, and the triangle formed by these points represents the gamut of colors that can be represented by the display using these primaries. For example, the FCC has defined three such points that represent the color primaries for the NTSC system. Likewise three such points are defined for the PAL system.

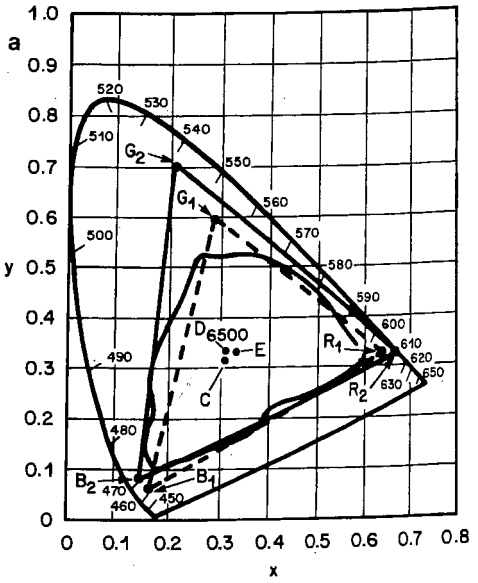


Fig. 5.6 Color space primaries for NTSC and PAL

The color space primaries for the NTSC and the PAL systems are shown in Fig. 5.6.

As mentioned earlier, in the United States in 1953 color was added to the monochrome TV system by migration to the NTSC system. This system has been adopted widely in North and South America, the Caribbean, and in parts of Asia. On the other hand the PAL and SECAM color TV systems originated in Europe and are in use in the remaining parts of the world. Although all three TV systems have basic differences in picture resolution and frame rates, they have considerable similarities as they all belong to the class of *composite systems.* A second class of TV systems is called *component systems* and thus far is used in most digital applications.

5.2.4 Composite Systems: NTSC, PAL, and SECAM

In composite systems, the luminance and chrominance signals are multiplexed within the same channel or on the same carrier signal. Composite formats[1,4,5,7] such as NTSC, PAL, or SECAM formats were designed to be compatible with the monochrome format. That is, mono-

chrome receivers could receive color signal transmission but would display it as a compatible monochrome signal. Likewise, monochrome transmission could be received by color receivers.

A camera imaging a scene generates for each pel the three color weights RGB, which may be further processed for transmission or storage. At the receiver, the three components are sent to the display, which regenerates the contents of the scene at each pel from the three color components. For transmission or storage between the camera and the display a luminance signal Y representing brightness and two chrominance signals representing color are used. The need for such a transmission system arose with NTSC, where compatibility with monochrome receivers required a black-and-white signal, which is now referred to as the Y signal. It is well known that the sensitivity of the human eye is highest to green light, followed by that of red, and the least to blue light. The NTSC system exploited this fact by assigning a lower bandwidth to the chrominance signals as compared to the luminance, Y, signal. This made it possible to save bandwidth without losing color quality. The PAL and SECAM systems also employ reduced chrominance bandwidths.

5.2.4.1 The PAL System

The YUV color space of PAL is employed in one form or another in all three color TV systems. The basic YUV color space can be generated from gamma-corrected RGB (referred to in equations as $R'G'B'$) components as follows:

$$Y = 0.299R' + 0.587G' + 0.114B'$$

$$U = -0.147R' - 0.289G' + 0.436B' = 0.492(B' - Y)$$

$$V = 0.615R' - 0.515G' - 0.100B' = 0.877(R' - Y)$$

The inverse operation, that is, generation of gamma-corrected RGB from YUV components, is accomplished by the following:

$$R' = 1.0Y + 1.140V$$

$$G' = 1.0Y - 0.394U - 0.580V$$

$$B' = 1.0Y - 2.030U$$

Although our primary interest is in the component color space of PAL, a brief discussion about how composite encoding is performed may be useful.

The Y, U, and V signals in PAL are multiplexed in a total bandwidth of either 5 or 5.5 MHz. With PAL, both U and V chrominance signals are transmitted with bandwidth of 1.5 MHz. A color subcarrier is modulated with U and V via Quadrature Amplitude Modulation (QAM) and the composite signal is limited to the allowed frequency band, which ends up truncating part of the QAM signal. The color subcarrier for PAL is located at 4.43 MHz. PAL transmits the V chrominance omponent as $+V$ and $-V$ on alternate lines. The demodulation of the QAM chrominance signal is similar to that of NTSC, which we will cover a bit more in detail. Here it suffices to say that the recovery of the PAL chrominance signal at the receiver includes averaging of successive demodulated scan lines to derive the U and V signals.

5.2.4.2 The NTSC System

The NTSC color space of YIQ can be generated from the gamma-corrected RGB components or from YUV components as follows:

$$Y = 0.299R' + 0.587G' + 0.114B'$$

$$I = 0.596R' - 0.274G' - 0.322B'$$
$$= -(\sin 33°)\,U + (\cos 33°)\,V$$

$$Q = 0.211R' - 0.523G' + 0.311B'$$
$$= (\cos 33°)\,U + (\sin 33°)\,V$$

The inverse operation, that is, generation of gamma-corrected RGB components from the YIQ composite color space, can be accomplished as follows:

$$R' = 1.0Y + 0.956I + 0.621Q$$

$$G' = 1.0Y - 0.272I - 0.649Q$$

$$B' = 1.0Y - 1.106I + 1.703Q$$

In NTSC, the Y, I, and Q signals are all multiplexed into a 4.2 MHz bandwidth. Although the Y component itself takes a 4.2 MHz bandwidth, multiplexing all three components into the same

4.2 MHz becomes possible by interleaving luminance and chrominance frequencies, without too much interference with one another. This is done by defining a color subcarrier at approximately 3.58 MHz. The two chrominance signals I and Q are QAM modulated onto this carrier. The envelope of this QAM signal is approximately the saturation of the color, and the phase is approximately the hue. The luminance and modulated chrominance signals are then added to form the composite signal. The process of demodulation first involves comb filtering (horizontal and vertical filtering) of the composite signal to separate the luminance and a combined chrominance signal followed by further demodulation to separate the I and Q components.

5.2.4.3 The SECAM System

The SECAM composite color space is $YDrDb$ and can be generated from gamma-corrected $R'G'B'$ components:

$$Y = 0.299R' + 0.587G' + 0.114B'$$

$$Db = -0.450R' - 0.883G' + 1.333B' = 3.059U$$

$$Dr = -1.333R' + 1.116G' - 0.217B' = -2.169V$$

The inverse operation, that is, generation of gamma-corrected RGB from $YDrDb$, can be accomplished as follows:

$$R' = 1.0Y - 0.526Dr$$

$$G' = 1.0Y - 0.129Db + 0.268Dr$$

$$B' = 1.0Y + 0.665Db$$

In the SECAM system, the video bandwidth is 6 MHz. The chrominance signals are transmitted on alternate lines by use of Frequency Modulation (FM) of the two color carriers which are located at 4.25 MHz and 4.41 MHz. Owing to the higher allocated bandwidth of 6 MHz, the demodulation of chrominance signals is much simpler than with QAM used in PAL and NTSC. Further, because of FM, gain distortion of the signal is somewhat less.

5.2.4.4 Comparison of NTSC, PAL, and SECAM

In Table 5.3 we provide a summary of the TV parameters used in the composite systems. Composite color TV systems resulted from the necessity of fitting extra color information in the same bandwidth as that used by the monochrome TV system and maintaining compatibility with monochrome TV receivers. Although the tradeoffs made were quite reasonable, understandably, some artifacts resulted. One of the disadvantages of the composite system is that for some pictures there is crosstalk between luminance and chrominance signals that gives rise to cross-color and cross-luminance artifacts. Cross-color artifacts result when the luminance signal contains frequencies near the subcarrier frequency resulting in spurious color, whereas cross-luminance artifacts are caused by a change of the luminance phase from field to field near edges of objects in a scene due to imperfect cancellation of the subcarrier.

5.2.5 The Component System

In a component TV system, the luminance and chrominance signals are kept separate, such as on separate channels or at different times. Thus the component TV system is intended to prevent the crosstalk that causes cross-luminance and cross-

Table 5.3 Summary of parameters used in worldwide Color TV Standards

Parameter	NTSC	PAL	SECAM
Field Rate (Hz)	59.94	50	50
Lines per Frame	525	625	625
Gamma	2.2	2.8	2.8
Audio Carrier (MHz)	4.5	QAM	FM
Color Subcarrier (MHz)	3.57	4.43	4.25 (+U), 4.4 (−V)
Color Modulation Method	QAM	QAM	FM
Luminance Bandwidth (MHz)	4.2	5.0, 5.5	6.0
Chrominance Bandwidth (MHz)	1.3 (I), 0.6 (Q)	1.3 (U), 1.3 (V)	>1.0 (U), > 1.0 (V)

chrominance artifacts in the composite systems. The component system is preferable in all video applications that are without the constraints of broadcasting, such as studio, digital storage media, and so forth.

Although any of the previously discussed sets of component signals[1,4,5,7] can be used, of particular significance is the CCIR-601* digital component video format. The color space of this format is *YCrCb* and is obtained by scaling and offsetting the *YUV* color space. The conversion from gamma-corrected *RGB* components represented as 8-bits (0 to 255 range) to *YCrCb* is specified as follows:

$$Y = 0.257R' + 0.504G' + 0.098B' + 16$$

$$Cr = 0.439R' - 0.368G' - 0.071B' + 128$$

$$Cb = -0.148R' - 0.291G' + 0.439B' + 128$$

In these equations, Y is allowed to take values in the 16 to 235 range whereas Cr and Cb can take values in the range of 16 to 240 centered at a value of 128, which indicates zero chrominance.

The inverse operation, that is, generation of gamma-corrected *RGB* from *YCrCb* components, can be accomplished as follows:

$$R' = 1.164(Y - 16) + 1.596(Cr - 128)$$

$$G' = 1.164(Y - 16) - 0.813(Cr - 128) - 0.392(Cb - 128)$$

$$B' = 1.164(Y - 16) + 2.017(Cb - 128)$$

In converting *YCrCb* to gamma-corrected *RGB*, the *RGB* values must be limited to lie in the 0 to 255 range as occasional excursions beyond nominal values of *YCrCb* may result in values outside of the 0 to 255 range.

The sampling rates for the luminance component Y and the chrominance components are 13.5 MHz and 6.75 MHz, respectively. The number of active pels per line is 720, the number of active lines for the NTSC version (with 30 frames/s) is 486 lines and for the PAL version (with 25 frames/s) is 576 lines. Using 8-bit per sample representation, the digital bandwidth of the uncompressed CCIR-601 signal can be calculated as 216 Mbit/s.

5.2.6 Color TV

We are now ready to discuss the color TV system[1–7] excluding details of transmission for simplicity. Fig. 5.7 shows both a composite color TV system and a component color TV system. The color space used in each case is referred to by the generalized notation of *YC1C2* and could be one of the various color spaces discussed.

As shown in Fig. 5.7, the RGB analog signal from the color camera undergoes gamma correction, resulting in a gamma corrected version

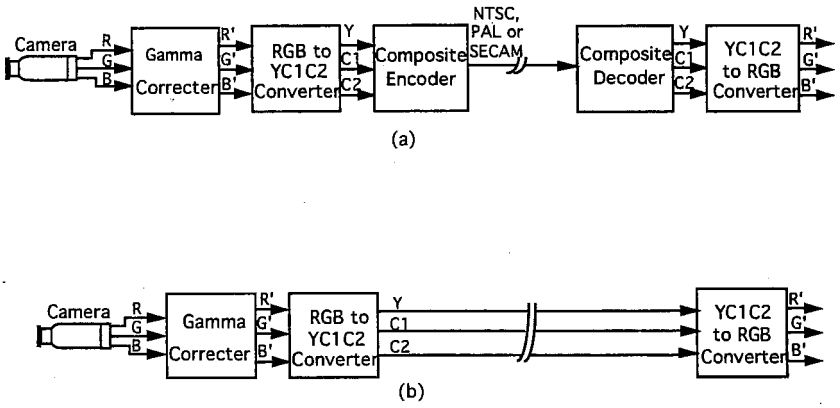


Fig. 5.7 Composite and component analog TV systems

*CCIR is now ITU-R.

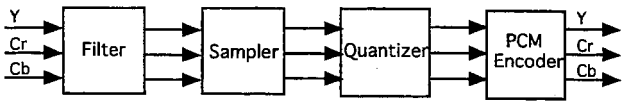


Fig. 5.8 Conversion of component analog TV signals to digital TV signals.

$R'G'B'$, which further undergoes color space conversion from $R'G'B'$ to $YC1C2$ depending on what $C1$ and $C2$ are (e.g., I and Q, U and V or something else) using the equations discussed. If the color TV system is composite, the three signals, Y, $C1$, and $C2$, undergo modulation encoding to generate a composite signal that is transmitted and at receiver undergoes the inverse operation from a single composite signal to three component signals and through $YC1C2$ color conversion back to $R'G'B'$ which is applied to display. On the other hand, in a component color TV system, the three signals are directly sent to the receiver without any composite encoding and are converted to $R'G'B'$ for display.

It is worth noting that we have assumed that the camera generates *RGB* and that the display uses the *RGB* signal. However, most home TV receivers directly take a composite signal, and any color conversions may be internal. Likewise, in nonbroadcast applications with inexpensive cameras, only a composite signal may be available at the camera output.

5.3 FROM ANALOG TO DIGITAL VIDEO

In the previous section we mentioned CCIR-601 component color signals in digital format and the digital bandwidth required. While it was not necessary to delve into how a digital representation can be obtained, now is probably a good time to do so. Furthermore, we will oversimplify the facts about digital conversion of analog video signal rather than get into detailed mathematical analysis.

The process of digitization of video consists of operations[1–7] of Prefiltering, Sampling, Quantization, and Encoding. To make our discussion more concrete, we will assume that we are dealing with three signals that form a CCIR-601 video format, that is, Y, Cr, and Cb, and that these signals undergo the steps we just listed. The sequence of operations is illustrated in Fig. 5.8.

5.3.1 Filtering

The filtering operation employed here is also referred to as prefiltering, as this operation is performed prior to sampling. Prefiltering can reduce the unwanted excessive frequencies in the signal as well as noise in the signal. The simplest filtering operation involves simply averaging of the image intensity within a small area around the point of interest and replacing the intensity of the original point by the computed averaged intensity. Prefiltering can sometimes be accomplished by controlling the size of the scanning spot in the imaging system.*

In dealing with video signals, the filtering applied on the luminance signal may be different than that applied on chrominance signals owing to different bandwidths required.

5.3.2 Sampling

Next, the filtered signal is sampled at a chosen number of sampling points. The sampling operation can be thought of as a switch that closes at uniform duration and senses the value of the amplitude of the input signal, after which it opens again awaiting closure at the next appropriate time. The minimum rate at which an analog signal must be sampled is called the Nyquist rate and corresponds to twice that of the highest frequency in the signal. For NTSC system this rate is $2 \times 4.2 = 8.4$ MHz and for PAL this rate is $2 \times 5 = 10$ MHz. It is normal practice to sample at a rate higher than this for ease of signal recovery. The CCIR-601 signal employs 13.5 MHz for luminance and half of that rate for chrominance signals. This rate is an inte-

*For CCD cameras, electrical filters are used.

gral multiple of both NTSC and PAL line rates but is not an integral multiple of either NTSC or PAL color subcarrier frequency.

5.3.3 Quantization

The sampled signal, which is now simply a sequence of pel values, is quantized next. The quantization operation here means that when the input pel value is in a certain range of values, we assign and output a single fixed value representing that range. Thus a discretized representation of the pel values can be obtained. The process of quantization results in loss of information since the quantized output pel value is less accurate than the input value. The difference between the value of the input pel and its quantized representation is called quantization error. The choice of the number of levels of quantization involves a tradeoff of accuracy of representation and the visibility of noise in the signal due to quantization errors. The process of quantization is depicted in Fig. 5.9.

5.3.4 PCM Coding

The last step in analog to digital conversion is encoding of quantized values. Here, by encoding we simply mean obtaining a digital representation. The type of encoding that we are interested in is called PCM, Pulse Code Modulation. Often video pels are represented by 8-bit PCM codewords, which means that with this type of coding, it is possible to assign to the pel one of the $2^8 = 256$ possible values in the range of 0 to 255. For example, if the quantized pel amplitude is 68, the corresponding 8-bit PCM codeword is the sequence of bits 01000100 which obviously stands for $1 \times 2^6 + 1 \times 2^2 = 68$.

5.3.5 Digital Component Video

In this section we briefly introduce the various digital component video formats[10,11] typically used in MPEG video coding. MPEG video coding makes some assumptions about the color spaces as well as format of luminance and chrominance component signals, although it is quite flexible about the size of pictures themselves. The video resolutions and formats typically used at normal TV or lower resolution are all derived from the CCIR-601 format. We will discuss actual derivation of various formats in the next section; for now, we focus only on what each of the formats looks like.

5.3.6 The 4:2:0 Format

By the 4:2:0 video format we do not imply a specific picture resolution, only the relative rela-

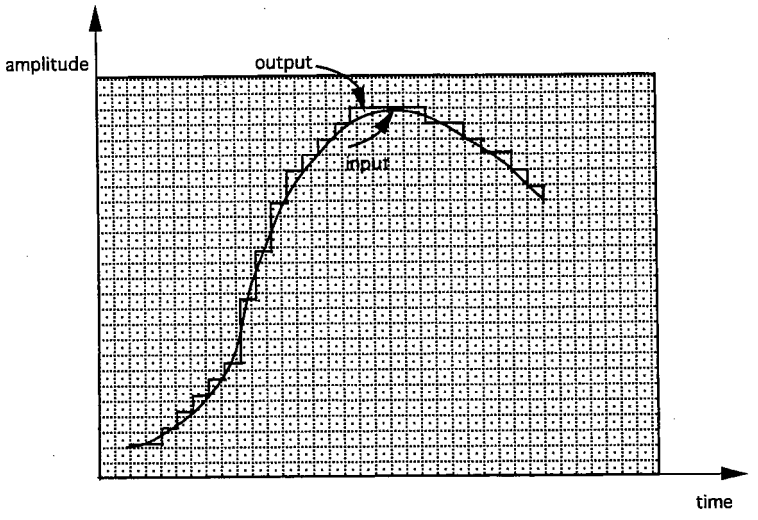


Fig. 5.9 Quantization of analog input to discrete output.

tionship between the luminance and chrominance components. In the MPEG-2 4:2:0 video format, in each frame of video, the number of samples of each chrominance component, *Cr* or *Cb*, is one-half of the number of samples of luminance, both horizontally and vertically. Alternatively, for every four samples of luminance forming a 2 × 2 array, two samples of chrominance, one a *Cr* sample and the other a *Cb* sample, exist. The exact location of chrominance samples with respect to luminance samples in a frame of 4:2:0 video is shown in Fig. 5.10.

MPEG-1 also uses the 4:2:0 video format, although there is a slight difference in location of the chrominance samples. In MPEG-1 the 4:2:0 video format has chrominance samples located midway between all four samples of luminance. During MPEG-2 video development, this slight change in location of MPEG-2 4:2:0 chrominance from the location of MPEG-1 4:2:0 chrominance was needed to maintain a consistent definition of 4:2:0, 4:2:2, and 4:4:4 formats employed by MPEG-2. As a matter of interest, the 4:2:0 format is sometimes referred to in the outside literature as the 4:1:1 format, since there is one *Cb* and one *Cr* sample per four luminance samples. However, this use should be avoided to reduce confusion owing to the existence of an older 4:1:1 format that uses 4:1 reduction of chrominance resolution in the horizontal direction only, but full resolution in the vertical direction.

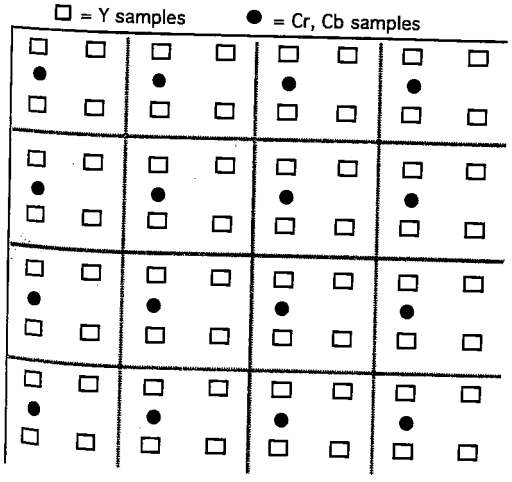


Fig. 5.10 Luminance and chrominance samples in a 4:2:0 video frame

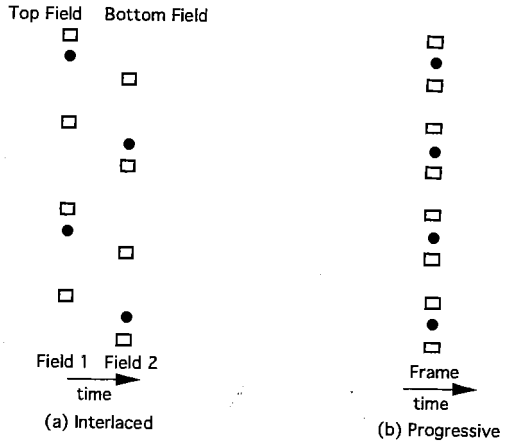


Fig. 5.11 Interlaced and Progressive representation of a 4:2:0 frame.

We now illustrate the vertical and temporal positions of samples in 4:2:0 by showing what amounts to the side view of a 4:2:0 frame, in Fig. 5.11. It is worth mentioning that a 4:2:0 frame may be interlaced or noninterlaced. The difference was pointed out earlier during our discussion on raster scanning—an interlaced frame is composed of lines of samples belonging to two different fields separated in time but taken together to form a frame, whereas in a noninterlaced or progressive frame there is no such separation in time between consecutive lines of samples.

Because of the special significance of 4:2:0 formats to MPEG-1 and MPEG-2 video coding, we now provide examples of specific image frame sizes used. The *Source Input Format* (SIF) is noninterlaced and uses image size of either 352 × 240 with 30 frames/s* or 352 × 288 with 25 frames/s and are correspondingly referred to as SIF-525 or SIF-625, since the 240-line format is derived from the CCIR-601 video format of 525 lines whereas the 288-line format is derived from the CCIR-601 video format of 625 lines. The two versions of CCIR-601 correspond to the two TV systems

*More precisely, 29.97 frames/s.

NTSC and PAL. The SIF format is typically used in MPEG-1 video coding. An interlaced version of SIF is called SIF-I and uses the same image size as SIF, that is either 352 × 240 with 30 frames/s or 352 × 288 with 25 frames/s and is used in layered MPEG-2 video coding (Chapter 9), although it is not as popular.

An interlaced video format that is somewhat more popular in MPEG-2 video coding is Half Horizontal Resolution (HHR) and uses an image size of either 352 × 480 with 30 frames/s or 352 × 288 with 25 frames/s. The interlaced format most often used in MPEG-2 video coding is 4:2:0 and uses an image size of either 720 × 480 with 30 frames/s or 720 × 576 with 25 frames/s. A word of caution about the image sizes mentioned in the various formats is in order, since MPEG video coding when using SIF, SIF-I, or HHI actually uses only 352 pels per line instead of 360 pels. This is so because the number of significant pels horizontally have to be a multiple of 16; this has to do with block sizes used in coding that we will discuss later. Of course, the chrominance resolution is half both horizontally and vertically in the case of each image resolution and format described. The various example image formats just discussed are shown in Fig. 5.12.

5.3.7 The 4:2:2 Format

In the 4:2:2 video format, in each frame of video, the number of samples per line of each chrominance component, *Cr* or *Cb*, is one-half of the number of samples per line of luminance. The chrominance resolution is full vertically, that is, it is same as that of the luminance resolution vertically. Alternatively, for every two samples of luminance forming a 2 × 1 array, two samples of chrominance, one a *Cr* sample and the other a *Cb* sample, exist. The exact location of chrominance samples with respect to luminance samples in a frame of 4:2:2 video is shown in Fig. 5.13.

MPEG-1 video coding allows only use of the 4:2:0 format; however, in specialized applications where higher chrominance resolution is required, MPEG-2 video coding also allows use of the 4:2:2 format. We now illustrate the vertical and temporal positions of samples in 4:2:2 by showing what amounts to the side view of a 4:2:2 frame, in Fig. 5.14. Although in principle a 4:2:2 frame may be interlaced or noninterlaced similar to a 4:2:0 frame, only an interlaced format is usually expected since the purpose of having full chrominance resolution vertically was mainly to eliminate color degradations associated with interlaced 4:2:0 chrominance.

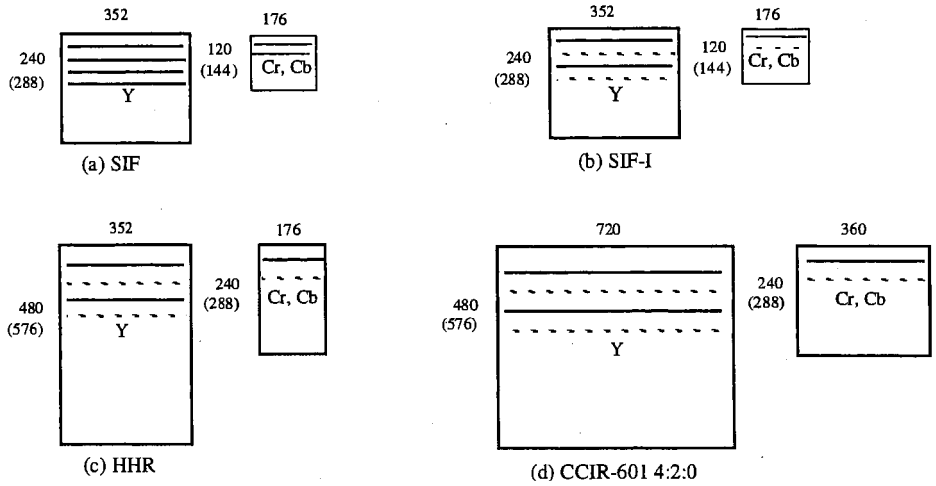


Fig. 5.12 SIF, SIF-1, HHR and CCIR-601 4:2:0 Video Formats

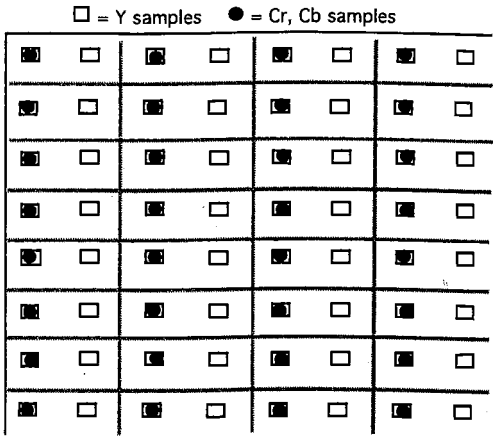


Fig. 5.13 Luminance and Chrominance samples in a 4:2:2 video frame.

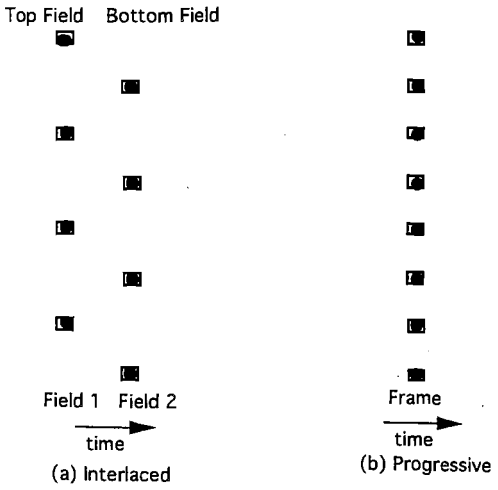


Fig. 5.14 Interlaced and Progressive representation of a 4:2:2 frame.

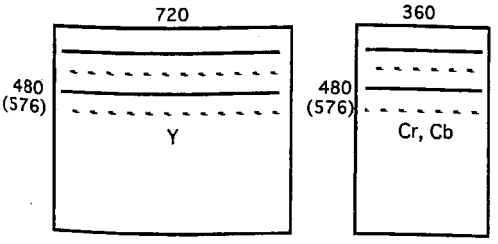


Fig. 5.15 CCIR-601 4:2:2 video format

As mentioned earlier, the 4:2:2 format can be used for MPEG-2 video coding. The main image resolution and format of interest in this case is CCIR-601 4:2:2 which uses an image size of either 720 × 480 with 30 frames/s or 720 × 576 with 25 frames/s. Of course, the chrominance resolution is half of this resolution horizontally. Figure 5.15 shows this image resolution and format.

5.3.8 The 4:4:4 Format

In a 4:4:4 video format, in each frame of video, the number of samples of each chrominance component, *Cr* or *Cb*, is the same as that of the number of samples of luminance, both horizontally and vertically. Alternatively, for every sample of luminance, two samples of chrominance, one a *Cr* sample and the other a *Cb* sample, exist. In a frame of 4:4:4 video the location of chrominance samples is the same as that of luminance samples as shown in Fig. 5.16.

The 4:4:4 format is useful mainly in computer graphics applications, where full chrominance resolution is required not only in the vertical but also in the horizontal direction. This format, although it is supported by MPEG-2, is not expected to find a wide use. In Fig. 5.17 we illustrate the vertical and temporal positions of samples in a 4:4:4 frame.

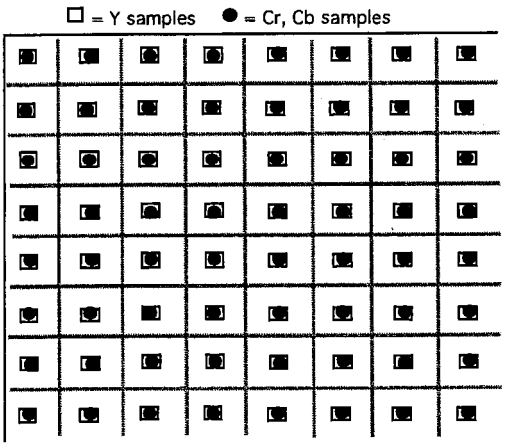


Fig. 5.16 Luminance and Chrominance samples in a 4:4:4 video frame.

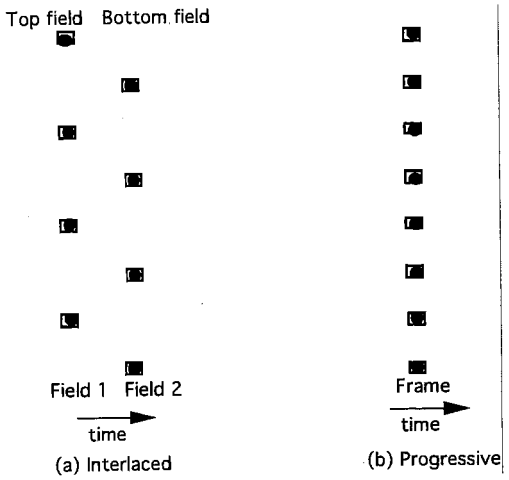


Fig. 5.17 Inerlaced and Progressive representation of a 4:4:4 frame

5.4 SPATIAL CONVERSIONS

In this section we discuss conversion[10,11] between CCIR-601 4:2:2 video format and the various sizes and formats mentioned in the previous section that are typically employed in MPEG video coding. Often, owing to the interlaced nature of CCIR-601 video, some of the conversions that we are about to discuss impact not only the spatial resolution of video but jointly the spatiotemporal resolution; however, we identify these conversions as spatial conversions to differentiate them from mainly temporal conversions discussed in the next section.

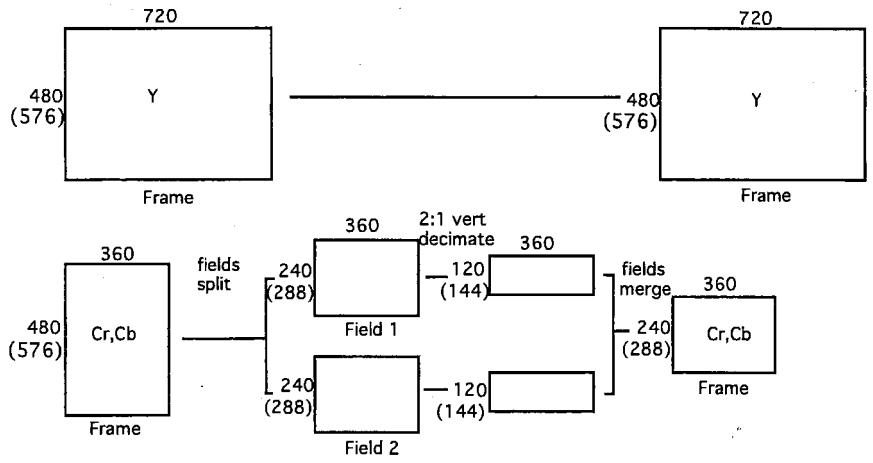
5.4.1 CCIR-601 4:2:2 and CCIR-601 4:2:0

The conversion of CCIR-601 4:2:2 to 4:2:0 involves reduction of vertical resolution of each of the two chrominance signals by one-half. In principle, this can be accomplished by simply subsampling by 2 vertically, which is same as throwing away every other line.

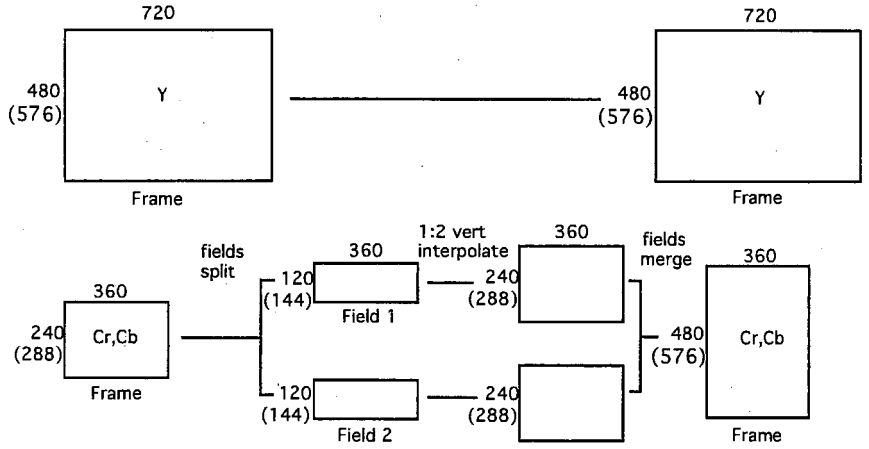
This, however, results in two problems: one, that of jaggard edges introduced on significant chrominance edges within each frame, and the other, a more serious one caused by the fact that since CCIR-601 4:2:2 is interlaced, simple vertical subsampling causes loss of one complete field of chrominance. These problems can be minimized by use of filtering although simply filtering across fields generates field to field artifacts due to motion in a scene. Thus, in each frame, each chrominance field should be filtered separately, field 1 by a filter of odd length and field 2 by a filter of even length (to generate lines at intermediate locations). Then each field is subsampled by a factor of 2 vertically. The filter used for filtering of chrominance field 1 is of length of 7 with coefficients $[-29, 0, 88, 138, 88, 0, -29]/256$ and that used for filtering chrominance field 2 is of length 4 with coefficients $[1, 7, 7, 1]/16$. The two resulting subsampled chrominance fields are merged to form a frame separately for each chrominance signal, which with the associated luminance frame is termed as the CCIR 601 4:2:0 frame. The operations involved in conversion of CCIR 601 4:2:2 to CCIR 601 4:2:0 are shown in Fig. 5.18.

The inverse operation, that is, conversion of CCIR 601 4:2:0 to CCIR 601 4:2:2, consists of upsampling the chrominance signal vertically by a factor of 2. Of course, upsampling can not really increase the resolution lost due to downsampling, but only increase the number of samples to that required for display purposes. Again, the simplest upsampling is pel repetition, which produces artifacts such as jaggy edges. Better upsampling involves use of an interpolation filter. Since we are dealing with interlaced 4:2:0 chrominance frames, interpolation needs to be performed separately on each field of chrominance samples. Referring to the two fields of a frame as field 1 and field 2, chrominance samples of field 1 of the 4:2:0 format are copied to form every other line of field 1 of the 4:2:2 format; the in-between lines of field 1 are generated by simple averaging using the line above and line below, which is same as using an interpolation filter of length 2 and coefficients $[1/2, 1/2]$.

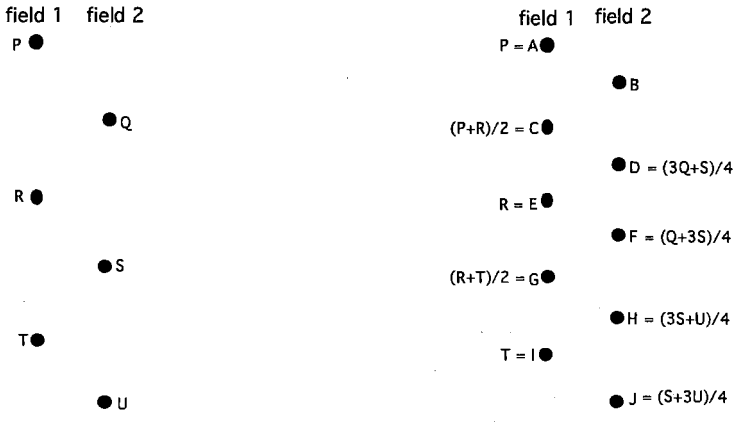
In field 2 of 4:2:2 format chrominance no samples of field 2 of 4:2:0 chrominance can be copied directly. All samples have to be calculated by using two different interpolation filters, each of length 2. The first interpolation filter has coefficients $[3/4, 1/4]$ and the second interpolation filter has coefficients $[1/4, 3/4]$. Each of the interpolation filters is used to compute alternate lines of field 2 of 4:2:2 chrominance. The various steps involved in con-



(a) From CCIR-601 4:2:2 to 4:2:0



(b) CCIR-601 4:2:0 to 4:2:2



(c) Chrominance samples in 4:2:0

(d) Chrominance samples in 4:2:2

Fig. 5.18 Conversion between CCIR-601 4:2:2 and 4:2:0

version of CCIR 601 4:2:0 to CCIR 601 4:2:2, including details of chrominance filtering, are shown in Figure 5.18.

5.4.2 CCIR-601 4:2:2 and SIF

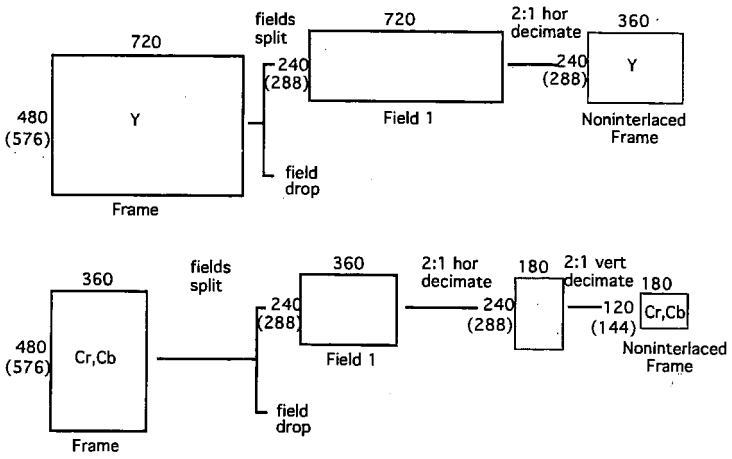
The CCIR-601 4:2:2 format is converted to the SIF format by first discarding field 2 of each frame of luminance and chrominance signals, retaining field 1 only. Next, the luminance and chrominance signals are horizontally subsampled by one-half it is filtered. This operation, known as horizontal decimation, consists of applying a decimation filter and retaining every other sample per line of luminance and chrominance signals. As an example, a filter of length 7 with coefficients [−29, 0, 88, 138, 88, 0, −29]/256 can be applied horizontally to every pel in every line of the original image to obtain a horizontally filtered image, followed by 2:1 subsampling horizontally. Eight luminance pels and four chrominance pels per line are then discarded to obtain 352 and 176 pels, respectively. The chrominance signals in the field are also vertically decimated by first filtering vertically, followed by 2:1 subsampling. The filter used for vertical decimation can be the same as that used for horizontal decimation. The various steps involved in conversion of CCIR-601 4:2:2 to SIF are shown in Fig. 5.19.

The inverse operation of conversion of SIF to CCIR-601 4:2:2 consists of upsampling of the luminance and each of the chrominance signals. The SIF luminance signal is first upsampled horizontally to twice the resolution, and then it is also vertically upsampled to twice the resolution. The SIF chrominance signals are first upsampled vertically to twice the resolution and they are then further upsampled horizontally and vertically again to twice the resolution to obtain the CCIR 601 4:2:2 format. As an example, the horizontal upsampling operation consists of first copying every pel of source resolution, say horizontal resolution of SIF, to every other pel location in the destination horizontal resolution of CCIR-601. Next, the intermediate missing pels are generated by applying filter coefficients to the pels copied earlier. The operation just described applies to horizontal upsampling of luminance and chrominance signals. An identical operation is applied for vertical upsampling. Both in horizontal and vertical upsampling the interpolation filter used is [−12, 140, 140, −12]/256. Note that although SIF can be upsampled to the same resolution as CCIR-601 4:2:2, the resulting signal is not quite same, because SIF is noninterlaced, whereas CCIR-601 4:2:2 is interlaced. The steps involved in the conversion of SIF to CCIR-601 4:2:2 are shown in Fig. 5.19.

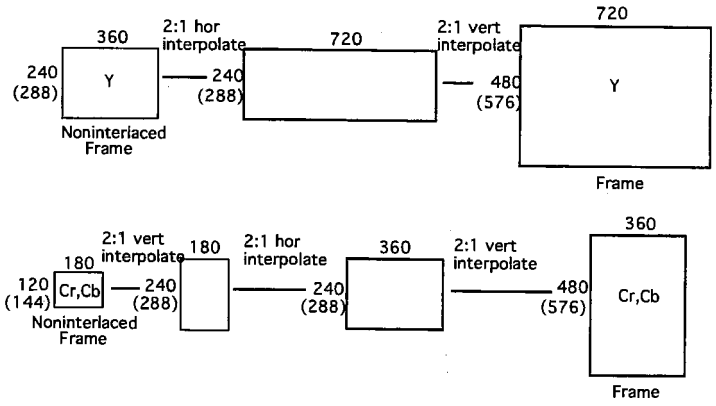
5.4.3 CCIR-601 4:2:2 and HHR

The conversion of CCIR-601 4:2:2 to HHR consists of reducing the horizontal resolution of the luminance signal by one-half, whereas for chrominance signals, both horizontal and vertical resolution is reduced by one-half. The process of reducing luminance and chrominance resolution horizontally to one-half is identical to the one performed for reducing the horizontal resolution of CCIR 601 4:2:2 to SIF, described earlier. However, as compared to the conversion to SIF, no vertical subsampling of luminance is needed and thus no field dropping is necessary. The process of conversion of the chrominance signal to half of its resolution vertically is identical to the process followed in the conversion of CCIR-601 4:2:2 to 4:2:0. Eight luminance pels and four chrominance pels per line are then discarded to obtain 352 and 176 pels, respectively. An example filter that can be used in all the decimation operations uses a length of 7 with coefficients [−29, 0, 88, 138, 88, 0, −29]/256 and is used to compute each pel in source resolution before subsampling by a factor of 2 to achieve half-resolution. The steps involved in the conversion of CCIR-601 4:2:2 to HHR are shown in Fig. 5.20.

The conversion of HHR to CCIR-601 4:2:2 consists of upsampling the luminance and chrominance by a factor of 2 horizontally. Since full luminance resolution is present vertically in the HHR format, no upsampling is needed vertically. The chrominance signals are then upsampled vertically. Here, the upsampling operation is the same as that described for upsampling of 4:2:0 to CCIR-601 4:2:2, and involves use of interpolation filters. The sequence of steps in conversion of HHR to CCIR-601 4:2:2 is also shown in Fig. 5.20.



(a) From CCIR-601 4:2:2 to SIF

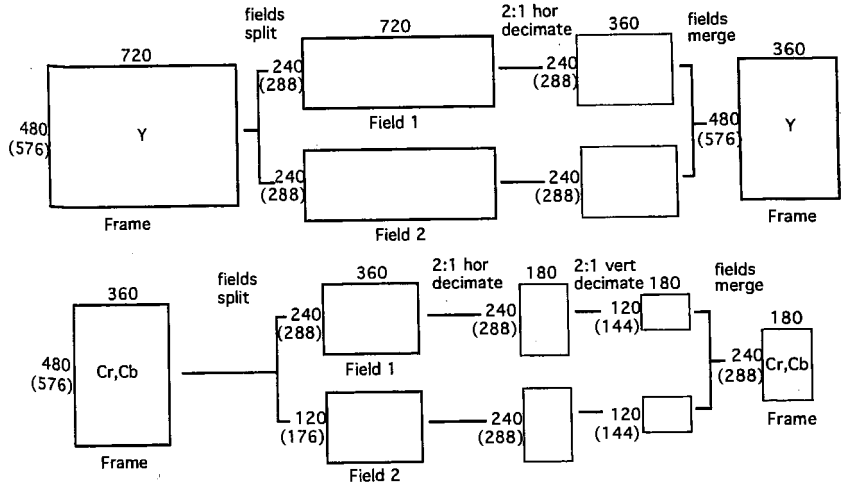


(b) From SIF to CCIR-601 4:2:2

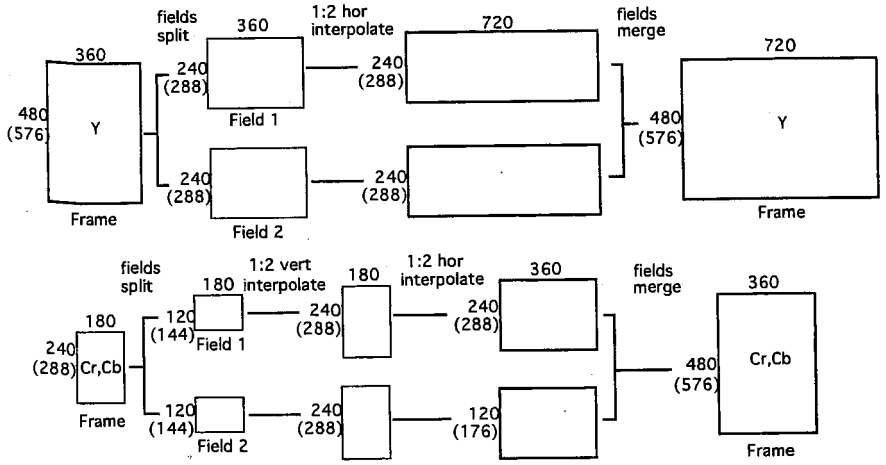
Fig. 5.19 Conversion between CCIR-601 4:2:2 and SIF. 8 out of 360 pels are discarded to obtain 352 luminance pels. Four out of 180 pels are discarded to obtain 176 chrominance pels.

5.4.4 CCIR-601 4:2:0 and SIF-I

In the conversion of CCIR-601 4:2:2 to SIF-I, although downsampling is performed to derive a resolution equal to that of SIF, the format of the resulting SIF is interlaced rather than the normal noninterlaced. A simple method is suggested here and is known to produce somewhat blurry images. For luminance downsampling, each frame of luminance is separated into individual fields and downsampled to one-half resolution horizontally, followed by further downsampling to one-half resolution vertically. The downsampling of chrominance is accomplished also by first separating chrominance to fields and then downsampling each field horizontally and vertically respectively, and finally again downsampling vertically prior to merging of the two fields of chrominance. Eight luminance pels and four chrominance pels per line are then discarded to obtain 352 and 176 pels, respectively. As an example, the filter specified by $[-29, 0, 88, 138, 88, 0, -29]/256$ can be used for horizontal



(a) From CCIR-601 4:2:2 to HHR



(b) From HHR to CCIR-601 4:2:2

Fig. 5.20 Conversion between CCIR-601 4:2:2 and HHR. 8 out of 360 pels are discarded to obtain 352 luminance pels. Four out of 180 pels are discarded to obtain 176 chrominance pels.

filtering before subsampling. For vertical subsampling of field 1, the same filter can be used, whereas for vertical subsampling of field 2, a filter with an even number of taps such as $[-4, 23, 109, 109, 23, -4]/256$ can be used. The sequence of steps in the conversion of CCIR-601 4:2:2 to SIF-I is shown in Fig. 5.21.

The conversion of SIF-I to CCIR-601 4:2:2 is the inverse operation. For luminance, it consists of separating each frame into fields and upsampling each field horizontally and vertically, followed by field merging. Next, the SIF-I chrominance is separated into individual fields and upsampled to twice the vertical resolution two

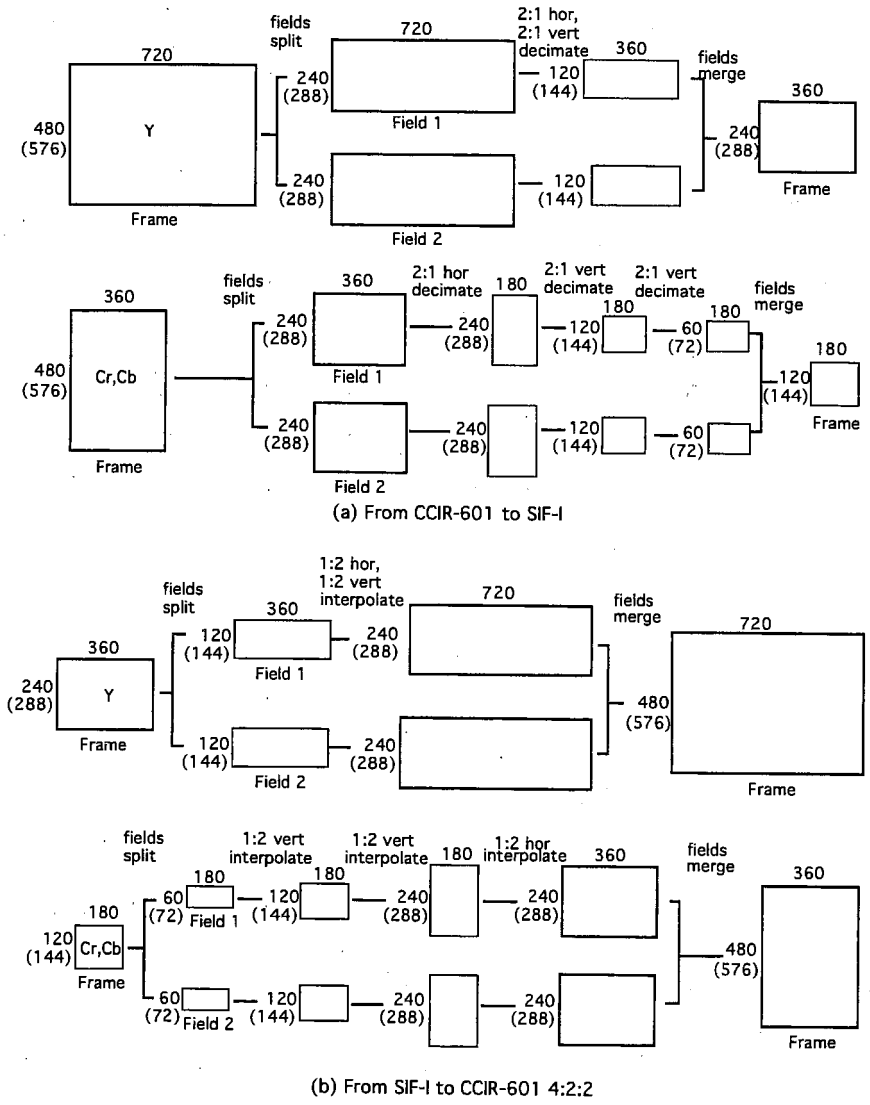


Fig. 5.21 Conversion between CCIR-601 4:2:2 and SIF-I. 8 out of 360 pels are discarded to obtain 352 luminance pels. Four out of 180 pels are discarded to obtain 176 chrominance pels.

times, followed by horizontal upsampling to twice the size. The resulting fields are merged. For upsampling, interpolation filters are used; in particular, horizontal upsampling can be performed using a $[-12, 140, 140, -12]/256$ filter. For vertical upsampling of field 1 the same filter as that used for horizontal upsampling can be used, whereas for vertical upsampling of field 2, an interpolation filter such as $[-2, 20, 110, 110, 20, -2]/256$ can be used. The sequence of steps in conversion of SIF-I to CCIR-601 4:2:2 are also shown in Fig. 5.21.

5.5 TEMPORAL CONVERSIONS

In the previous section we discussed some basic approaches for conversion between the source of CCIR-601 4:2:2 format to the various resolutions and formats used in MPEG-1 and MPEG-2 video coding and back to display resolutions and formats. Conversions of horizontal sizes and some conversions of vertical sizes are called spatial conversions, while other vertical size conversions also change field rates and are thus referred to as vertical/temporal or here simply as temporal conversions.[1,4,10,11] We have chosen to include only those conversions that may be directly relevant within the context of MPEG-2 video coding and display.

As a typical example, interlaced video of 60 fields/s normal rate, coded by MPEG-2, may have to be displayed on a computer workstation operating at 72 progressive frames/s. Another example is that of 24 frames/s film coded by MPEG-1 or MPEG-2 which needs conversion to 60 fields/s for display on a TV monitor. Yet another example may involve a PAL video format CD employing 50 fields/s compressed by MPEG-2 to be played on NTSC video format equipment using a display rate of 60 frames/s. In discussing the following general techniques for conversion, we do not assume a specific spatial resolution; our main purpose is to show temporal resolution conversions only.

5.5.1 Interlaced and Noninterlaced

We first consider the case of conversion[2–4] of interlaced video of 50 or 60 fields/s to noninterlaced video with the same number of noninterlaced frames/s. This operation is also called deinterlacing or line-doubling and involves generation of a noninterlaced frame corresponding to each field of interlaced video, effectively doubling the number of samples in the vertical direction.

A simple method to convert interlaced to noninterlaced is based on *line interpolation* in each field, which involves generating intermediate lines between each pair of consecutive lines in each field. A simple method to achieve line interpolation is by straightforward averaging of two lines to produce a line in between, which is the same as saying the interpolation filter has length 2 with coefficients of $[1/2, 1/2]$. The complete noninterlaced frame can be now created by alternately copying lines of a field and interpolating lines, generating a frame with twice as many lines as a field.

Yet another technique to accomplish interlaced to noninterlaced conversion is by *field merging*, which consists of generating a frame by combining lines of two consecutive fields to produce a frame of twice the number of lines per field. It turns out that although in stationary areas, field merging results in full vertical resolution, in the areas with motion, field merging produces judder artifacts. For improved results, line interpolation and field merging could be combined, such that line interpolation would be used for moving areas and field merging for stationary areas. However, such adaptivity itself can introduce visible artifacts owing to switching between methods, and so using a weighting function to combine line interpolation and field merging methods may be a feasible solution.

As an alternative to all this complexity, a simple method for conversion of interlaced to noninterlaced uses a *frame based line interpolation* and works similar to line interpolation; however, the interpolated line is generated by applying the interpolation filter on an interlaced frame. The noninterlaced frame is generated by alternating between copying a line from a field of an interlaced frame and interpolating the next line using the interlaced frame. The interpolation filter applied on frame lines has length 5 and coefficients of $[-1/8, 1/2, 1/4, 1/2, -1/8]$. The details of this method are shown in Fig. 5.22.

We now consider the case of conversion of noninterlaced to interlaced video, in which, given full resolution for each frame at say 60 frames/s, we reduce the vertical resolution by a factor of 2 to obtain 60 fields/s such that fields can be paired to form interlaced frames for display.

The simplest technique for noninterlaced to interlaced conversion is based on *line subsampling* vertically of each frame to form a field. On consecutive frames, line subsampling is implemented to yield fields of opposite parities that can be interleaved to form interlaced frames. Thus the lines of alternate fields are at identical locations but the

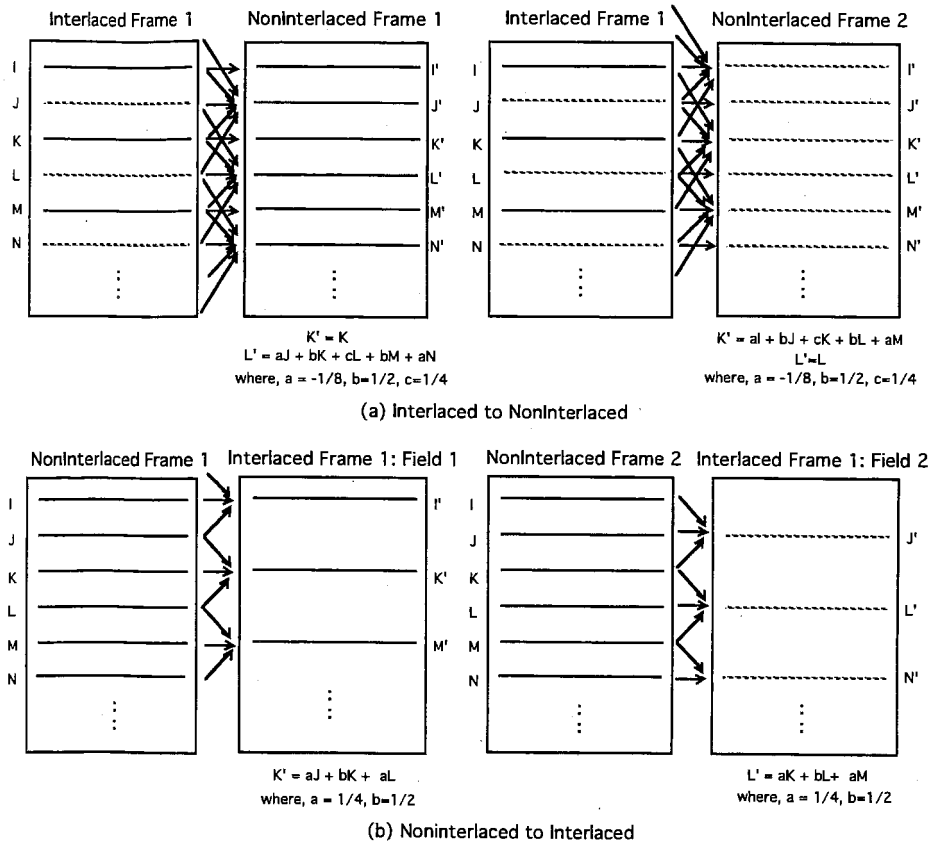


Fig. 5.22 Temporal Conversion between Interlaced and Noninterlaced

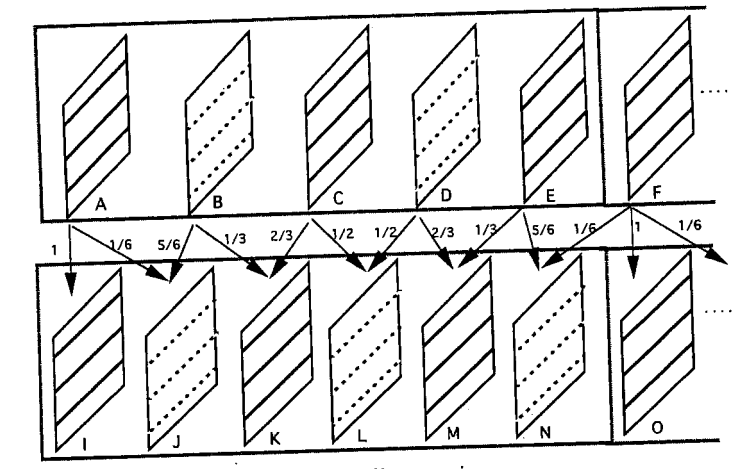
lines of consecutive fields are offset by half of the line distance between lines in a field.

Further improvements in noninterlaced to interlaced conversion is possible by *line weighting*, in which each line of a field is formed by a weighted combination of consecutive lines in the noninterlaced frame. This method, employing a simple example filter of length 3 with coefficients [$^1/_4$, $^1/_2$, $^1/_4$], is shown in Fig. 5.22. For better results, a filter of length 5 or 7 may be necessary.

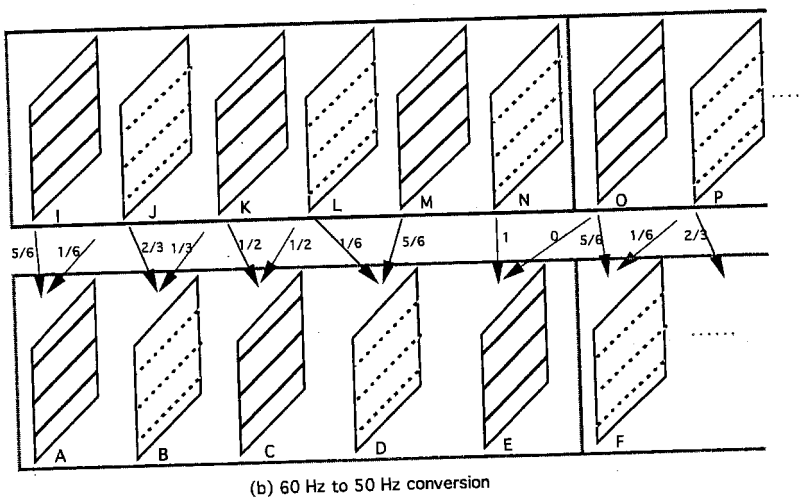
5.5.2 Field Rate Conversion

We now discuss basic techniques for conversion[2-4] of field rates between NTSC and PAL TV systems. To convert 60 fields/s to 50 fields/s, for every six fields of input, five fields are output. A simple way of doing that is to discard one out of every six fields, and from the 60 fields/s source rate, the fields closest in timing to the 50 fields/s destination rate are selected. However, this simple method does not work very well, as with fast motion, glitches in time become apparent. A somewhat better solution is to interpolate fields in time by a weighted combination of fields at the source rate of 60 fields/s to produce destination fields at 50 fields/s. Figure 5.23 shows this operation, including the weighting factors needed for 60 fields/s to 50 fields/s conversion.

The inverse operation, that is, conversion of 50 fields/s video to 60 fields/s video, can also be done similarly using temporal interpolation. This is also



(a) 50 Hz to 60 Hz conversion



(b) 60 Hz to 50 Hz conversion

Fig. 5.23 Temporal Conversion between 50 and 60Hz Field rates

shown in Figure 5.23, including the weighting factors needed for the conversion.

It should be noted that the temporal interpolations just described have a problem when there is motion in a scene. In principle a much better method is to compensate for motion before interpolation. In this method each *target field* is segmented into nonoverlapping small blocks (say 8×8 or 16×16) and motion vectors are computed by block matching with respect to a *reference field*. These motion vectors are scaled by a factor reflecting the relative distance of the field to be interpolated from the target field as compared to the total distance between the target and the reference field. To clarify, let us assume we want to interpolate a field located at a distance of two-thirds from a target field and one-third from a reference field. The motion vectors would then have to be scaled by a factor of two-thirds and motion-compensated blocks would be obtained from the corresponding reference field. There are still some problems with this approach, for example, block matching motion vectors may not reflect actual motion and thus may not be meaningfully scaled. Also, truncation

or rounding errors in motion vectors affect the accuracy of motion compensation. The following two chapters will discuss motion compensation for compression in much more detail, but here our primary purpose was to mention that motion compensation can be used in field rate conversion. Incidentally, commercially available standards converters that perform conversion between NTSC and PAL TV formats use the same basic principles discussed here, including those that use motion compensation.

5.5.3 Film and Interlaced Video Conversions

As discussed earlier, film is usually recorded at the rate of 24 frames/s. For transfer of film to PAL or SECAM, it is often played back a little faster, at 25 frames/s. This results in a slight distortion of pitch of the accompanying sound; however, this distortion is not often very noticeable. For transfer of film to NTSC, which uses 30 interlaced frames/s, 3:2 pulldown is necessary. This involves generating five fields of video per pair of film frames, resulting in 60 fields/s. The process of conversion[1–4] of film to interlaced NTSC video is shown in Figure 5.24. As shown by solid arrows in the figure, a film frame may be used to generate three fields, two of which are essentially identical, whereas the next film frame may be used to generate two fields and so on. Depending on the method used for film to video conversion, the three fields generated from a film frame may be exactly identical. For scenes with high-speed motion of objects, the selection of a film frame to be used for generating a field may be performed manually to minimize motion artifacts.

Sometimes film may already be available in the TV format at 30 interlaced frames/s, because film may have been converted to video using a Telecine

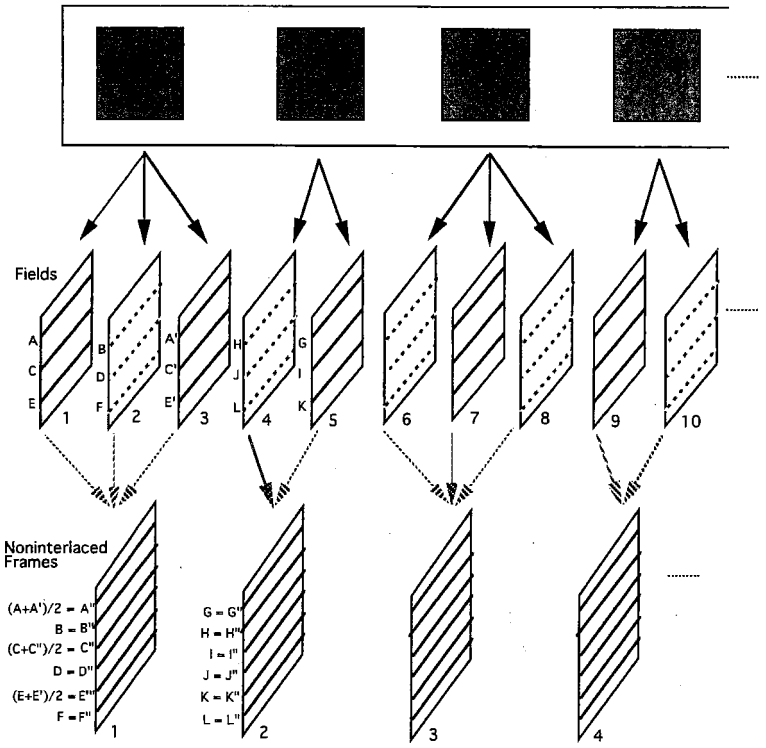


Fig. 5.24 Temporal Conversion between Film and Interlaced video, also referred to as 3:2 pulldown.

camera which can output interlaced video by performing automatic 3:2 pulldown to yield 60 fields/s. If so, it can be more efficiently encoded by MPEG video coding by first undoing the 3:2 pull down to regenerate 24 frames/s noninterlaced video. In undoing the 3:2 pulldown before coding, it may be a good idea to generate the noninterlaced frame by first averaging the first and third out of every group of 3 fields to reduce noise and then combining it with the second field to form a noninterlaced frame. The operation of undoing 3:2 pulldown is shown by dotted arrows in Figure 5.24.

However, since NTSC television uses 60 fields/s for display, after MPEG video decoding, a 3:2 pulldown would be necessary to convert 24 frames/s to 60 fields/s rate for display. On the other hand, if film is available at noninterlaced 24 frames/s, it can be coded directly by MPEG video coding and the only conversion needed is a 3:2 pulldown to produce 60 fields/s for display on TV.

5.6 MIXING AND KEYING

We now discuss combining of video signals by two methods, one called *mixing*[2-4] and the other called *keying*[2-4]

Mixing is used to overlay text or graphics on a video signal or fade to or from a color, say black. Mixing may consist of something as simple as switching between two video signals or something more such as a blended transition between two video signals.

Keying is used to generate special effects such as inserting a background from one scene into another scene containing foreground objects. Keying usually happens within video frames. It basically involves substituting a background signal in portions of the foreground signal based on some attribute such as a luminance or a chrominance value, or a specified region.

As a matter of interest, morphing involves smooth transition over time from an object of interest in one video sequence to an object of interest in another scene. It involves defining key points between the two objects and blending of shape as well as blending of amplitudes of signals.

5.6.1 Mixing

Mixing can be defined as the addition of two or more input video signals, each in some proportion *alpha* to generate a single mixed output video signal. The alpha information, α, refers to the degree of opacity of each scene and each video signal may carry its own α, a video signal with higher α contributing proportionally higher value to the output. Generalized alpha mixing can be represented by the following:

$$\text{Video} = \alpha_1 \times \text{Video1} + \alpha_2 \times \text{Video2} + \ldots$$

A specific situation occurs when a crossfade is implemented between two scenes. Typically, in the case of crossfade, alpha values and up to 1 and a single value of alpha is sufficient, resulting in the following operation:

$$\text{Vmix} = \alpha \times \text{Video1} + (1 - \alpha) \times \text{Video2}$$

Here, when α is 1 it implies the absence of Video2, whereas when α is 0, only Video2 is present. In between these two extremes, a weighted combination of the two scenes results. By increasing the α values as a function of time, a crossfade from one scene to another, say from Video2 to Video1, can be implemented. If the values are increased gradually the crossfade is slow, otherwise, a fast crossfade is accomplished.

5.6.2 Luma Keying

In luma (short for luminance) keying, a level of luminance signal is specified for the scene containing the foreground such that all pels above that value are replaced by the corresponding pels of the scene containing the background. Luma keying can also be implemented by specifying two threshold values Y_l and Y_h for the luminance of the foreground, Y_{fg}. More specifically, the background can be keyed into the brightest areas in the foreground signal by using a keying signal k. The foreground can be retained in the low-brightness areas, and a linear blended signal can be introduced in other regions.

This is done by defining the keying signal in terms of luminance values as follows:

$k = 1$ when $Y_h < Y_{fg}$ use background

$k = 0$ when $Y_{fg} < Y_l$ use foreground

$k = (Y_{fg} - Y_l)/(Y_h - Y_l)$ else $Y_l \leq Y_{fg} \leq Y_h$ use blended

Alternatively, it is also possible to introduce the background into the darkest areas of the foreground signal.

5.6.3 Chroma Keying

Chroma keying consists of substituting a background signal in place of the foreground signal based on some key color identified in the foreground video signal. Chroma keying is also popularly known as the "blue screen," due to frequent choice of a bright blue color for keying in the background signal.

If chroma keying is implemented as a switch within a frame between foreground and background sources, problems occur in areas where the foreground is transparent since the background is not visible through these areas; also, shadows from the foreground are not present in areas containing the background. By implementing chroma keying as a linear combination of foreground and background, the problems due to transparent foreground can be resolved. However, the problem of shadows is not fully resolved. By using a uniformly lit blue background on which foreground objects cast shadows, and using this signal for mixing with the background, the effect of foreground shadows can be transferred to the areas where the background is selected. This type of chroma keying is called *shadow chroma keying* and also results in improvements in areas where the foreground is transparent. Chroma keying also has some problems when foreground colors are close to the key color. It also appears that although a lighted blue background resolves many problems, it also introduces a new one due to color leakage between the foreground signal and the blue background. The steps in chroma keying of two video signals are shown in Figure 5.25.

A good method to resolve problems associated with chroma keying involves processing foreground and background scenes separately and using two keying signals, k_{fg} for foreground and k_{bg} for background, each with values in the 0 to 1 range.

5.7 SUMMARY

In this chapter, we first learned about the raster scanning of video, human visual system, color spaces, and various formats used by MPEG video. Next, we discussed spatial conversions used in preprocessing prior to MPEG video encoding and postprocessing after video decoding for display purposes. Then, we described several temporal conversions used in preprocessing and postprocessing. Finally, we discussed video mixing, a topic of general interest in many applications. We now summarize the highlights of this chapter as follows:

- Raster scanning can be either interlaced or noninterlaced (progressive). Both TV and motion pictures, although fairly different, use the same basic principle of portraying motion in a scene by showing slightly different images taken over time to replicate motion in a scene. The image aspect ratio (IAR) controls the aesthetic appearance of displayed image frames and generally refers to the ratio of width to height.
- Color spaces of composite TV systems—NTSC, PAL, and SECAM—are all derived from YUV space. However, the component system, and in particular the CCIR-601 video format, is of primary interest in MPEG coding.
- Analog to digital conversion involves the steps of filtering, sampling, quantization, and PCM coding.
- MPEG coding employs various formats derived from the CCIR-601 digital video format. MPEG-1 video coding was optimized for the SIF noninterlaced format. MPEG-2 video coding was optimized for the CCIR-601 4:2:0 interlaced formats. MPEG-2 video coding often also employs HHR interlaced formats.
- Spatial conversions are either through decimation or interpolation, where decimation means subsampling to reduce resolution and interpolation means upsampling to produce a higher number of samples. To avoid aliasing, filtering is necessary before subsampling. To avoid upsampling jaggies, an interpolation filter is used instead of simple pel or line repetition. When performing the decimation or interpolation on interlaced frames, these operations are performed on individual fields rather than frames.

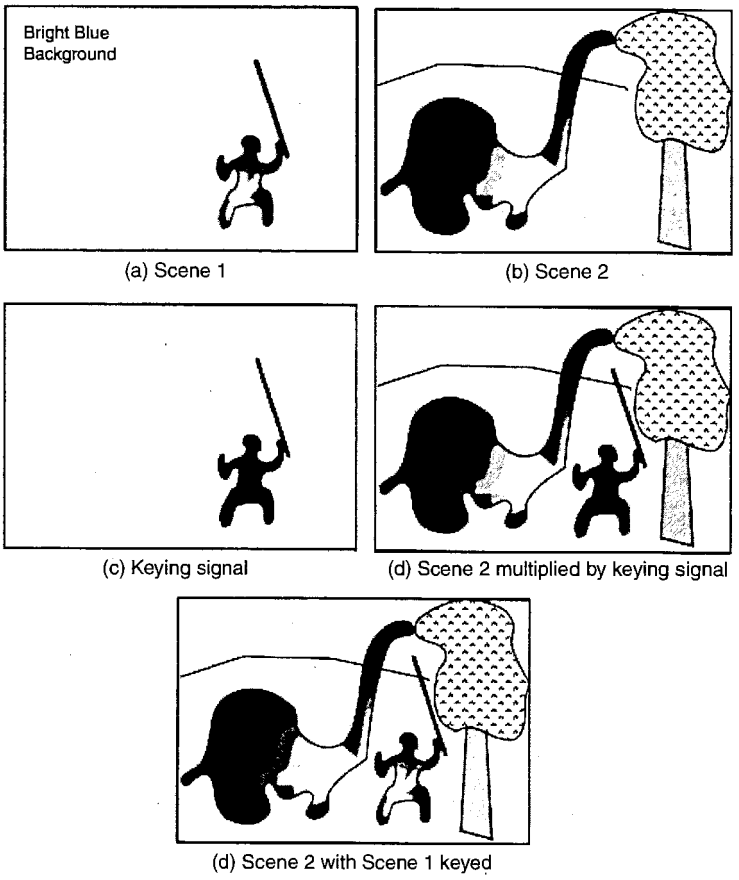


Fig. 5.25 Chroma Keying

• Temporal conversions involve conversions between interlaced and noninterlaced frames, conversion of field rate between 50 and 60 field/s TV systems, and 3:2 pull down between film and interlaced video. In field-rate conversion, sometimes simple methods such as weighted field interpolation are not sufficient and scene motion based weighted combination may be necessary. If film material is available as 60-Hz interlaced video, it may be converted to 24-Hz noninterlaced (progressive) format before MPEG coding.

• Mixing of several video signals to produce a new mixed signal can be done by hard-keying or by soft-keying (alpha blending). Keying can be done by using selected thresholds on brightness values or by using a selected color; the corresponding methods are referred to as luma-keying or chroma-keying. Chroma-keying is often referred to as blue-screen, because of the usual selection of blue for keying.

REFERENCES

1. A. N. Netravali and B. G. Haskell, *Digital Pictures: Representation, Compression and Standards,* Second Edition, Plenum Press, New York (1995).
2. A. F. Inglis, *Video Engineering,* McGraw-Hill, New York (1993).
3. K. B. Benson and D. G. Fink, *HDTV: Advanced Television for the 1990s,* McGraw-Hill, New York (1991).

4. K. JACK, *Video Demystified: A Handbook for the Digital Engineer,* Brooktree, (1993).
5. T. RZESZEWSKI (Ed.), *Color Television,* IEEE Press, New York (1983).
6. A. N. NETRAVALI and B. PRASADA (Ed.), *Visual Communication Systems,* IEEE Press, New York (1989).
7. W. K. PRATT, *Digital Image Processing,* John Wiley & Sons, New York (1978).
8. A. C. LUTHER, *Digital Video in the PC Environment,* McGraw-Hill, New York (1989).
9. A. ROSENFELD and A. C. KAK, *Digital Picture Processing,* Academic Press, New York (1982).
10. Video Simulation Model Editing Committee, MPEG Video Simulation Model 3 (SM3) (July 1990).
11. Test Model Editing Committee, "Test Model 5," *ISO/IEC JTC1/S29/WG11/N0400* (April 1993).

12

DIGITAL VIDEO NETWORKS

Broadband access networks to support digital multimedia services present two interrelated challenges: (1) design of the best network architecture that is cost effective and provides easy migration from the current situation, and (2) creating a uniform control and software infrastructure for end-to-end support and modular growth of multimedia services. The current situation is rather complicated because of regulation and participation of different industry segments (see Fig. 12.1). Direct Broadcast Satellite (DBS) and Terrestrial Broadcast both transmit directly to the consumer's home. Cable Television (CATV) offers a large analog bandwidth over coaxial (coax) cable but over a shared channel (referred to as logical bus); Telephone Companies (Telco) offer a reliable, low bandwidth but with an individual connection (referred to as a star topology).

For delivery of digital multimedia information, the facilities may employ constant bitrate (CBR) or variable bitrate (VBR) transmission. The elementary streams will usually be packetized using the MPEG-2 Systems multiplexing standard. Further processing may also be carried out for particular transmission networks. For example, Asynchronous Transfer Mode (ATM) networks further packetize the digital data into 53 byte *Cells* having 47 or 48 byte payloads. Satellite and other wireless transmission channels typically require the addition of extra bits for Forward Error Correction (FEC).

CATV and Telco networks are both wired connections. Both are being upgraded using optical fiber technology to provide enhanced transmission capacity—both analog and digital. Three dominant wired architectures have emerged over time without a clear consensus on their comparative advantages and disadvantages. The first one, called the hybrid fiber-coax (HFC), can deliver 60 to 150 channels of analog (AM-VSB*) video while at the same time providing digital services using time division and frequency division multiplexing (TDM and FDM) all within a 500- to 1000-MHz bandwidth.

Service providers with more faith in switched digital services prefer point-to-point fiber-to-the-curb (FTTC) architectures, typically served by a 1 Gbits/s curbside switch. As an example, one such switch could provide switched 50 Mbits/s to each of about two dozen homes. Providing analog video is problematic on FTTC and usually needs an analog overlay on top of this architecture. The overlay could be configured along the HFC architecture.

In Europe and Japan, with less aggressive bandwidth expectation and less emphasis on analog video, point-to-multipoint passive optical networks (PONs) are used. PONs use passive splitters to

*Amplitude Modulation—Vestigial Side Band.

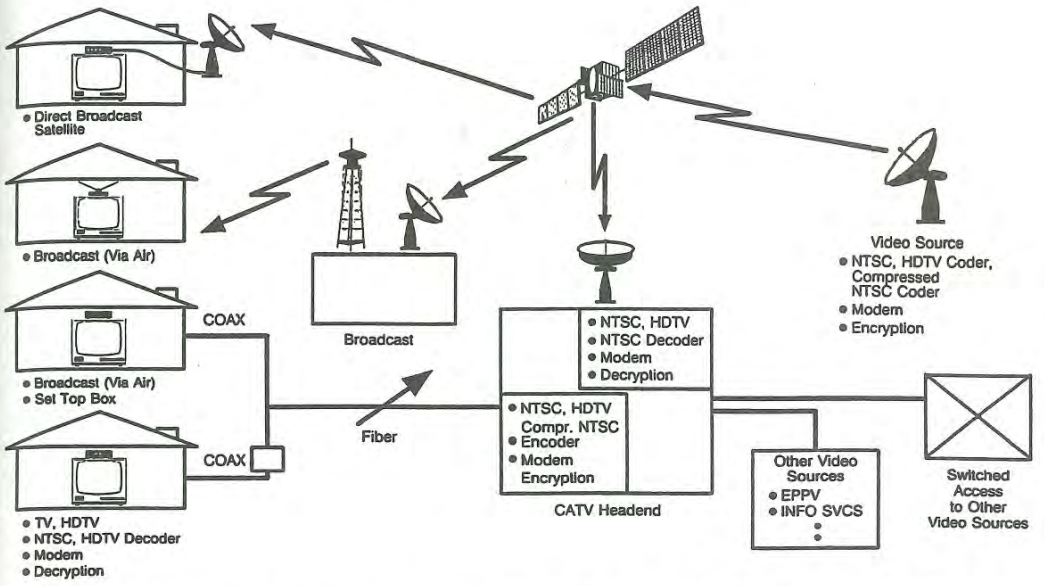


Fig. 12.1 Delivery of broadband video is currently over several media: (a) Direct Broadcast Satellite, (b) Terrestrial Broadcast, and (c) Wired systems (CATV and Telco). Each of these media are in different stages of becoming digital. This does not include delivery using stored media such as video-cassettes and CD-ROMS.

route a fiber feeder to a group of closely located homes, thereby reducing cost.

Each of the three architectures can provide on-demand digital video services. The difference is in the digital modulation schemes used and how the bandwidth is shared. The FTTC architecture uses baseband digital modulation and point-to-point distribution to individual customers from a time-multiplexed fiber feeder. On the other hand, the HFC architecture utilizes advanced digital modulation technology, for example, QAM (Quadrature Amplitude Modulation) or VSB (Vestigal Sideband) and a coax bus fed by a frequency multiplexed fiber feeder for distribution to customers. Between these two wired alternatives, there is a wide choice in other parameters, such as packetization, protocols, signaling and control, and so forth. Some of these will be discussed in this chapter.

There are nonwired alternatives as well. Terrestrial broadcasting, which has served analog television for many decades, may become digital as well. Although digitization is led by HDTV, digital SDTV* and other data-broadcasting may follow soon. In addition, high-power satellites are giving rise to direct broadcasting of digital video using small dish antennas and inexpensive electronics.

Figure 12.2 provides an end-to-end functional block diagram for delivering server content to the consumer terminals independent of the specific medium. Content is stored by service providers in multimedia servers (see Chapter 13). The access network includes all active and passive devices, as well as all cabling and service interface technologies. The access node provides appropriate format conversions to meet internal network interfaces for all the services. It is typically located within the Telco Central Office or CATV headend, which is usually within several tens of miles of the consumer home. The service node is usually a switch that connects a consumer to the server(s) of any service provider. Service management provides the gateway(s) through which a consumer selects information providers, manages service options, and navigates among multiple choices. Integrated

*Standard Definition TV, for example, NTSC in North America and Japan, PAL and SECAM in Europe and Asia.

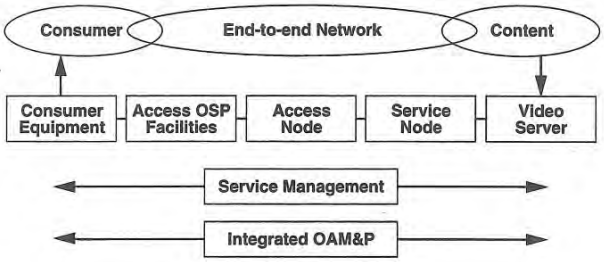


Fig. 12.2 To provide a full spectrum of normal and broad-band services, the platform must contain consumer equipment, an access network, and servers with content.

OAM&P (Operations, Administration, Maintenance, and Provisioning) refers to all the operations capabilities designed to operate a full-service network with high efficiency and minimal human intervention.

In this chapter, we describe each of the media alternatives from a full-service end-to-end network perspective, cover their strengths and weaknesses, and describe a possible evolution path into the future.

12.1 CURRENT METHODS OF ACCESS[1-5]

Telcos provide telephone service over a network constructed of cabled sets of wire pairs, varying in count between six and 3600 wire pairs per cable (see Fig. 12.3). This supports a 4-kHz analog bandwidth for voice. Individual wire pairs are bundled together at various points in the network, to aggregate larger and larger groups of wire pairs, eventually connecting to the telephone central office. An important point is that each customer premise equipment (CPE) has a separate connection to the Telco central office and that the bandwidth is not shared. Therefore, the Telco architecture is said to have a star topology.

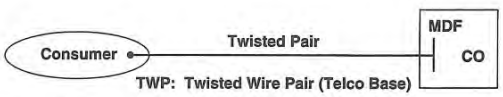


Fig. 12.3 Telco Architecture is narrowband, with twisted copper pairs connecting consumer equipment to the Main Distribution Frame (MDF) in the Central Office (CO).

Cable TV service providers transmit analog video signals from the headend (HE) to the consumer's home (see Fig. 12.4) over fiber coax networks. Lasers at the headend transmit the full RF spectrum (from about 50 MHz to typically either 330 MHz or 550 MHz) to "fiber nodes" serving 500 to 2000 homes. From the fiber node the signal is distributed electrically to homes via an amplified tree and branch coaxial cable network. Multiple-home taps are spaced along the coax distribution bus, and from each tap individual homes are connected via coax "drops." All consumers receive the same 40+ channels of composite VSB video, each with a bandwidth of 6 MHz. Trap filters are often placed in series with the drop termination to control access to premium services.

Long amplifier cascades for the older "all-coax networks" have been replaced by direct fiber links,

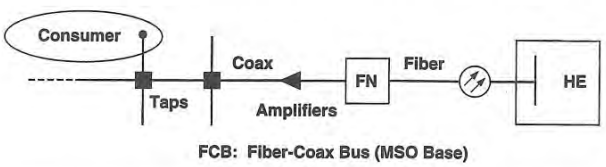


Fig. 12.4 Current CATV delivery system. Signals travel from the Head End (HE) to a fiber node in the CATV plant, followed

resulting in a maximum of three to ten amplifiers between a consumer and the HE. This improves both reliability and signal quality. CATV systems do not employ sophisticated operations support systems, and their networks provide little or no power redundancy. They are also cheaper on a per consumer basis than Telco networks.

12.2 NEW BROADBAND WIRED ACCESS SCENARIOS[6-12]

Both Telcos and CATV operators can provide incremental upgrades for the short term. The Asymmetrical Digital Subscriber Line (ADSL) offers 1.5 to 6.0 Mbits/s downstream transport, depending on the length and the quality of the loop (see Fig. 12.5). However, at this time, it is too expensive, and moreover it does not provide an integrated means of both digital and AM-VSB transmission of video. While it has the advantage of serving a sparsely distributed subscriber base with investment added largely as subscribers are added, it also requires central office (CO) modems and extra switching functionality in the CO to accomplish a channel for "broadcast" services. These aspects limit the overall popularity of the ADSL technology.

Fig. 12.5 Asymmetrical Digital Subscriber Line (ADSL) provides a substantial increase in bitrate compared with the current telephone plant. However, it does not provide for evolution to the higher bidirectional bitrates that may be required in the future.

CATV systems can construct incrementally a Bidirectional Fiber Coax Bus (BFCB) by (1) increasing downstream bandwidth to 750 MHz and (2) providing a very narrowband upstream channel for limited interactive control, as shown in Fig. 12.6. These steps are equivalent in cost to that required to build a hybrid fiber coax network, with substantial improvement in service capability.

Three dominant architectures are gaining momentum among service providers. The first of these, called Hybrid Fiber Coax (HFC), is being considered by all the CATV operators (and some Telcos) as an extension of the current CATV fiber-coax topology (Fig. 12.7). The second architecture, called the Fiber to the Curb (FTTC), is being studied by the Telcos (Fig. 12.8). The third architecture, popular in Europe and Japan, is called the Passive Optical Network (PON) as shown in Fig. 12.9. All three architectures and their numerous variations have relative strengths and weaknesses, and the choice between them is influenced to a

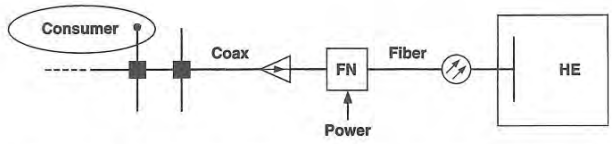


Fig. 12.6 A Bidirectional Fiber Coax Bus (BFCB) provides an in-between step from the current CATV architecture to the Hybrid Fiber Coax (HFC).

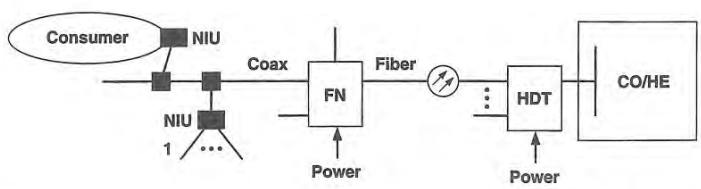


Fig. 12.7 HFC is a strong candidate for evolution of the current CATV architecture and has also been chosen by several Telcos.

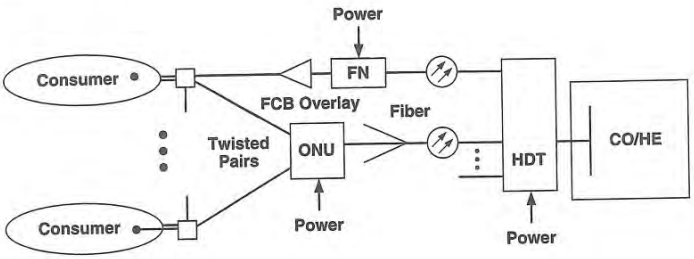


Fig. 12.8 The fiber-to-the-curb (FTTC) system provides fiber almost to the customer premise (HE/CO to ONU). Consumer homes connect to the ONU by either twisted pair (as shown) or by a coax. Provision of analog broadcast video requires a Fiber-Coax Bus (FCB) overlay.

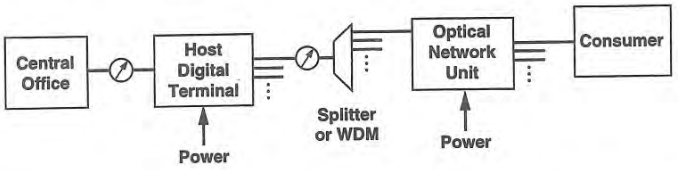


Fig. 12.9 PON systems are basically point-to-multipoint in that they use passive splitters to share a fiber feeder among many closely spaced homes.

large extent by the relative weight given to these factors by service providers.

The HFC architecture provides significant upstream bandwidth in the 5 to 40 MHz band. From the headend in the "downstream" direction, linear photonic transmission equipment is used to transport telephony, various data services, switched digital video, broadcast digital video, and broadcast analog AM-VSB cable TV signals to a "fiber node" (FN) which typically serves up to 500 homes. From the fiber node, transmission is bidirectional over a coaxial cable tree and branch topology to Network Interface Units (NIUs), which serve a very small group of homes. Broadband video (analog and digital) signals are carried into the home on coaxial cable for distribution to cable set-top boxes (STBs). Digital signals are encoded and decoded only once at the HDT (or elsewhere) using standard modulation techniques (e.g., 64-QAM) without any form of further transcodings. This improves the quality of the signal displayed on the consumer TV.

In the FTTC architecture, all traffic except analog video is carried digitally over point-to-point fiber facilities from the central office to the optical

several dozen homes. However, two practical considerations force a significant variation from this ideal. To power the ONUs and to transport broadcast analog TV, practical FTTC systems require an overlay carrying electrical power and analog TV channels.

A single point-to-point fiber might serve a single ONU, and through it a single home (Fiber-to-the-Home or FTTH), giving unlimited bidirectional digital bandwidth to the home. However, to reduce costs, both the fiber and ONU must be shared. Such sharing converts the point-to-point physical fiber star architecture into a point-to-multipoint architecture.

Hybrid twisted-pair/coax connects the ONU to the home. Twisted pair cable is used to carry the digital video signal in the 5 to 40 MHz band from the ONU to a distribution terminal no more than 500 feet away, where an RF combiner puts this signal on the coaxial drop carrying the analog TV signal. It is this 500 foot limitation on the range of the digital video signal on twisted pairs that limits the number of homes served by a single ONU.

The PONs currently under development provide narrowband services to multiple ONUs in a

12.9. Each fiber leaving the CO or host digital terminal serves multiple (8 to 32) ONUs. Various types of ONUs are used, depending on whether the fiber is terminated at a home, business, curbside pedestal, or multiunit building. Each ONU receives a baseband broadcast optical signal at approximately 50 Mbits/s, from which it selects the appropriate information. Upstream transmission from each ONU must be timed, in accordance with provisioned time-slot allocations, to avoid colliding at the HDT with transmission from other ONUs. Upstream and downstream transmissions are separated into alternating frames, using time-compression multiplexing, or can be full duplexed at two different wavelengths, for example, 1.30 and 1.55 micron.

12.3 WIRED ACCESS COMPARED

All the above access architectures, HFC, FTTC, and PON, provide full service capability with differing difficulty. The current generation of analog services can be better handled by HFC systems, since both FTTC and PON require special additions for providing analog video. In addition, the three systems can support: (1) data services, for example, 1.5 Mbits/s data that require wideband upstream access and (2) a goal of $1000 per home.

The future generation of services that need to be supported require all-digital transport and a broadband capability even in the upstream direction. Broadband bidirectional access could be provided by either HFC, FTTC, or PONs and many of their variations.

Investment-wise, the Bidirectional Fiber Coax Bus (Fig. 12.6) represents a relatively modest upgrade of existing fiber coax networks by providing:

- Digital downstream capability for video on-demand channels frequently shared with today's analog cable TV
- Very limited upstream bandwidth for control information of Pay-Per-View and Digital Video services.

To satisfy the full-service objective, the HFC architecture will have to be used by enabling more upstream bandwidth and making the bidirectional bandwidth both more reliable and available. This may require fiber to be driven to smaller nodes than the existing coax plant, and the drops may have to be reengineered.

Both Telcos and CATV companies have a significant investment in their current plant and must evolve this to meet the needs of the future services. CATV systems can either evolve through the intermediate step of a bidirectional fiber coax bus or go directly to full-service HFC. For Telcos, the ADSL technology provides an increment in bandwidth, but does not provide an evolutionary path toward full-service capability. Telcos must therefore begin constructing broadband access either via HFC or FTTC systems. The deciding factor may be provision of analog broadcast video. From this broadband base, both Telcos and CATV companies can evolve to an all-digital network as the consumer environment evolves and technology makes costs fall.

The HFC architecture, relative to FTTC and PON, has a lower first cost of implementation. The major reason for this is that, in the case of FTTC or PON, the analog cable TV must be carried with special arrangements such as a broadcast fiber coax overlay in the case of FTTC. For the CATV companies, the HFC architecture has a perfect "fit" to the embedded plant.

For the Telco, FTTC is a more natural fit to their operations plans and craft skills. In the currently deployed configurations, FTTC or PON get fiber closer to the home than does HFC. Some service providers want to get fiber physically closer to the home, while others focus on a predominantly passive plant and see little difference in the two architectures.

Current economics permits HFC systems to put the last point of network intelligence very close to the customer, even all the way to the home, while FTTC or PON systems are economical only with ONUs (the last point of network intelligence for these systems) which serve a few dozen homes. Points of network intelligence closer to the customer facilitate more complete automated fault isolation and more automated service provisioning. HFC systems provide bandwidth closer to the customer because all of the bandwidth carried on the

medium is available to every home in HFC systems, while in FTTC or PON the broad bandwidth of the fiber is terminated at the ONU. In FTTC systems, the hybrid twisted-pair/coax drop limits the bandwidth available to or from any home to only a small fraction of that on the fiber bus serving the ONU.

HFC systems being deployed by both Telcos and the CATV providers usually retain the set-top-box signaling and control technologies of the cable TV industry, while FTTC or PON systems typically plan to use the ATM protocol for the upstream signaling. This consideration allows FTTC or PON systems easier migration to packet-based services through ATM connectivity. However, ATM-capable modems are emerging from several vendors that will provide equivalent capability to HFC systems as well.

Evolving technology might make these three architectures ultimately converge. Over time, high-powered laser transmitters and linear optical fiber amplifiers will make video distribution directly to FTTC ONUs, or to HFC fiber nodes both serving approximately the same number of homes. Sometime thereafter, steadily declining electronics costs will permit further convergence of the architectures. Eventually the long held dream of "fiber directly to the home" will become possible. But far more important than this long-term system evolution view is the view of how the broadband access systems being deployed today will be evolvable to provide service parity with the newer systems that eventually emerge. All the three architectures, HFC, FTTC and PON, have this *in situ* evolutionary potential.

12.4 TERRESTRIAL BROADCASTING[12]

Current advanced television (ATV) research for terrestrial broadcasting in the VHF/UHF bands is converging toward a fully digital implementation. In the United States, a Grand Alliance* has been formed by proponents of digital HDTV† systems. The planned system is a digital simulcast with the current analog NTSC transmission (see Fig. 12.10). The goal of the Grand Alliance is to introduce an ATV terrestrial transmission standard for the United States. Similar digital initiatives are beginning to emerge in Europe and Japan.

In a digital ATV system, an MPEG-2 compressed HDTV signal is transmitted at approximately 15 Mbits/s. This data rate is sufficient to provide a satisfactory distribution‡ quality HDTV service. The aggregate data rate for an ATV sys-

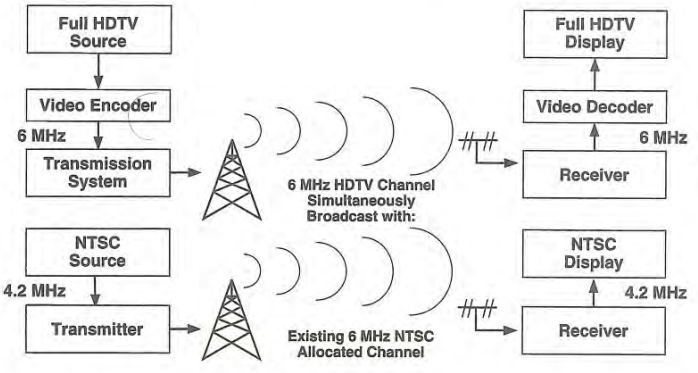


Fig. 12.10 Terrestial Simulcast of digital HDTV and analog NTSC with similar content. See Chapter 14.

*AT&T, David Sarnoff Research Center, General Instrument, M.I.T. North American Philips, Thomson Consumer Electronics, and Zenith Electronics.

†See Chapter 14 for a description of HDTV.

tem, which includes compressed video, compressed audio, conditional access, and an auxiliary data channel, is around 18 to 20 Mbits/s. In North America, this data rate must be squeezed into a channel of 6 MHz bandwidth for terrestrial broadcasting. For some other countries, the same data rate only needs to fit into a 7 or 8 MHz channel. Other considerations are: (1) the system must be able to sustain multipath distortion; (2) the system should be robust to interference to and from existing analog TV services; (3) the system should provide fast carrier recovery and system synchronization for rapid channel changing; and (4) the system should provide easy transcoding to and from other transmission media, such as satellite, cable, fiber, mobile reception, and computer networks.

Transmission of 18 to 20 Mbits/s in a 6 MHz terrestrial band could be achieved by a 16-QAM scheme where in-phase (I) and quadrature (Q) components each carries data of 2 bits/symbol. The symbol rate of each component is approximately equal to the usable bandwidth. Another alternative is a four-level VSB modulation scheme, where each symbol again carries 2 bits and the symbol rate is about twice the usable bandwidth. Thus, the bit rate is about the same as for 16-QAM. The noise performances of 4-VSB and 16-QAM are similar.

Another choice is between single-carrier modulation (SCM) and multicarrier modulation (MCM), such as orthogonal frequency division multiplexing (OFDM) currently under investigation in many countries in Europe and elsewhere. In SCM, the information bits are used to modulate a carrier that occupies the entire bandwidth of the RF channel. The SCM-QAM technique has long been used in satellite communications, line-of-sight digital microwave radio, telephone modems, and so forth.

In MCM, QAM symbols are used to modulate multiple low data-rate carriers, which are transmitted in small bands at different frequencies within the channel. Since each QAM modulated carrier has a $\sin(x)/x$ frequency spectrum, the carrier spacing is selected so that each carrier is located at the zero crossing points of the other carriers to achieve frequency domain orthogonality. Although the carrier spectra overlap, they can be orthogonally demodulated without intercarrier interference. MCM can be implemented via the digital Fast Fourier Transform (FFT), where an inverse FFT is used as the modulator and an FFT as the demodulator. One important feature of MCM is that, by inserting a guard interval of small duration between the MCM symbols, or FFT blocks, and using "cyclic extension," the intersymbol interference (ISI) can be eliminated. However, the amplitude/phase distortion within a MCM symbol still exists. Another advantage is that the MCM signal spectrum is more rectangular in shape, which means a higher spectrum efficiency than the SCM system. There are many projects in Europe to implement an MCM system for digital SDTV/HDTV broadcasting. However, more experience is necessary before a detailed comparison can be made.

MCM is more robust to time domain impulse interference, since the interference will be averaged over the entire FFT block. However, MCM is vulnerable to single frequency tone interference. SCM needs to reserve part of the spectrum for pulse shaping (frequency domain), while MCM needs to insert guard intervals (time domain). For channels with multipath distortion, SCM may need a training sequence to assist adaptive equalizer convergence and synchronization. MCM usually sends our reference carriers to obtain channel state information for frequency domain equalization and channel decoding.

The Grand Alliance HDTV system uses a concatenuated forward error correction (FEC) code with the inner code implemented using trellis coded modulation (TCM) and the outer code implemented using a Reed–Solomon (R–S) code. The inner code combines coding and modulation into one step to achieve high coding gains without affecting the bandwidth of the signal. In a TCM encoder, each symbol of n bits is mapped into a constellation of $(n + 1)$ bits, using a set partitioning rule. To achieve a spectrum efficiency of 4 bits/s/Hz, n is set to 4. The resulting TCM constellation is 32-QAM. The 4-VSB modulation, having 2 bits/symbol, can be trellis coded into 8-VSB, having 3 bits/symbol, including one error protection bit, for transmission. This scheme also achieves a spectrum efficiency of 4 bits/s/Hz,

since the symbol rate of VSB system is twice that of the QAM system. It should be noted that TCM coding only increases the constellation size and uses this additional redundancy to trellis-code the signal; there is no bandwidth expansion. A TCM code can be decoded with the Viterbi algorithm that exploits the soft decision nature of the received signal. The coding gain for a two-dimensional TCM code over a Gaussian channel is around 3 dB for a BER of 10^{-15}.

The output of the inner code delivers a symbol error protection rate, of approximately 10^{-3} to the outer code. In a R–S encoder, information data is partitioned into blocks of N information bytes (N is typically between 100 and 200). The output of the R–S encoder has a block size of $N + 2t$, where $2t$ additional bytes are generated for each block, which can correct t-byte transmission errors. Since more data are "generated" by a R–S encoder, additional bandwidth is required to transmit the coded data. Because of the bandwidth constraint in broadcasting, the redundancy must be limited to about 10 percent.

The motivation for using TCM as the inner code and R–S as the outer code is that the TCM code has good noise-error performance at low signal-to-noise ratios (SNR), and a BER* versus SNR curve with a relatively slow roll-off. On the other hand, an R–S code has a very fast BER roll-off, but requires additional bandwidth. In a concatenated system, the bandwidth efficient TCM code is implemented to achieve high coding gain and to correct short bursts of interferences, such as NTSC synchronization pulses from an interfering analog TV channel in a nearby city†. The required output BER is around 10^{-3} to 10^{-4}. The R–S code is used to handle the burst errors generated by the inner TCM code and to provide a BER of 10^{-9} or lower. The drawback of the concatenated coding system is that it has a brick-wall or threshold type of performance degradation. The signal dropout is within 1 dB of SNR change around the threshold. Some argue that such a sharp "cliff" is inappropriate for broadcasting and is one of the serious shortcoming of digital broadcasting.

The concatenated coding can be applied in both SCM and MCM systems. With today's implementation of this channel coding technique, system carrier-to-noise ratio (C/N) for a BER $\leq 10^{-9}$ needs to be about 15 to 16 dB for the ATV system. An interleave and deinterleave is also typically used to exploit the error correction ability of FEC codes. Since most errors in the raw bitstream occur in bursts, they often exceed the error correction ability of the FEC code. Interleaving can be used to decorrelate these bursts into isolated errors that can be handled by the FEC code.

For SCM systems, multipath distortion results in intersymbol interference (ISI) and may cause loss of data. For MCM systems, if a guard interval is implemented and the multipath spread is less than the guard interval, there will be no intersymbol interference. However, there are still in-band fadings, which could cause severe amplitude/phase distortion for high-order QAM signals. In both SCM and MCM some measures must be taken to combat the multipath distortion. One way to reduce the multipath distortion is to use a high gain directional antenna. For a SCM system, a decision feedback equalizer is also used to minimize the multipath distortion. A training sequence can be embedded in the transmitted signal to assist fast convergence and system synchronization. However, any adaptive equalizer can also cause noise enhancement, which would decrease the system noise performance under a multipath scenario.

For a MCM system, the use of a guard interval can eliminate the ISI, but it also reduces data throughput. To minimize the loss of throughput, the size of the FFT must be increased. However, the size of the FFT is limited by digital signal processing (DSP) hardware capability and receiver phase noise. To compensate for in-band fading, a frequency domain equalizer can be used in combination with soft decision trellis decoding using channel state information. One of the distinct advantages of MCM over SCM with an adaptive equalizer is that MCM is not sensitive to variations in delay as long as the multipath falls within the

*Bit Error Rate

†Interference from a nearby city using the same channel is called *co-channel interference*. See Chapter 14.

guard interval and the interleave can effectively decorrelate the faded signal. Adaptive equalization performs better on short-delay multipaths than on long-delay multipaths. Therefore, MCM is potentially a better candidate for cell based networks, where there are long delay manmade multipaths. For a SCM system, a directional antenna must be used at some sites where strong multipath distortions are encountered.

The composite analog TV signal has a nonflat frequency spectrum, with three distinct carriers for luminance, chrominance, and audio. The luminance carrier has the highest power, of which the main contributor is the synchronization pulses. For an SCM system, an adaptive equalizer can be used to reduce the impact of co-channel analog TV interference. A few decibels gain of carrier-to-interference ratio (C/I) can be achieved when adaptive equalization is used. An adaptive equalizer can also exploit the cyclostationary nature of the co-channel ATV interference. Other approaches include the use of a comb filter to create spectrum notches to reduce interference at the expense of a 3 dB noise enhancement, and using spectrum gaps to avoid interference with the penalty of reducing throughput. An MCM system is vulnerable to co-channel analog TV interference because of its nonflat spectrum. MCM carriers colocated with interfering analog luminance, chrominance, and audio carriers may suffer from strong interference. To avoid this, some MCM carriers can be turned off to create spectrum "holes" with the penalty of reducing system throughput. This can take up a substantial fraction of the 6-MHz bandwidth. A better solution is to rely on an error correcting code to recover lost data. An interleaver and channel estimator combined with a soft decoding algorithm and error erasure could be used to combat co-channel analog TV interference. The performance of this approach has yet to be demonstrated.

For a SCM system, the peak-to-average power ratio depends on the channel filter roll-off. A faster roll-off, which has a higher spectrum efficiency, will result in a higher peak-to-average ratio. The typical peak-to-average ratio is about 7 dB. Some ATV systems take advantage of the nonsymmetric property of the analog VSB TV receiver input filter to transmit more power or to implement a pilot tone without increasing the co-channel interference into the analog channel of a nearby city. For an MCM system, the amplitude distribution is closer to Gaussian, having a relatively high peak-to-average ratio. Tests show that a SCM system is marginally better, by 1 to 2 dB, than a flat spectrum MCM system. However, if spectrum shaping is used, opening holes in the spectrum where analog carriers are normally located, a few decibels gain may be achieved by MCM systems.

12.5 Direct Broadcast Satellites[14]

A Direct Broadcast Satellite (DBS) system provides approximately 150 channels of direct-to-the-home TV in a compressed form that can be received by an 18-inch diameter dish antenna, decoded, and made ready for display to the consumer television set (see Fig. 12.11). In a little over a year since its introduction, more than 1 million direct broadcast satellite receivers have been installed in United States homes. Equally impressive, the entire receiver electronics system, including the antenna, low-noise front end, and set-top box are sold at less than $800 per system. The quality of video is generally comparable to that of broadcast or cable video.

The information flow from the video source to the satellite and back to the consumer's home is shown in Fig. 12.12. Video signals on earth from a variety of sources are compressed with the MPEG-2 algorithm, multiplexed perhaps using statistical multiplexing (see Chapter 8), and then sent using

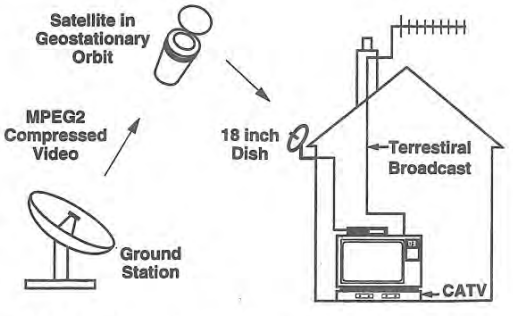


Fig. 12.11 Direct Broadcast Satellite System.

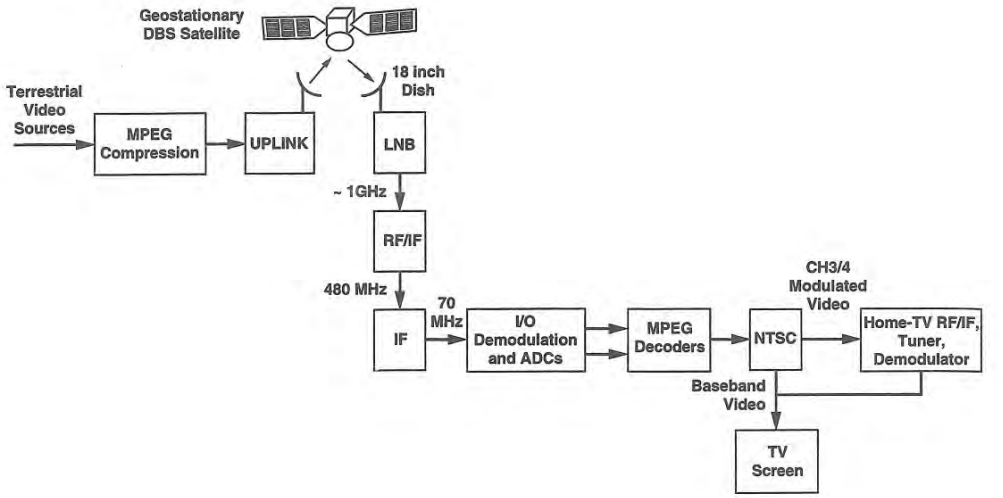


Fig. 12.12 Information Flow from Direct Broadcast Satellite to the consumer's home.

an existing uplink to a geostationary statellite. The signal representing this bitstream is amplified by transponders within the satellite and beamed directly to the small dish antenna at each consumer's home. The modulation technique that is used often is Quadrature Phase-Shift Keying (QPSK), which provides 2 bits per symbol. The modulator creates In-phase (I) and Quadrature (Q) components of bitstreams which are then added before up-conversion to a higher frequency for transmission. Forward Error Correction (FEC) is required for noise tolerance and correction of transmission errors. Convolution (trellis) encoding provides protection against short impulsive error bursts. Reed–Solomon block codes help detect and correct longer duration errors.

The first of the DBS satellites currently in orbit was launched in December, 1993. It carries 16 transponders. Each transponder has a 120W output,* about 10 to 20 times the output of a typical communications satellite designed for a much larger receiver antenna. The transponder and antenna gain on the satellite yield a downlink radiated signal of more than 50 dBW, depending on receiver location. The receiver's parabolic antenna provides a gain of approximately 30 dB at 12 GHz. The antenna must be installed with a clear view of the southern horizon and pointed at the satellite, which is geosynchronously stationed at 101° W longitude and aimed at the continental United States.

In spite of the high power and high antenna gain of the transponders and the satellite, high performance is still required of the front-end receiver circuitry. The gain of the receiver antenna partially compensates for the 200 dB path loss incurred at 12 GHz over the 37,000 km synchronous orbit distance. The DBS receiving antenna uses a Low-Noise Block (LNB) mounted at the focal point of the dish to provide gain for the approximately −100 dBm received signal. The RF and IF stages of the receiver circuitry are designed by balancing the allocation of gain, noise, and cost among various blocks, while providing protection against overload, which can easily occur at signal levels higher than −80 dBm. The low-noise block is more than just an amplifier. It also down-converts the 12 GHz untuned signal to a range of 950 to 1450 MHz. The next stage is to further down-convert the signal to a single channel at around 70 MHz and apply synchronous demodulation to get both the In-Phase (I) and Quadrature (Q) digital

*An alternative configuration allows eight transponders to operate at 240 W.

components. Such a tuned signal is then ready for data recovery.

The processing performed in the set-top box does the QPSK demodulation to get a symbol rate of approximately 21 MHz, which produces a raw bitrate of about 42 Mbits/s. The output data from the demodulator is then fed to an FEC decoder, which incorporates a Viterbi decoder concatenated with a Reed–Solomon decoder. The final error-corrected payload is either 23 Mbits/s or 30 Mbits/s, depending on configuration and FEC settings.

The output of the FEC decoder is fed to an MPEG Systems demultiplexer, followed by the audio/video decompression devices. The resulting *Y Cb Cr* video components are then usually combined to produce a composite NTSC signal. This is then either passed directly to the baseband inputs of the TV or up-converted to channel 3 or 4 and passed to the RF antenna inputs.

The current Direct Broadcast Systems distribute many of the same programs seen on the cable television systems. However, there is no local programming. In addition, DBS is able to show special interest and special event programming. An attractive user-friendly interface is created on a channel that is reserved for on-screen menus and selection guides. The growth of the DBS system in the United States has been much higher than expected. The early experience in Europe and Japan also shows significant promise. With the cost of the receiver electronics coming down, we expect strong demand and growth of direct broadcast satellite services.

REFERENCES

1. R. C. Menendez, D. L. Waring, and D. S. Wilson, "High-Bit-Rate Copper and Fiber-to-the Curb Video Upgrade Strategies," *Proc. IEEE International Conference on Communications ICC* '93 (May 23–26, 1993).
2. Lipson et al. "High Fidelity Lightwave Transmission of Multiple AM-VSB NTSC Signals," *IEEE Trans. Micro. Theory Techniques,* 38:483–493 (1990).
3. Olshanksky et al., "Subcarrier Multiplexed Broad-Band Service Network: A Flexible Platform for Broad-Band Subscriber Services," *J. Lightwave Technol.,* 11:60–69 (1993).
4. R. Jones and R. B. Sharpe, "A Cost Effective Digital Broadband Passive Optical Network," *Proc. NFOEC,* Nashville, TN (1991).
5. X. P. Mao et al., "Brillouin Scattering in Lightwave AM-VSB, Transmission Systems," in *Tech. Dig. Opt. Fiber Comm.,* San Jose, CA (February, 1992).
6. S. D. Dukes, "Photonics for Cable Television System Design," *Commun. Eng. Design,* pp. 34–48 (May 1992).
7. L. Ellis, "The 5–30 MHz Return Band: A Bottleneck Waiting to Happen?," *Commun. Eng. Design,* pp. 40–43 (December 1992).
8. W. Shumate, "Economic Considerations for Fiber to the Subscriber," *Proc. 1993 European Conf. in Optical Communication,* 1:120–123 (1993).
9. J. A. Chiddix, "Fiber Optic Technology for CATV Supertrunk Applications," in *NCTA '85 Technical Papers,* 1985.
10. J. A. Chiddix, "Optical Fiber Super-Trunking, The Time Has Come: A Performance Report on a Real-World System," in *NCTA '86 Technical Papers, 1986 and IEEE Journal on Selected Areas in Communications,* SAC-4 (5) (August 1986).
11. P. A. Rogan, R. B. Stelle, III and L. D. Williamson, "A Technical Analysis of a Hybrid Fiber/Coaxial Cable Television System," in *NCTA '88 Technical Papers,* 1988.
12. D. L. Waring and W. Y. Chen, "Applicability of ADSL to Support Video Dial Tone in the Copper Loop," *IEEE Commun. Mag.* 32 (5):102–109 (May 1994).
13. W. F. Schruber, "Advanced TV Systems for Terrestial Broadcasting: Some Problems and Some Proposed Solutions," *Proc. IEEE,* pp. 958–978 (June 1995).
14. J. Wood, *Satellite Communications DBS Systems,* Focal Press (1992).

13

INTERACTIVE TELEVISION

Interactive television (ITV) is a new form of digital consumer multimedia service that can give viewers much greater control over the content of the programs than is possible with conventional analog television.[1–3] Advances in audio/video compression, multimedia database systems, ongoing deployment of broadband networks, inexpensive home terminals, and user-friendly interfaces will provide the infrastructure for offering a wide variety of interactive video services to consumers at home via standard television. Such services include video on demand, video telephony, multimedia information retrieval, distance-education, home shopping, and multiplayer, multilocation video games, and so forth. This chapter deals with a platform over which many of these services can be implemented.

The most important aspect of interactive television is the ability of the viewer to exercise both coarse and fine-grain control over the contents of the programming being viewed. By comparison, current television allows the viewer to select only among a number of channels that broadcast a predetermined presentation to a large audience. This presentation follows a predefined sequence from start to finish and cannot be customized in any manner by an individual viewer. In interactive television, viewers can directly control the content displayed on their TV screens. With a proper

only a program, but also other related information that is referenced in the program. The selected multimedia material could contain still images, moving video, and audio. Creating a customized entertaining experience with audio/video whose quality is comparable to state-of-the-art conventional TV programming is the challenge of interactive TV. Computer video games, for example, have always been interactive, but in most cases the pictures that were controlled were cartoon-like. The quality of these pictures has been steadily rising as the result of the growing programming sophistication, inexpensive hardware, and the desire of the viewers to see exactly what they want in a time frame that they can allocate among other competing demands on their time.

ITV programs generally consist of a collection of media elements together with software that controls the program flow in response to consumer inputs and directs how the media elements create the multimedia presentation. Media elements, such as audio and video clips, still images, graphics, text, etc., are the primitives of the presentation and are created as part of the process of producing the ITV application.

Each viewer of an ITV application is totally independent. This requires that a separate instance of an application be executed by the ITV system for each viewer, although the stored or synthesized

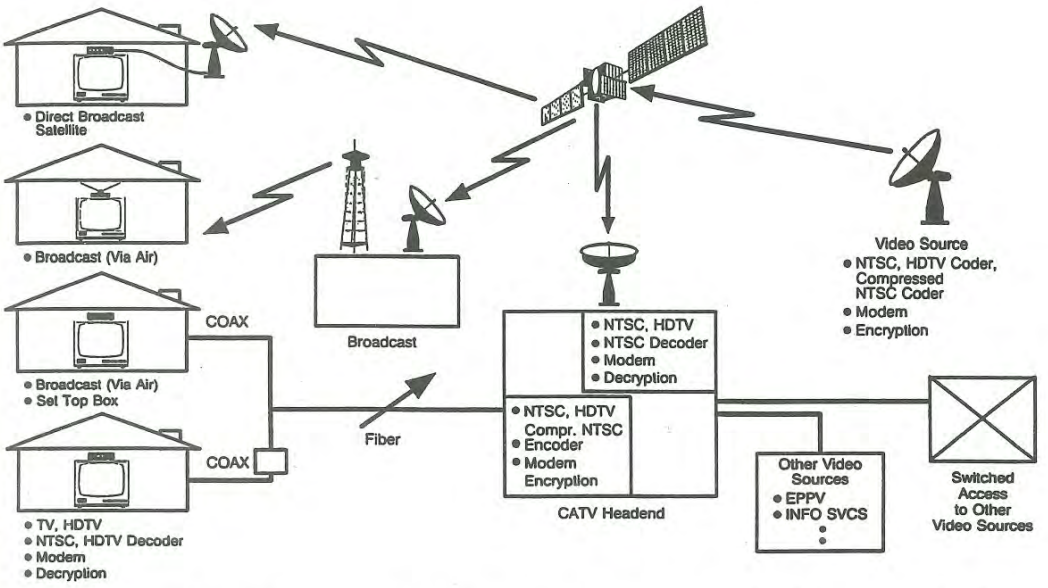


Fig. 13.1 This block diagram shows a variety of services such as video-telephony, games, and ITV. The system consists of servers, a distribution network, and the set-top box (customer-premise equipment, CPE).

cations bandwidth must be allocated for each viewer, a situation that requires a logical star topology similar to the telephone network rather than the shared bandwidth topology of the conventional broadcast or cable TV (see Chapter 12).

Economically providing the appropriate computing, storage, and communication facilities required for each viewer of an ITV system is the key enabler for such services. An ITV system deals with data types (e.g., audio and video clips) that are typically large (even after compression) and must be processed within real-time constraints. The data throughput or bandwidth required for each viewer can be 25 to 100 times that required for standard voice telephony.

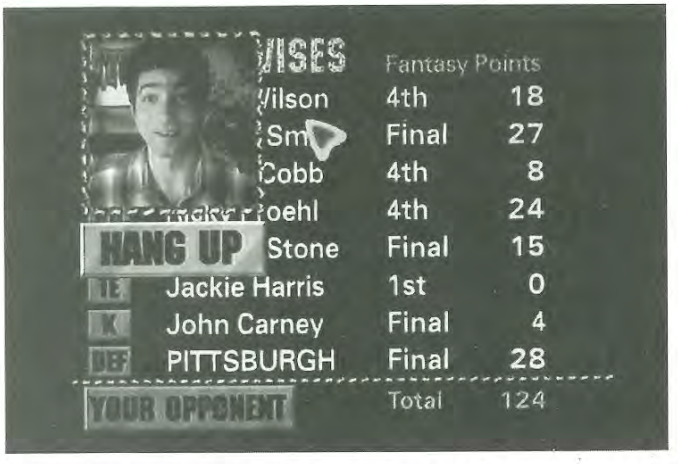


Fig. 13.2 An experimental fantasy sports league application. A videophone call is in progress in the window at the upper left corner. The rectangular area labeled "Hang Up" is a button to be clicked by the user (via the remote control) to terminate the call.

Fig. 13.3 A screen from an experimental children's educational application. The areas labeled "Zap, Erase, Clue, Loot" and the boxed letters *N, L, G, T, I,* and *H* are all buttons. The button *N* is shown highlighted after being clicked.

Fig. 13.4 A screen from an experimental home shopping application. As the user clicks the buttons labeled with manufacturer names, video presentations about various juice extractors are played in the window to the right. Clicking the button labeled Consumer Service will initiate a telephone call (audio or video) to a customer sales representative who can answer questions about the products.

The four main components of interactive television systems are: (1) a home terminal, commonly known as the set-top box (STB) or customer premises equipment (CPE), (2) an access network, (3) a network-based server, and (4) a powerful user-friendly interface. Figure 13.1 shows a typical configuration of an ITV system. Figures 13.2 to 13.4 show typical video screens from an experimental ITV system.[5] In this chapter, we discuss the CPE, network-based servers, and user interface. The access network has

13.1 CUSTOMER PREMISES EQUIPMENT

The cost of the CPE is a dominant factor in the overall cost of the ITV system. The less expensive the CPE, the more likely it is that people will try the service and become repeat users. The CPE (or STB) typically takes the form of a box sitting on top of the TV set.[4] This CPE connects to both the television and to an external communication net-

satellite) via a subscriber "drop" or "loop" from an outside pole, underground conduit, or direct broadcast antenna. Each TV set in a home typically requires its own STB.

For standard, noninteractive, analog TV programs delivered via a CATV plant, a STB includes analog frequency translation, program descrambling circuitry, and a simple wireless remote control. Such a STB has a very low cost and is often owned and supplied by the network operator.

When interactive services are offered, the complexity and cost of the STB increases. Since the digital services are new "add-ons," the new STB must provide both the analog functions outlined above as well as the digital functions of audio and video decompression, demodulation to recover the digital feeds, decryption, an upstream modem for communicating consumer control requests back to the program source, plus a user-friendly interface. This type of STB, for use on a CATV plant, often costs several times the basic analog STB. However, this cost is expected to go down in the future.

The basic interactive STB just described is capable of supporting a variety of interactive video services. Among them are the popular ones such as movies-on-demand and home shopping. To support these services, a source of interactive service programming (called a server) sends an individual, digitally compressed bitstream to the STB of the consumer currently using a particular service provided from that server. As noted previously, each feed will be distinct because each consumer may be viewing a different portion or a different presentation of the program.

The bitstream representing the audio/video program sent to the basic interactive STB needs to be decompressed and fed to the TV set. All of the processing required to create the real-time interactive program is usually performed at the program server. This might include the composition of multiple audio and video elements to create the final presentation. This partitioning of functionality assures that the cost of the STB is minimized. Sharing of network-based functionality to reduce CPE, and thereby the total system cost, is common in the network-based services industry, including telephony. However, this strategy of reducing CPE costs by sharing functionality in the network can restrict the types of interactive applications that can be effectively supported. Highly interactive applications, such as rapid reaction video games (called "twitch" games), require extremely low latency from the time a consumer presses a button until the effect is seen on the screen.

A consumer's remote control input is transmitted from the STB back to the server in the network. This input can change the flow of the program being executed by the server on behalf of that consumer. The audio and video bitstream produced by the server is compressed, often in real-time,* and modulated for transmission to the consumer's STB. The STB then simply demodulates the received signal to recover the transmitted bitstream and decompresses it to create the presentation on the TV set.

The principal latency in this architecture does not arise from the transmission time of either the control input from the STB to the server or from the transmission of the video from the server to the STB. The latency arises because compression schemes, like MPEG, require one or more frame delays in the encoding process plus additional frame delays in the decoder. In interactive graphics, 100 ms, or roughly 3 NTSC frame times, is typically regarded as the upper bound on latency for maintaining the consumer's feeling of instant control response. To reduce the decompression latency, the STB could support the local execution of highly interactive graphical applications. In this case, the STB has direct access to the locally synthesized graphics without the delay imposed by compression and decompression for transmission from the server. However, local execution of applications requires additional capabilities in the STB, with a corresponding increase in cost. At a minimum, it requires a processor with memory for program and data storage. Additional capabilities could include sound generators, video layering, special effects, control ports, and so forth. The fea-

*In many instances the server may store bitstreams compressed previously in non-real-time. In some other instances, an uncompressed bitstream may be sent to the STB.

tures that can be added to an intelligent high-end STB are essentially unlimited, and such a STB could resemble a high-end multimedia personal computer with a modem, but without the display (since a TV is used as the display).

In addition to the reduction in interactive latency, STBs with significant local processing power can assume more of a client–server relationship with the server as opposed to the source-sink model of the basic STB and server. The client–server relationship complicates the development of applications and introduces potential security and authentication problems. However, it can define open interfaces between the different software modules (in the server and the client) and therefore offers flexibility in developing applications by multiple vendors. As the cost of digital hardware continues to decline, the price difference between the basic and high-end STBs will shrink significantly. In the long run, all STBs will have substantial capability, and the functionality of the basic STB will be incorporated directly into television sets. With further increases in local processor power, it will be possible for most functions of a STB, including decompression, modems, graphics, and so forth to be implemented entirely in software.

In the telephone network, the network–CPE interface is well defined and open, enabling consumers to purchase or lease telephone sets with features to match their needs. On the other hand, most CATV system operators view the STB as part of a network, and lease STBs to their customers as part of their service package. STBs used in CATV systems differ in their frequency plans, scrambling and address control systems, signaling, and modem capabilities. This approach has limited the services available to the consumers to only those supplied by the CATV operator. It has led to unfortunate situations where some features of expensive TVs, such as picture-in-picture, cannot be used on a CATV system, and consumers cannot use a VCR to record one program while watching another. If an interface between a CATV network and the STB were well defined and open, consumer electronics firms could manufacture STBs with a variety of feature sets to meet different consumer needs, or build the basic STB functionality directly into the TV set to lower the cost.

13.2 INTERACTIVE TELEVISION SERVERS

ITV servers are a collection of computing, storage, and communications equipment that implement interactive video services. A service may require more than one server to be implemented, or a server may implement more than one service. The overall architecture of a server is shown in Fig. 13.5. All subsystems of the server communicate with each other via a local high-bandwidth interconnect and a switch. This architecture can be used to scale the capabilities of the server incrementally, and it provides isolation between the various subsystems.

A storage subsystem stores the media (audio/video/data) samples. It may use different combinations of tapes, disks, and semiconductor storage technologies to meet the capacity and latency requirements of the various applications. As discussed later, these storage technologies are typically combined into a hierarchy to optimize performance for a given cost.

Application processors interact with the storage subsystem to deliver the output of various media to the subscribers' CPE or to compose media for eventual presentation. Media composition subsys-

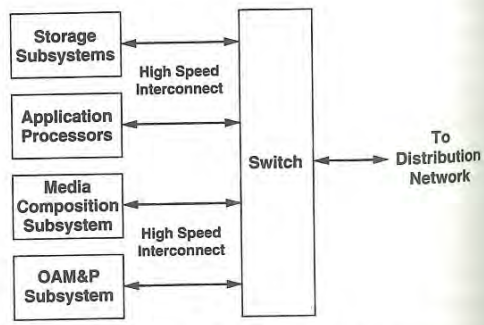


Fig. 13.5 Architecture of ITV server includes several subsystems optimized for specific functions and a high-speed network connecting them. OAM&P refers to operations, administrated maintenance, and provisioning.

tems use media samples from the storage subsystem, along with control commands from the application processors, to create a single composite compressed audio and video bitstream as required by each STB.

As mentioned previously, when driving basic low-cost STBs, servers emulate compressed digital program sources. Here the servers generate compressed audio and video streams in standard formats that require only decompression in the STB. Consumers' remote control button-press signals are carried back to the server for interpretation by the service application. With larger processing power, STBs will be able to combine and manipulate media samples. In such cases, the server need only send primitive media elements to the STB to be processed under the control of the locally executing services application.

A special class of servers called the gateway server (see Fig. 13.1) provides the navigation and other functions required to support the consumer's interface for the selection of services. These servers present "menus" of available services (e.g., movies-on-demand, home shopping, and broadcast television), accept the consumer's selection, and then hand off control to the server providing the selected service. The visual interface and navigation scheme of a gateway server are completely programmable. A high-end interactive media server can present a very rich full-motion audio/video interface, whereas a low-end system can provide simple text-based menus. Gateway servers allow customization by individual consumers of the on-screen appearance and operation of the system (see Fig. 1.5). Consumers might simply designate their favorite services to be shown first or they might engage "intelligent agents" to notify them of programs or events of potential interest. This could also provide different views of the system for each individual in a household. For example, when the TV is turned on with a child's remote control, the system could show only programs or services suitable for children.

Two key components of the server technology are (1) the logical organization of the multimedia samples in a file system or database and (2) techniques by which media components can be "continuously" (real-time) recorded or played back from the server. We review these two aspects below.

13.2.1 Multimedia Data Storage

Multimedia storage systems are characterized by many new capabilities as compared to standard ASCII storage systems (either files or databases); for example, massive storage volumes, large objects, rich object types, more flexible relationships among objects, temporal composition of media objects, synchronization of various media for effective presentation, etc. Among the choices for logical organization of the multimedia data are:

- Multimedia file systems
- Extended relational database management systems (RDBMS), which support multimedia as binary objects and in some cases support the concepts of inheritance and classes
- Object-oriented databases.

13.2.1.1 Multimedia file systems

File systems provide support for storing data in most computer systems. Interfaces between file systems and other parts of the software system of a computer are well known, understood, and continue to evolve. This is true of both single standalone computers as well as networked computers with a shared-network file system. In cases where additional functionality is desired (e.g., concurrent transactions, flexible queries, indexing, report generation) database are used.

There are many disadvantages to file systems when they are used as DBMSs:

- They do not provide physical data independence. Application developers must be intimately familiar with the physical layout of data.
- They do not provide backup or recovery mechanisms beyond those provided by the underlying operating system, which are often inadequate.
- They are not supported by a query language such as the Structured Query Language (SQL). Consequently, application programs must manipulate the file data using lower-level programming languages.
- They are not very portable, since file systems vary among different operating systems.

However, the additional functionality provided by a database system comes at the expense of sub-

stantial overhead. File systems are therefore used when the above functionality is not required or when the overhead of a database system cannot be tolerated. Typically, movies-on-demand systems, where records are almost always accessed in the same sequence, may be implemented via a file system. To satisfy the stringent requirements of continuous media, many modifications are necessary. Much work has been done in this area and a number of implementations of file systems exist that can support multimedia.

13.2.1.2 Relational Database Management Systems (RDBMS) for Multimedia

Relational databases are a collection of tables (called relations) containing data supplied by the user. Data are extracted by composing queries that "cut and paste" selected tables. Rules for cutting and pasting are based on relational algebra and are embodied as a query language (e.g., SQL). Standard database features include functionality such as concurrency control, recovery, and indexing. Relational databases have traditionally been designed to operate mostly with small fields (large multimedia objects will lose efficiency in a DBMS). Also the shared data is usually of a single type, namely ASCII characters.

The key limitations of relational databases are in two areas: the relational data model and the relational computational model. Relational databases have been extended by incorporating an additional data type, commonly known as a *binary large object* (BLOB), for binary free-form text images or other binary data types. Relational database tables include location information for the BLOBs, which may be stored outside the database on separate servers, thereby extending the database to access these BLOBs. Extended relational databases provide a gradual migration path to more object-oriented databases. Relational databases also have the strength of rigorous management for maintaining the integrity of the database, an important feature that has been lacking in early object-oriented databases. A number of database vendors, with a large invested base in relational database products, offer a variety of extensions to handle multimedia pointers to continuous media data (i.e., audio and video).

13.2.1.3 Object-Oriented Database Management Systems (OODBMS) for Multimedia

Object-oriented database management systems incorporate a collection of user-defined object types (called *classes*). For each class, appropriate operations and data structures are defined either by the system or by the user. Most object databases are modeled around the class paradigm of the object-oriented languages (such as C++ or small talk). In the context of multimedia, OODBMS provide the following useful characteristics:

- The natural ability to express relationships between objects in different classes (e.g., relationships between component audio and video objects to create a "composite" multimedia presentation).
- Unlike in a relational database, there is no mismatch between the application's and the database's view of the world, since both have the same data model.

Different components (e.g., a sequence of frames representing a scene in a movie) of an MPEG-2 bitstream can be stored as different objects. This will allow the original MPEG-2 stream to be customized by selecting a subset of the components. An object-oriented database supports both *encapsulation* (the ability to deal with software entities as units) and *inheritance* (the ability to create new classes derived in part from existing object classes). These concepts constitute the fundamental tenets of the object-oriented paradigm.

The class definition in the object model has special applicability for multimedia data. Text, audio, still images, and video can each be of a different object class. Stated differently, the class definitions associate specialized processing to typed data, that is, the ways in which the various primitive media types are accessed and used. OODBMS also provide advantages in terms of the speed and a wider range of capabilities for developing and maintaining complex multimedia applications. OODBMS capabilities, such as extensibility, hierarchical struc-

timedia applications. *Message passing*, for example, allows objects to interact using each other's methods, which can be customized by the programmer.

Complex "composite" objects can be constructed that collect various "primitive" media objects such that the desired presentation and interaction are captured in the composite object. OODBMS are *extensible* (i.e., users can define new operations as needed) and allow incremental changes to the database applications. These changes would be more difficult to express in a relational language (e.g., SQL).

Summing up, the object-oriented paradigm offers several advantages for multimedia systems:

- Databases that understand the types of multimedia objects and will ensure that appropriate operations are applied to each class.
- Easy to manipulate component multimedia objects that allow the making of a composite multimedia presentation.
- Facilities for the user to customize object classes and processing methods for each class
- Better sharing of component media objects and therefore the need to store only one copy for each media object.

The OODMS must be capable of indexing, grouping, and storing multimedia objects in distributed hierarchical storage systems, and accessing these objects on a keyed basis. A multimedia object (e.g., a video presentation) may be a component of a multiple hypermedia* document such as memos, presentations, video sales brochures, and so forth. The design of the object management system should be capable of indexing objects in such a manner that there is no need to store multiple copies.

13.2.2 Recording and Retrieval for Multimedia[5-9]

Digital audio/video is a sequence of quanta (audio samples or video frames). Retrieval and presentation of such quanta require continuity in time; that is, a multimedia database server must ensure that the recording and presentation follows the real-time data rate. During presentation, for example, the server must retrieve data from the disk at a rate that prevents the output device (such as a speaker or video display) from overflowing or running out of samples.

To reduce the initiation latency and buffering requirement, continuous presentation requires either (1) a sequence of periodic tasks with deadlines, or (2) retrieval of data from disks into buffers in rounds. In the first approach, tasks include retrieval of data from disks, and deadlines correspond to the latest time (e.g., MPEG Systems Time Stamps) when data must be retrieved from disks into buffers to guarantee continuous presentation (see Fig. 13.6). In the latter approach, in every round sufficient data for each media stream is retrieved to ensure continuous presentation. The server has to supply the buffers with enough data to ensure continuity of the playback processes. Efficient operation is then equivalent to preventing buffer overflow/underflow while minimizing buffer size and the initiation latency. Since disk data

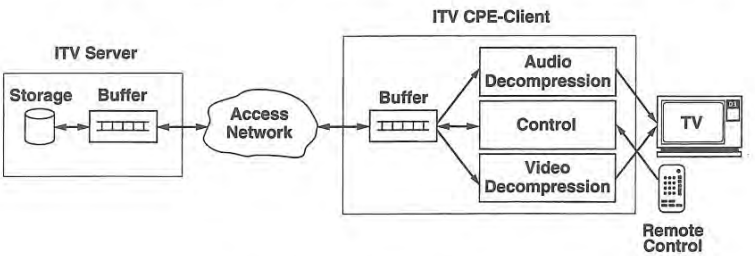


Fig. 13.6 Since the data flow is bursty owing to varying compression ratio and jitter in the storage system, buffers are required in the CPE and the server to ensure smooth data flow to the presentation hardware (i.e., TV).

*Hypermedia documents contain pointers to other documents or to other places in the current document.

transfer rates are significantly higher than a single stream's data rate, a small buffer should suffice for conventional file and operating systems support for several media streams.

The real challenge is in supporting a large number of streams simultaneously. ITV servers must process retrieval requests from several CPEs simultaneously. Even when multiple CPEs access the same file (such as a popular movie), different streams might access different parts of the file at the same time. Since disk I/O rates significantly exceed those of single streams, a larger number of streams can be supported by multiplexing a disk head among several streams. This requires careful disk scheduling so that each buffer is adequately served. In addition, new streams should not be admitted unless they can be properly served along with the existing ones.

13.2.2.1 Disk Scheduling

Database systems employ disk scheduling algorithms, such as "first come, first serve" and "shortest seek time first," to reduce seek time and rotational latency, to achieve high throughput, and to enforce fairness for each stream. However, continuous media constraints usually require a modification of traditional disk-scheduling algorithms so they can be used for multimedia servers.

One commonly used disk scheduling algorithm, called the *Scan-EDF* (Earliest Deadline First), services the requests with earliest deadlines first. However, when several requests have the same deadline, their respective blocks are accessed with a shortest-seek-time algorithm. Scan-EDF becomes more effective as the number of requests having the same deadline increase. This happens more frequently as the ITV server services a larger number of CPEs.

If data are retrieved from disks into buffers in rounds, a commonly used disk scheduling algorithm is *C-LOOK* (Circular LOOK), which steps the disk head from one end of the disk to the other, retrieving data as the head reaches one of the requested blocks. When the head reaches the last block that needs to be retrieved within a round, the disk head is moved back to the beginning of the disk without retrieving any data, after which the

13.2.2.2 Buffer Management

Most ITV servers process multimedia stream requests from the CPE in rounds. During a round, the amount of data retrieved by the buffer could be made equal to the amount consumed by the CPE. This means that, on a round-by-round basis, data production never lags display, and there is never a net decrease in the delay of buffered data. Therefore, such algorithms are referred to as buffer-conserving. A non-buffer-conserving algorithm would allow retrieval to fall behind display in one round and compensate for it over the next several rounds. In this scenario, it becomes more difficult to guarantee that no starvation of the display will ever take place.

For continuous display, a sufficient number of blocks must be retrieved for each CPE-client during a round so that starvation can be prevented for the round's entire duration. To determine this number, the server must know the maximum duration of a round. As round length depends on the number of blocks retrieved for each stream, unnecessarily long reads must be avoided. To minimize the round length, the number of blocks retrieved for each CPE during each round should be proportional to the CPE's display rate.

13.2.2.3 Admission Control

To satisfy the CPE's real-time requirements, an ITV server must control admission of new CPEs so that existing CPEs can be serviced adequately. Strict enforcement of all real-time deadlines for each CPE may not be necessary since some applications can tolerate occasional missed deadlines. A few lost video frames or a glitch in the audio can occasionally be tolerated, especially if such tolerance reduces the overall cost of service. An ITV server might accommodate additional streams through an admission control algorithm that exploits the statistical variation in media access times, compression ratios, and other time-varying factors.

In admitting new CPEs, ITV servers may offer three categories of service:

- *Deterministic.* Real time requirements are guaranteed. The admission control algorithm must con-

- *Statistical.* Deadlines are met only with a certain probability. For example, a CPE may subscribe to a service that guarantees meeting deadlines 95% of the time. To provide such guarantees, admission control algorithms must consider the system's statistical behavior while admitting new CPEs.
- *Background.* No guarantees are given for meeting deadlines. The ITV server schedules such accesses only when there is time left after servicing all guaranteed and statistical streams.

In servicing CPEs during a round, CPEs with deterministic service must be handled before any statistical ones, and all statistical-service CPEs must be serviced before any background CPEs. Missed deadlines must also be distributed fairly so that the same CPE is not dropped each time.

13.2.2.4 Physical Storage of Multimedia

A multimedia server must divide video and audio objects into data packets on a disk. Each data packet can occupy several physical disk blocks. Techniques for improving performance include optimal placement of data packets on the disk, using multiple disks, adding tertiary* storage to gain additional capacity, and building storage hierarchies.

Packets of a multimedia object can be stored contiguously or scattered across the storage device. Contiguous storage is simple to implement, but subject to editor fragmentation. It also requires copying overheads during insertions and deletions to maintain contiguity. Contiguous layouts are useful in mostly-read servers such as video-on-demand, but not for read-write servers. For multimedia, the choice between contiguous and scattered files relates primarily to intrafile seeks. Accessing a contiguous file requires only one seek to position the disk head at the start of the data, whereas with a scattered file, a seek may be required for each packet read. Intrafile seeks can be minimized in scattered layouts by choosing a sufficiently large packet size and reading one packet in each round. It improves disk throughput and reduces the overhead for maintaining indexes that a file system requires.

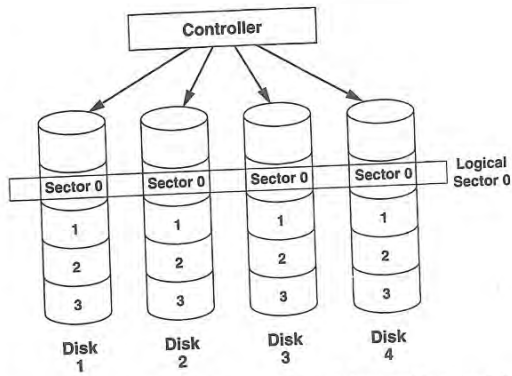


Fig. 13.7 Data is striped across several disks and is accessed in parallel.

To increase the number of possible concurrent access of a stored multimedia object, it may be scattered across multiple disks. This scattering can be achieved by using two techniques: data striping and data interleaving. RAID (redundant array of inexpensive disks) technology (see Fig. 13.7) uses an array of disks and stripes the data across each disk. Physical sector "n" of each disk in the array is accessed in parallel as one large logical sector "n." In this configuration, the disks in the set are spindle synchronized and operate in lock-step mode. Thus, the transfer rate is effectively increased by the number of disk drives employed. Striping does not improve the seek time and rotational latency, however.

In data interleaving, the packets of multimedia objects are interleaved across the disk array with successive file packets stored on different disks. A simple interleave pattern stores the packets cyclically across an array of N disks. In data interleaving, the disks in the array operate independently without any synchronization. Data retrieval can be performed in one of two ways: (1) One packet is retrieved from each disk in the array for each stream in every round or (2) data are extracted from one of the disks for a given CPE in each round. Data retrieval for the CPE cycles through all disks in N successive rounds. In each round, the retrieval load can be balanced by staggering the streams. Thus, all streams still have the same round

*The first two storage media are buffer RAM and main disk.

length, but each stream considers the round to begin at a different time.

A combination of striping and interleaving can be used to scatter a file in many disks from a server cluster. This technique is often used in constructing a scalable video that can serve a large number of CPEs accessing the same copy of the file.

The reliability of the server will become an important issue as the number of users who rely on ITV services increases. A major failure point in the server is the storage subsystem. With the increase in the number of disks, the reliability of the server decreases. Even if the mean time between failures (MTBF) of a single disk is between 20 and 100 years, as the number of disks increases, the MTBF of the disk subsystem decreases to weeks. This is because the MTBF of a set of disks is inversely proportional to the number of disks, assuming that the MTBF of each disk is independent and exponentially distributed. Further, owing to disk striping, the number of files affected per failure increases.

To provide continuous, reliable service to CPEs, the disk subsystem should employee redundancy mechanisms such as data replication and parity. However, these common techniques need to be extended for ITV servers so that in the presence of a failure all the admitted CPEs can be still serviced without interruption.

13.2.2.5 Updating ITV Servers

Copying of multimedia objects is expensive because their size is large. Copying may be reduced by considering object files to be immutable. Editing is then enabled by manipulating tables of pointers to the media component objects. Obviously, small insertions will not find pointers as valuable as large ones. Also, for deletion, if the multimedia object is an integral number of disk blocks, then the file map could simply be modified.

It is possible to perform insertion and deletions in a time that is proportional to the size of the data inserted/deleted rather than to the whole file. A common scheme is where packets must be filled to a certain minimum level to support continuous retrieval. The insertion/deletion will then consist of some number of full packets plus a remaining partially filled packet. All the packets are then inserted/deleted by modifying the file map, and then data are distributed among adjacent packets of the file to meet the required fill level.

13.2.2.6 VCR-like Functions

An ITV server must support VCR-functions such as pause/resume, fast-forward, and fast-rewind. The pause/resume function is a challenge for buffer management because it interferes with the sharing of streams among different viewers. The fast-forward and fast-rewind functions are implemented by playing back at the normal rate while skipping packets of data. This is easy to do by skipping from one I-frame to another; otherwise interframe dependencies complicate the situation.

The simplest method of fast-forward is to display only I-frames. However, this reduces flexibility. Another way is to categorize each frame as either relevant or irrelevant to fast-forward. During normal operation, both types of frames are retrieved, but during fast-forward, only the fast-forward frames are retrieved and transmitted to the CPE-buffer. It is assumed that the decoding of fast-forward blocks by CPE is straightforward. Scalable compression schemes are readily adapted to this sort of use, although the drawback here is that it poses additional overhead for splitting and recombining blocks.

13.3 USER INTERFACES[10-12]

User interface designs for ITV are more involved than for standard TV owing to the richness of the types of possible interactions with the consumer. As mentioned earlier, they need to be extremely easy to use, as well as intuitive. They will most likely vary among different applications. There will be a small number of ITV generic program types, such as movies-on-demand, shopping, and so forth, and certain "look-and-feel" guidelines will emerge for user interfaces and for ITV programming in general. Typically, multimedia interfaces are designed using a mixture of four kinds of modules:

1. Media editors
2. Multimedia object locators and browsers (navigation)
3. Authoring tools for creating program logic
4. Hypermedia object creation.

The last item is currently less relevant to ITV and is therefore not discussed here. However, it may become more widely available in the future. A number of media editors already exist to create "primitive" media objects such as text, still images, and video and audio clips. A media editor is then used for editing primitive multimedia objects to construct a composite object that is suitable as a multimedia presentation. The basic functions provided by the editor are delete, cut, copy, paste, move, and merge. The real challenge is to present these basic functions in the most natural manner that is intuitive to the user. For example, the move operation is usually represented by highlighting text and dragging and dropping it to the new location. However, the same metaphor does not work for all multimedia objects. The exact implementation of the metaphor and the efficient implementation for that metaphor will distinguish such editors from each other.

Navigation refers to the sequence in which the objects are browsed, searched, and used. Navigation can be either direct or predefined. If direct, then the user determines the next sequence of actions. If predefined, for example, searching for the next program start, the user needs to know what to expect with successive navigation actions. The navigation can also be in a *browse* mode, in which the user wants to get general information about a particular topic. Browsing is common in systems with graphical data. ITV systems normally provide direct navigation as well as browse options.

The browse mode is typically used to search for a specific attribute or a subobject. For example, in an application using dialog boxes, the user may have traversed several dialog boxes to reach a point where the user is ready to play a sound object. A browse mode may be provided to search the database for all sound objects appropriate for that dialog box. An important aspect of any multimedia system is to maintain a clear perspective of the objects being displayed and the relationship between these objects. The relationship can be associative (if they are a part of a set) or hierarchical. Navigation is undoubtedly the first generic application for ITV. Various flavors have been created, for example, mapping of ITV programs to channel numbers, and interfaces that show "picture-in-picture." Navigation is rightly the first application focus, acknowledging the real challenge ahead in helping viewers find their way in the (hoped for) proliferation of programming in the interactive world.

The interactive behavior of ITV determines how the consumer interacts with the services available. Some of the design elements are:

- Data entry dialog boxes
- A source-specific sequence of operations depicted by highlighting or enabling specific menu items
- Context-sensitive operation of buttons
- Active icons that perform ad hoc tasks (e.g., intermediate save of work).

The look and feel of a service depends on a combination of the metaphor (e.g., VCR interface) being used to simulate real-life interfaces, the windows guidelines, the ease of use, and the aesthetic appeal. An elegant service has a consistent look and feel. The interface will be more friendly if, for example, the same buttons are found in the same location in each dialog box and the functions of an icon toolbar are consistent. User interfaces for multimedia applications require careful design to ensure that they are close adaptations of the metaphors to which consumers are accustomed in other domains. Two metaphors relevant to ITV are the Aural and VCR interfaces described below.

A common approach for speech recognition (also called the Aural metaphor) user interfaces has been to graft the speech recognition interface into existing graphical user interfaces. This is a mix of conceptually mismatched media which makes the interface cumbersome. A true aural user interface (AUI) would accept speech as direct input and provide a speech response to the user's actions. The real challenge in designing AUI systems is to creatively substitute voice and ear for the keyboard and display, and to be able to mix and match them. Aural cues can be used to represent icons, menus, and windows of a GUI.* Besides computer technology, AUI

*Graphical User Interface

designs involve human perception, cognitive science, and psychoacoustic theory. The human response to visual and aural inputs is markedly different. Rather than duplicate the GUI functions in an AUI, this requires new thinking to adapt the AUI to the human response for aural feedback. AUI systems need to learn so that they can perform most routine functions without constant user feedback, but still be able to inform the user of the functions performed on the user's behalf with very succinct aural messages. This is an area of active research.

A common user interface for services such as video capture, switching channels, and stored video playback is to emulate the camera, TV, and VCR on the screen. A general TV/VCR and camera metaphor interface will have all functions one would find in a typical video camera when it is in a video capture mode. It will have live buttons for selecting channels, increasing sound volume, and changing channels. The video is shown typically as a graduated bar, and the graduations are based on the full length of the video clip. The user can mark the start and end points to play the clip or to cut and paste the clip into a new video scene. The screen duplicates the cut and paste operation visually. Figure 1.5 shows an interface with the TV/VCR metaphor. A number of video editors emulate the functionality of typical film-editing equipment. The editing interface shows a visual representation of multiple film strips that can be viewed at user-selected frame rates. The user may view every *n*th frame for an entire scene or may wish to narrow down to every frame to pinpoint the exact point for splicing another video clip. The visual display software allows index markers on the screen for splicing another video clip down to the frame level. The screen display also shows the soundtracks and allows splicing soundtracks down to the frame level.

A good indexing system provides a very accurate counter for time as well as for individual frames; it can detect special events such as a change in scenes, start and end of newly spliced frames, start and end of a new soundtrack, etc. Also, the user can place permanent index markings to identify specific points of interest in the sequence. A number of indexing algorithms have been published in the literature.[13]

Indexing is useful only if the audio/video are stored. Since audio and video may be stored on separate servers, synchronization must be achieved before playback. Indexing may be needed to achieve the equivalent of a sound mixer and synthesizer found, for example, in digital pianos. Exact indexing to the frame level is desirable for video synchronization to work correctly. The indexing information must be stored on a permanent basis, for example, as SMPTE time codes or MPEG Systems time stamps. This information can be maintained in either the soundtrack or the video clip itself and must be extracted if needed for playback synchronization. Alternatively, the composite object created as a result of authoring can maintain the index information on behalf of all objects.

13.4 CONTENT CREATION FOR ITV[14–19]

A good authoring system must access predefined media and program elements, sequence these elements in time, specify their placement spatially, initiate actions in parallel, and indicate specific events (such as consumer inputs or external triggers) and the actions (some may be algorithmic functions) that may result from them. Some of these algorithmic functions may be performed by both the CPE and the server when the application is executed. Existing commercial tools for creating interactive multimedia applications run the gamut from low-level programming languages to packages that support the relatively casual creation of interactive presentations. Two very broad categories are (1) visual tools for use by nonprogrammers and (2) programming languages. The latter might include specialized "scripting" languages as well as more traditional general-purpose programming languages, perhaps extended by object-types or libraries that add ITV functionality. Usually the media elements used by these tools are created with a variety of other tools designed for particular media types, such as for still images, video and audio segments, and animation.

The design of an authoring system requires careful analysis of the following issues:

1. Hypermedia design details
2. User interfaces

3. Embedding/linking multimedia objects to the main presentation
4. Multimedia objects—storage and retrieval
5. Synchronization of components of a composite multimedia stream.

Hypermedia applications present a unique set of design issues not commonly encountered in other applications. A good user interface design is crucial to the success of a hypermedia application. The user interface presents a window to the user for controlling storage and retrieval, connecting objects to the presentation, and defining index marks for combining different multimedia streams. While the objects may be captured independently, they have to be played back together for authoring. The authoring system must allow coordination of many streams to produce a final presentation. A variety of authoring systems exist depending on their role and flexibility required. Some of these are described below.

13.4.1 Dedicated Authoring Systems

Dedicated authoring systems are designed for single users and single streams. The authoring is typically performed using desktop computer systems on multimedia objects captured by a video camera or on objects stored in a multimedia library. Dedicated multimedia authoring systems are not usually used for sophisticated applications. Therefore, they need to be engineered to provide user interfaces that are extremely intuitive and follow real-world metaphors (e.g., the VCR metaphor). Also flexibility and functionality has to be carefully limited to prevent overly complex authoring.

13.4.2 Timeline-Based Authoring

In a time-line-based authoring system, objects to be merged are placed along a timeline. The author specifies objects and their position in the timeline. For presentation in a proper time sequence each object is played at the prespecified object was captured in a timeline, it was fixed in location and could not be manipulated easily. Editing a particular component caused all objects in the timeline to be reassigned because the positions of objects were not fixed in time, only in sequence. Copying portions of the timeline became difficult as well. Newer systems define timing relations directly between objects. This makes inserting and deleting objects much easier because the start and end of each object is more clearly defined.

13.4.3 Structured Multimedia Authoring

Structured multimedia authoring is based on structured object-level construction of presentations. A presentation may be composed of a number of video clips with associated sound tracks, separate music tracks, and other separate soundtracks. Explicit representation of the structure allows modular authoring of the individual component objects. The timing constraints are also derived from the constituent data items. A good structured system also allows the user to define an object hierarchy and to specify the relative location of each object within that hierarchy. Some objects may have to undergo temporal adjustment, which allows them to be expanded or shrunk to better fit the available time slot. The navigation design of the authoring system should allow the author to view the overall structure, while at the same time being able to examine specific object segments more closely. The benefit of an object hierarchy is that removing the main object also removes all subobjects that are meaningful only as overlays on the main object. The design of the views must show all relevant information for a specific view. For example, views may be needed to show the object hierarchy plus individual component members of that hierarchy, and to depict the timing relation between the members of the hierarchy.

13.4.4 Programmable Authoring Systems

mable authoring systems provide such additional capability in the following areas:

- Functions based on audio and image processing and analysis
- Embedded program interpreters that use audio and image-processing functions.

Thus the main improvement in building user programmability into the authoring tool is not only to perform the analysis, but also to manipulate the stream based on the analysis results. This programmability allows the performing of a variety of tasks through the program interpreter rather than manually. A good example of how this is used is the case of locating video "silences" which typically occur before that start of a new segment (cut). The program can be used to help the user to get to the next segment by clicking a button rather than playing the video. Advances in authoring systems will continue as users acquire a better understanding of stored audio and video and the functionalities needed by users.

The development and deployment of interactive television on a large scale presents numerous technical challenges in a variety of areas. As progress continues to be made in all these areas, trial deployments of the interactive services are necessary to ascertain consumer interest in this new form of communication. We are certainly rather naive today about the future of these services. However, it seems certain that the variety of trials now being conducted will find a subset of this technology that is popular with consumers and a solid business proposition for service providers. This will provide the definition of services and the economic drivers necessary for these services to prosper.

REFERENCES

1. D. E. Blahut, T. E. Nichols, W. M. Schell, G. A. Story, and E. S. Szurkowski, "Interactive Television," *Proc. of IEEE,* pp. 1071–1086 (July 1995).
2. L. Crutcher and J. Grinham, "The Networked Jukebox," *IEEE Trans. Circuits Systems,* pp. 105–120 (April 1994).
3. G. H. Arlen, *Networked Multimedia: A Review of Interactive Services and the Outlook for Cable-Delivered Services,* Arlen Communications (April 1993).
4. E. G. Bartlett, *Cable Television Technology and Operations,* McGraw-Hill, New York (1990).
5. D. Anderson, Y. Osawa, and R. Govindam, "A File System for Continuous Media," *ACM Trans. Computer Systems* 10(4):311–337 (November 1992).
6. A. L. Narasimha Reddy and J. C. Wyllie, "I/O Issues in a Multimedia System," *Computer,* 27(3): 69–74 (March 1994).
7. P. Lougher and D. Shepherd, "The Design of a Storage Server for Continuous Media," *Computer J.* 36(1):32–42 (February 1993).
8. P. Venkat Rangan and H. M. Vin, "Efficient Storage Techniques for Digital Continuous Multimedia," *IEEE Trans. Knowledge Data Eng.* 5(4):564–573 (August 1993).
9. B. Ozden, R. Rostogi, and A. Silberschatz, "A Framework for the Storage and Retrieval of Continuous Media Data," *Proc. IEEE Conf. on Multimedia Computing and Systems* (May 95).
10. J. Kuegel, C. Keslein, and J. Rutledge, "Toolkits for Multimedia Interface Design," *Exhibition 92,* pp. 275–285.
11. R. Hayakawa, H. Sakagami, and J. Bekimoto, "Audio and Video Extensions to Graphical User Interface Toolkits," *3rd Int'l Workshop on Operating System Support for Audio/Video,* No. 92.
12. M. Blather, R. Deneburg (Eds.), *Multimedia Interface Design,* Addison-Wesley, Reading, MA (1991).
13. A. Hampapur, R. Jain, and T. E. Weywouth, "Indexing in Video Databases," *SPIE,* 2420: 293–305 (1995).
14. J. Conklin, "Hypertext: An Introduction and Survey," *IEEE Computer* (September 1987).
15. L. Hardman, G. van Rossum, and O. Bultermans, "Structured Multimedia Authoring," *ACM Multimedia,* pp. 283–289 (August 1993).
16. MACRO MEDIA Direction Overview Manual, Macromedia Inc. (1991).
17. Script X Language Guide, Kaleida (1993).
18. MACRO MEDIA Authorwave Working Model, Macromedia, Inc. (1993).
19. Apple Computer, *Inside MacIntosh: Quick Time,* Addison-Wesley, Reading, MA (1993).

14

HIGH-DEFINITION TELEVISION (HDTV)

Analog television was invented and standardized over fifty years ago, mainly for over-the-air broadcast of entertainment, news, and sports. In North America, the basic design was formalized in 1942, revised in 1946, and shortly after, limited service was offered. Color was added in 1954 with a backward compatible format. European broadcasting is equally old; monochrome television dates back to the 1950s and color was introduced in Germany in 1965 and in France in 1966. Since then, only backward compatible changes have been introduced, such as multichannel sound, closed-captioning, and ghost cancellation. The standard has thus evolved[4] in a backward compatible fashion, thereby protecting substantial investment made by the consumers, equipment providers, broadcasters, cable/ satellite service providers, and program producers.

Television is now finding applications in telecommunications (e.g., video conferencing), computing (e.g., desktop video), medicine (e.g., telemedicine), and education (e.g., distance learning). To enable many of these applications, television is being distributed by a variety of means not imagined at the time of standardization, such as cable (fiber or coaxial), video tapes or discs (analog or digital), and satellite (e.g., direct broadcast). It has also been recognized that the television signal packaged for terrestrial broadcasting, while cleverly done in its time, is wasteful of the broadcast spectrum and can benefit from modern signal processing. In addition, the average size of television sets has grown steadily, and the average viewer is getting closer to the display.

These newer applications, delivery media, and viewing habits are exposing various limitations of the current television system. At the same time a new capability is being enabled owing to the technology of compression, inexpensive and powerful integrated circuits, and large displays. Thus, we have an unstoppable combination of technological push with commercial pull.

High-Definition Television (HDTV) systems now being designed are more than adequate to fill these needs in a cost-effective manner. HDTV will be an entirely new high-resolution and wide-screen television system, producing much better quality pictures and sound. It will be one of the first systems that will not be backward compatible.

Worldwide efforts to achieve practical HDTV systems have been in progress for over 20 years. While we do not provide a detailed historical perspective on these efforts, a summary of the progress made towards practical HDTV is as follows:

- HDTV in Japan
  - —1970's NHK started systems work; infrastructure by other Japanese companies
  - —1980's led to MUSE, 1125/60 2:1 interlaced, fits 27 MHz satellite transponder
  - —Satellite HDTV service operating; NHK's format international exchange standard

—Narrow-MUSE proposed to FCC for USA
—Future of MUSE not promising, switch to all digital system expected

- HDTV in Europe
  —In 1986, broadcasters and manufacturers started HD-MAC effort
  —HD-MAC intended backward compatible with PAL and D-MAC; satellite (DBS)
  —HD-MAC obsolete; new system with digital compression/modulation sought
- HDTV in North America
  —In 1987, broadcasters requested FCC which resulted in formation of Advisory Committee (ACATS)
  —ACATS proposed hardware testing of 23 systems, 5 survived around test time
  —All 4 digital systems in running after tests, formed Grand Alliance in 1993

This chapter summarizes the main characteristics of HDTV and its expected evolution.

14.1 CHARACTERISTICS OF HDTV

HDTV systems are currently evolving from a number of proposals made by different organizations in different countries. While there is no complete agreement, a number of aspects are common to most of the proposals. HDTV is characterized by increased spatial and temporal resolution; larger image aspect ratio (i.e., wider image); multichannel CD-quality surround sound; reduced artifacts compared to the current composite analog television; bandwidth compression and channel encoding to make better utilization of the terrestrial spectrum; and finally, digital to provide better interoperability with the evolving telecommunications and computing infrastructure.

The primary driving force behind HDTV is a much better quality picture and sound to be received in the consumer home. This is done by increasing the spatial resolution by more than a factor of 2 in both the horizontal and vertical dimensions. Thus, we get a picture that has about 1000 scan lines and more than 1000 pels per scan line.

In addition, it is widely accepted (but not fully agreed to) that HDTV should eventually use only progressive (noninterlace) scanning at about 60 frames/s to allow better fast-action sports and far better interoperability with computers, while also eliminating the artifacts associated with interlace. The result of this is that the number of active pels in an HDTV signal increases by about a factor of 5, with a corresponding increase in the analog bandwidth.

From consumers' experience in cinema, a wider image aspect ratio (IAR) is also preferred. Most HDTV systems specify the image aspect ratio to be 16:9. The quality of audio is also improved by means of multichannel, CD-quality, surround sound in which each channel is independently transmitted.

Since HDTV will be digital, different components of the information can be simply multiplexed in time instead of frequency multiplexed on different carriers as in the case of analog TV. For example, each audio channel is independently compressed, and these compressed bits are multiplexed with compressed bits from video, as well as bits for closed-captioning, teletext, encryption, addressing, program identification, and other data services in a layer fashion.

Unfortunately, increasing the spatial and temporal resolution of the HDTV signal and adding multichannel sound also increases its analog bandwidth. Such an analog signal cannot be accommodated in a single channel of the currently allocated broadcast spectrum.

Moreover, even if bandwidth were available, such an analog signal would suffer interference both to and from the existing TV transmissions. In fact, much of the available broadcast spectrum can be characterized as a fragile transmission channel. Many of the 6-MHz TV channels are kept unused because of interference considerations, and are designated as *taboo* channels.

Therefore, all the current HDTV proposals employ digital compression, which reduces the bitrate from approximately 1 Gbit/s to about 20 Mbits/s. The 20 Mbits/s can be accommodated in a 6-MHz channel either in a terrestrial broadcast spectrum or a cable television channel. This digital signal is incompatible with the current television system and therefore can be decoded only by a special decoder.

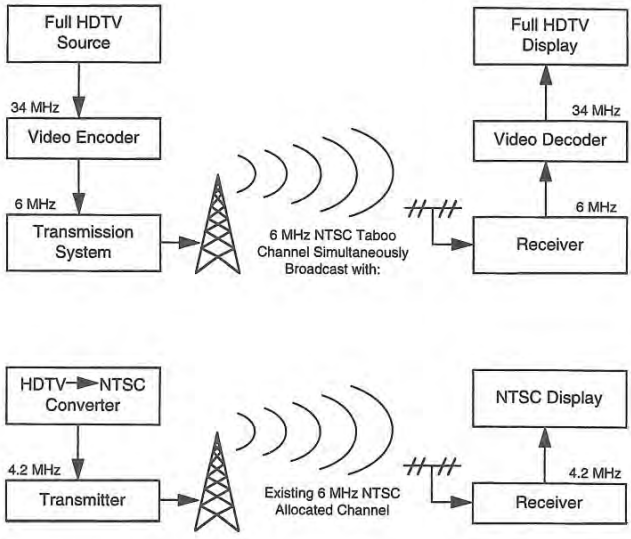


Fig. 14.1 Simulcasting of digital HDTV with NTSC analog TV. Under current channel assignments, many channels remain vacant (i.e., taboo) to avoid co-channel and adjacent channel interference. Since digital signals have lower transmission power, they can use the taboo channels in the new channel assignment.

In the case of terrestrial broadcast it is also necessary to reduce interference to and from other broadcast signals, including the analog television channels. Therefore, digital HDTV modulates the compressed audio and video bits onto a signal that has much lower power than the analog TV signal and has no high-power carriers, while requiring the same analog bandwidth as the current analog TV. In addition, most HDTV proposals simulcast* the HDTV signal along with the current analog TV having the same content (see Fig. 14.1).

A low-power, modulated digital HDTV signal is transmitted in a taboo channel that is currently unused because of co-channel and adjacent channel interference considerations. Simultaneously, in an existing analog channel, the same content (down-converted to the current television standard) will be transmitted and received by today's analog television sets. By proper shaping of the modulated HDTV signal spectrum, it is possible to avoid its interference into distant co-channel as well as nearby adjacent channel analog television signals. At the same time, distant co-channel and nearby adjacent channel analog television signals must not degrade the HDTV pictures significantly because of interference.

This design allows previously unused terrestrial spectrum to be used for HDTV. Of course, it could also be used for other services such as digital Standard Definition Television (SDTV), computer communication, and so forth. Consumers with current television sets will receive television as they do now. However, consumers willing to spend an additional amount will be able to receive HDTV in currently unused channels. As the HDTV service takes hold in the marketplace, spectrally inefficient analog television can be retired to release additional spectrum for additional HDTV, SDTV, or other services. In the United States, the FCC plans to retire NTSC after 15 years from the start of the HDTV broadcasts.

Another desirable feature of HDTV systems is easy interoperability with the evolving telecommunications and computing infrastructure. Such interoperability will allow easy introduction of different services and applications, and at the same time reduce costs because of commonality of components and platforms across different applica-

*The same program is sent on two channels.

tions. There are several characteristics that improve interoperability. Progressive scanning, square pels, and digital representation all improve interoperability with computers. In addition, they facilitate signal processing required to convert HDTV signals to other formats either for editing, storage, special effects, or down-conversions to current television. The use of a standard compression algorithm (e.g., MPEG[2,3]) transforms raw, high-definition audio and video into a coded bitstream, which is nothing more than a set of computer instructions and data that can be executed by the receiver to create sound and pictures.

Further improvement in interoperability at the transport layer is achieved by encapsulating audio and video bitstreams into fixed-size packets. This facilitates the use of forward error correction (necessary to overcome transmission errors) and provides synchronization and extensibility by the use of packet identifiers, headers, and descriptors. In addition, it allows easy conversion of HDTV packets into other constant size packets adopted by the telecommunications industry, for example, the Asynchronous Transfer Mode (ATM) standard for data transport.

Based on our discussion thus far the requirements for HDTV can be summarized as follows:

- Higher spatial resolution
  —Increase spatial resolution by at least a factor of 2 in horizontal and vertical direction
- Higher temporal resolution
  —Increase temporal resolution by use of progressive 60 Hz temporal rate
- Higher Aspect Ratio
  —Increase aspect ratio to 16:9 from 4:3 for a wider image
- Multichannel CD quality surround sound
  —At least 4 to 6 channel surround sound system
- Reduced Artifacts as compared to Analog TV
  —Remove composite format artifacts, interlace artifacts etc
- Bandwidth compression and channel coding to make better use of Terrestrial spectrum
  —Digital processing for efficient spectrum usage
- Interoperability with evolving telecommunication and computing infrastructure

14.2 PROBLEMS OF CURRENT ANALOG TV

Current analog TV was designed during the days when both the compression, modulation, and signal processing algorithms and the circuitry to process them were not adequately developed. Therefore, as the current TV began to get deployed for different applications, its artifacts as well as inefficiency of spectral utilization became bigger problems. Digital HDTV attempts to alleviate many of these problems. *Interline flicker, line crawl,* and *vertical aliasing* are largely a result of interlaced scanning in current television. These effects are more pronounced in pictures containing slowly moving high-resolution graphics and text with sharp edges. Most pictures used for entertainment, news, and sports did not historically contain such sharp detail, and these effects were tolerably small. The computer industry has largely abandoned interlace scanning to avoid these problems. Progressive scanning and transmission of HDTV can eliminate these effects almost entirely.

Large area flicker as seen from a close distance on large screens is the result of our ability to detect temporal brightness variations, even at frequencies beyond 60 Hz, in the peripheral areas of our vision. The cost of overcoming this problem in terms of bandwidth (i.e., by increasing the frame rate) is too large and is therefore not addressed in any current HDTV proposal. Computer displays employ much higher frame rates (up to 72 Hz) to overcome large area flicker. However, the brightness of a computer display is usually much higher than that of home TV, and this accentuates the effect, thereby requiring the larger frame rate.

Static raster visibility is more apparent on larger displays since viewers can visually resolve individual scan lines. The increased spatial resolution in HDTV will reduce this effect substantially.

Cross-color and cross-luminance are a result of the color modulation used to create the analog composite signal. HDTV will use component signals. Moreover, in digital HDTV, each color component will be compressed separately, with the compressed bits multiplexed in time. Thus, these two effects

other artifacts also arise, owing to transmission. For example, *ghosts,* which are due to multiple receptions of the same signal delayed with respect to each other, cannot be entirely removed from the analog signal. In digital HDTV, ghost cancellation can be done more successfully using adaptive equalizers, generally leaving no impact on the transmitted bitstream.

Another transmission problem is the interference from the other TV signals. As an example, the NTSC signal, containing high-energy sync pulses with sharp transitions, has poor spectrum utilization because of a high degree of *co-channel* and *adjacent channel* interference. Co-channel interference constrains the minimum geographic distance between transmitters on the same frequency band. Adjacent channel interference precludes using spectrally adjacent channels to serve the same area unless the transmitters are located together, thus ensuring near-equivalent, noninterfering signal strengths at all receiving locations. However, this is usually not possible. Both of these interferences are reduced by keeping adjacent channels in the spectrum vacant (i.e., taboo channels) at any one place, while using these vacant channels at other surrounding places. As a result, in the United States, only about 20 channels are usable in each location out of a total of 68. In the United Kingdom, only about 4 out of 44 channels are usable.

The problems of current analog TV can be summarized as follows:

- Interline Flicker, Line Crawl and Vertical Aliasing
  - —More predominant with high resolution moving graphics and text
  - —Can be eliminated by 60 Hz progressive format
- Large Area Flicker
  - —Seen on closeup of screen; due to brightness variations at freq < 60 Hz
  - —Can be resolved by going to 72 Hz, but bandwidth expensive
- Static Raster Visibility
  - —Visible on large displays, higher spatial resolution can reduce this
- Cross-color and Cross-luminance
  - —Caused by color modulation in composite; component signals can eliminate it
- Ghosts
  - —Multiple receptions of same signal, ghost cancellation easier for digital system
- Co-channel and Adjacent Channel Interference
  - —Co-channel limits distance between transmitters on same frequency band
  - —Adjacent channel precludes spectrally adjacent channels to serve same area
  - —Currently reduced by keeping spectrally adjacent channels vacant or taboo

14.3 GRAND ALLIANCE HDTV SYSTEM

The FCC considered various options for introducing HDTV service in the United States and finalized the following requirements:

- HDTV service should not interfere with current broadcast TV
  - —Digital HDTV and analog TV will be simulcast so as not to make current TV receivers immediately obselete
  - —HDTV and TV services are expected to coexist for many years so HDTV should not cause interference to analog TV reception
- HDTV service must fit in a 6 MHz channel
  - —Only one 6 MHz channel will be available per broadcaster, so HDTV while providing fairly good picture quality must fit in bandwidth of analog channel
  - —More bandwidth may be available later if analog TV is retired
- Terrestrial solution for HDTV to be usable on other Media
  - —Interoperability between Terrestrial HDTV and HDTV on Cable
- Other aspects of HDTV to be considered
  - —Channel switching in reasonable amount of time
  - —Interworking of contribution and distribution quality HDTV
  - —Recording of HDTV signal for professional and consumer use

Without going into a detailed historical perspective on U.S. HDTV efforts we present a snapshot of events leading to formation of a joint effort on part of various proponents for U.S. HDTV.

- FCC: Advisory Committee on Advanced TV (ACATS) in 1987
- To recommend toFCC a standard for terrestrial broadcast HDTV
- From 1987 to 1991 many proposals to ACATS, 5 survived
- Tests Sept 1991–Oct 1992; 4 digital, 1 analog proposal
- Results Feb 1993, major advantages of digital, drop analog
- May 1993 all 4 digital proponents joined in Grand Alliance (GA)
- GA participants: AT&T,[8] Sarnoff,[9] GI,[6,7] MIT,[7] Philips,[9] Thomson,[9] Zenith[8]

The Grand Alliance HDTV (GA-HDTV) system is a proposed[10–11,13–16] *transmission* standard considered by the FCC for North America. It is expected that a *production* standard that is optimized for operation within a studio, but at the same time interoperable with the transmission standard, will emerge once the transmission standard is approved.* Efforts to this end are underway within the FCC Advanced Television Systems Committee (ATSC). Similarly using the transmitted signal, it is expected that different manufacturers of consumer television receivers will create optimum displays consistent with picture quality obtainable for a given cost.

14.3.1 Overall GA-HDTV System

The GA-HDTV supports signals in multiple formats as follows:

1280 pels × 720 lines at frame rates of 24, 30, 60 Hz progressive

and

1920 pels × 1080 lines at frame rates of 24 Hz, 30 Hz progressive, and 30 Hz interlaced.

Thus, HDTV may start with available, inexpensive interlaced equipment, but migrate to progressive scanning at some later date. In addition, higher frame rate film sources are supported directly.

Frame rates may also be divided by 1.001 in systems that require compatibility with NTSC. With a 16:9 image aspect ratio these formats create square pels for both 720 and 1080 line rasters. Such flexibility in scanning allows different industries, program producers, application developers, and users to optimize among resolution, frame rate, compression, and interlace artifacts.

The GA-HDTV system incorporates a subset of the MPEG-2 video compression standard[3] described in Chapter 8. In addition, it includes several enhancements in the encoder built for initial FCC testing. For example, the GA-HDTV system can process film originated material at 24 or 30 frames/s by adaptively adjusting the compression algorithm at the encoder (see Fig. 14.2). Also, compression efficiency is improved by appropriate prefiltering/subsampling to handle signals of different formats and of varying quality (e.g., noisy signals). Among the compression techniques used are: DCT of motion-compensated signals with field/frame prediction, flexible refreshing (both I-frame as well as pseudo-random Intra-blocks), bidirectional prediction (B-frames), and forward analysis/perceptual selection for rate control and best picture quality.

The GA-HDTV system uses five-channel audio with surround sound, which compresses into a 384 kbits/s bit stream. Audio, video, and auxiliary data are packaged into fixed-length packets in conformity with the MPEG-2 Systems transport layer. MPEG-2 transport packets are 188 bytes long, consisting of up to 184 bytes of payload and 4 bytes of header. These packets easily interoperate with ATM since they have approximately a 4:1 ratio with ATM cells, which are 48 bytes long plus 5 bytes of header.

The modulation scheme chosen for over-the-air is 8-VSB (Vestigial Side Band), which gives 32.28 Mbits/s of raw data in a 6-MHz bandwidth. After error correction, 19.3 Mbits/s are available. Coaxial cable can carry more than this in a 6-MHz channel, since there is no co-channel and very little adjacent channel interference. Cable standards are under discussion. Thus, the GA-HDTV system attempts to

* The production standard needs to be interoperable with, but is not constrained by the transmission requirements. Production

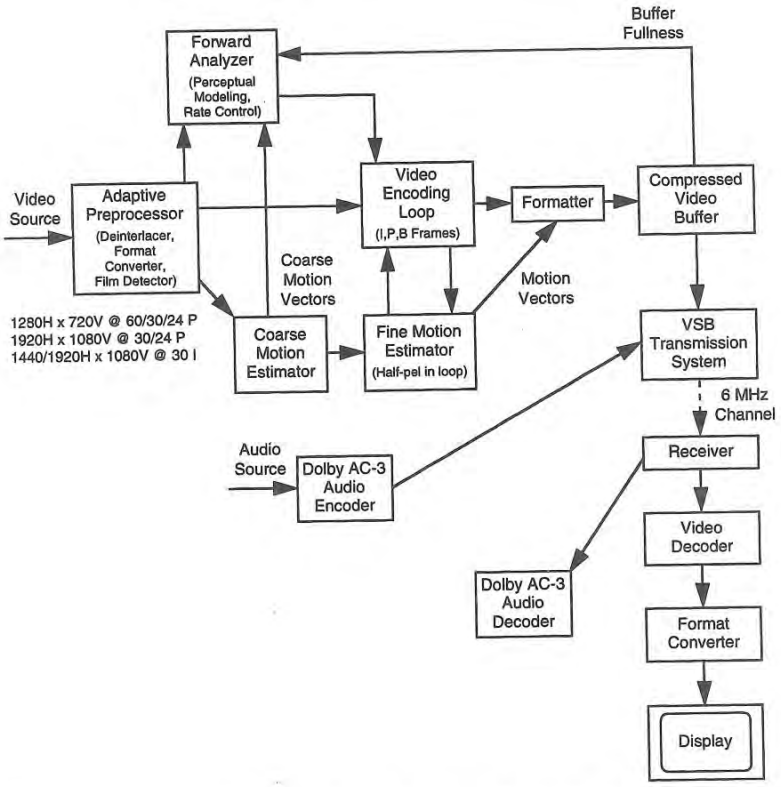


Fig. 14.2 Grand Alliance system overview. The block diagram shows major function blocks for the compression encoder–decoder for audio and video as well as the transmission system.

make the best compromise to suit different applications and provide for maximum interoperability.

We now summarize the aforementioned high level characteristics of the GA-HDTV system

- Video Compression
  - —SMPTE video format standards, 16:9 image aspect ratio, square pels
  - —MPEG-2 Video standard at Main Profile at High Level, bitstream packetized
- Audio Compression
  - —Audio sampling at 48 kHz
  - —AC-3 multichannel, bitstream packetized
- Transport
  - —Audio, Video PES packets along with ancillary data presented to mux
  - —Output of mux is fixed length 188 byte MPEG-2 Transport stream packets
- Modulation
  - —MPEG-2 Transport stream packets go to modulator for channel coding
  - —VSB transmission system, two modes 8-VSB and 16-VSB

The Grand Alliance would have wished to include a 1920 × 1080 progressive scan format at 60 Hz. However, the 6-MHz terrestrial broadcast channel does not allow for the required bitrate. As the NTSC audience declines and the interference into NTSC becomes less of a concern, there will be an opportunity to increase the data rate to support, in a compatible manner, 1920 × 1080 progressive scan at a 60-Hz frame rate.

The GA-HDTV system may allow for temporal scalability as well, so that an enhanced bitstream to handle 1920 × 1080 progressive scan at a 60-Hz frame rate can be created in the future. Figure 14.3 shows one possible scenario. A 30 frames/s subset

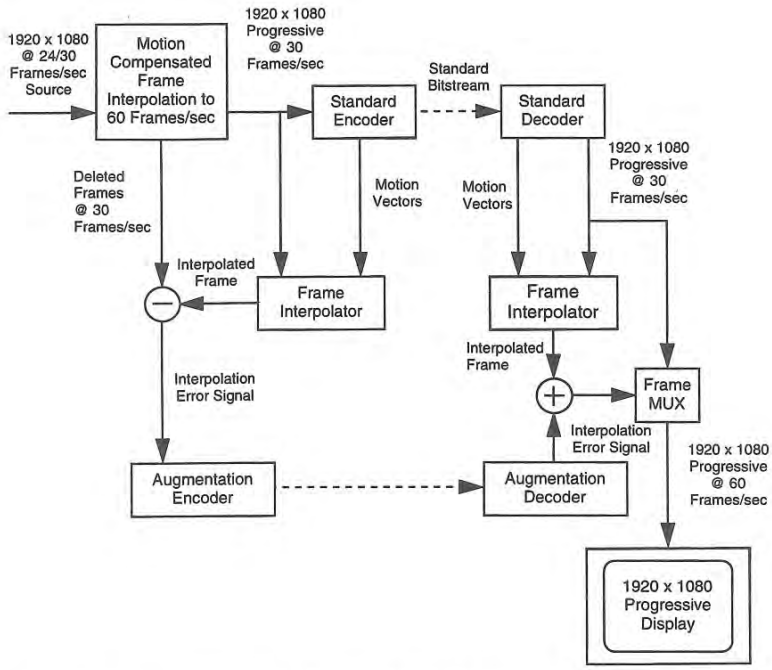


Fig. 14.3 An arrangement for using temporal scalability to enable a backward compatible enhancement from 30 Hz to 60 Hz progressive HDTV. Standard GA-HDTV cannot currently accommodate 60-Hz, 1080-line progressive HDTV. If transmission capacity is available in the future, this diagram shows a possible system for enhancing 24- or 30-Hz film for display at 60 Hz.

would be coded in the standard way while the remaining frames would be predicted by an interpolator. The difference between the predicted interpolation and the actual frame would be coded using the MPEG-2 algorithm[2] and sent to the decoder. Appropriately computed motion vectors could be used for both motion compensation and interpolation. This scheme would have the advantage of providing improved temporal performance for existing film material when properly applied.

14.3.2 GA-HDTV Video Issues and Parameters

We now list the main issues in GA-HDTV video and the choice of parameters in MPEG-2 video coding that can address these issues. The list of issues is summarized as follows:

- Raw rate of 1 Gbit/s or more to be compressed to about 20 Mbit/s, ie, compression by a factor
- Picture formats with 16:9 image aspect ratio and square pels
  —1280×720 at 60, 30 and 24 Hz progressive
  —1920×1080 at frame rates of 30 Hz interlaced and 30 and 24 Hz progressive
- Compression scheme with following properties for High Efficiency:
  —Source Adaptive Preprocessing
  —Temporal Redundancy Reduction
  —Spatial Redundancy Reduction
  —Perceptual Irrelevancy Exploitation
  —Efficient Entropy Coding
  —Frequent Refresh

Next in Table 14.1 we summarize[11,13,16] specific MPEG-2 Video coding parameters suitable for GA-HDTV.

14.3.3 GA-HDTV Audio Issues and Parameters

We now list the main issues in GA-HDTV

Table 14.1 GA-HDTV Video Parameters Summary

Video Parameter	Value for Format 1	Value for Format 2
Active pels	1280 (hor) × 720 (vert)	1920 (hor) × 1080 (vert)
Total Samples	1600 (hor) × 787.5 (vert)	2200 (hor)× 1125 (vert)
Frame Rate	60 Hz progressive/ 30 Hz progressive/ 24 Hz progressive	30 Hz interlaced/ 30 Hz progressive/ 24 Hz progressive
Chrominance Sampling	4:2:0	
Image Aspect Ratio	16:9	
Data Rate	Selected Fixed Rate (10 – 45 Mbit/s)/Variable	
Colorimetry	SMPTE 240M	
Picture Coding Types	Intra (I-), Predictive (P-), Bidirectionally Predictive (B-) pictures	
Video Refresh	I-picture/progressive refresh	
Picture Structure	Frame	Frame/Field (interlace only)
Coefficient Scan	ZigZag	Zigzag/Alternate
DCT coding Modes	Frame	Frame/Field (interlace only)
Motion Comp Modes	Frame	Field/Dualprime (interlace only)
Motion Vector Range	Horizontal: Unlimited by Syntax Vertical: −128, + 127.5	
Motion Vector Precision	1/2 pixel precision	
DC Coefficient Precision	8 bits/9 bits/10 bits	
Rate Control	Modified TM5 with Forward Analyzer	
Film Mode Processing	Automated 3:2 pull down detection and coding	
Max VBV Buffer size	8 Mbits	
Inter/Intra Quantization	Download scene dependent matrices	
VLC Coding	Coefficient VLC Intra and Inter 2D VLC tables	
Error Concealment	Motion compensated frame holding (slice level)	

AC-3 audio coding address these issues. The list of issues is summarized as follows.

- Main audio service can range from monophonic, through stereo upto 6 channel surround service (left, center, right, left surround, right surround, and subwoofer). Sixth channel conveys only low frequency (subwoofer) info and is called 0.1 channel; total of 5.1
- Additional service can be provided, for example:
  —Service for hearing or visually impaired
  —Multiple simultaneous languages
- Audio sampling rate is 48 kHz, with 6 channels and 18 bits per sample. Raw data rate is 5 Mbit/s and needs to be compressed to 384 kbit/s, a compression factor of 13
- From a multichannel service, if mono or stereo outputs are needed, downmix is done at the decoder
  —Downmix may be done in frequency domain to reduce decoder complexity
  —Program originator indicates in bitstream which downmix coefficients for program

Next, in Table 14.2 we summarize specific Dolby AC-3 audio coding parameters suitable for GA-HDTV

14.3.4 GA-HDTV Transport Issues and Parameters

We now list the main issues in GA-HDTV Transport and the choice of parameters in MPEG-2 Transport system. The list of issues is summarized as follows.

- A Transport packet is composed of 4 bytes of header followed by 184 bytes of payload
  —Header identifies content of packet and the nature of the data. It also provides info about packet synchronization, error handling and conditional access
  —For conditional access, header indicates if the payload data is scrambled
  —Sometimes additional header information in adaptation header, a variable length field in

Table 14.2 GA-HDTV Audio Parameters Summary

Audio Parameter	Value
Number of Channels	5.1
Audio Bandwidth	10–20 kHz
Sampling Frequency	48 kHz
Dynamic Range	100 dB
Compressed Data Rate	384 kbit/s

Table 14.3 GA-HDTV Transport Parameters Summary

Transport Parameter	Value
Multiplex Technique	MPEG-2 Systems Layer
Packet Size	188 bytes
Packet Header	4 bytes including sync
Services	
Conditional Access	Payload scrambled on service basis
Error Handling	4-bit continuity counter
Prioritization	1 bit/packet
System Multiplex	Multiple program capability described in PSI Stream

payload of Transport Packet. This is used for audio-video synchronization, random access into compressed bitstream, and local program insertion

- Transport stream provides interoperability with ATM
  —ATM cell contains 5 byte header and 48 byte payload; a transport packet (188 bytes) can fit into four ATM cells ($4 \times 48 = 192$)

Next, in Table 14.3 we summarize[11,13] specific MPEG-2 Transport parameters suitable for HDTV.

14.3.5 GA-HDTV Transmission Issues and Parameters

We now list the main issues[11,13] in GA-HDTV Transmission and the choice of parameters in the modulation scheme chosen.

- Vestigial Sideband (VSB) transmission system with 2 modes
  —8-VSB for terrestrial broadcasting
  —16-VSB for high data rate cable transmission
- Both Modes use Reed-Solomon coding, segment sync, a pilot, and a training signal. Terrestrial mode adds trellis coding
- Symbol rate and Payload Data Rate
  —Terrestrial mode uses 10.76 Msymbols/s at 3 bits/symbol; payload 19.3 Mbit/s
  —Cable mode uses 10.76 Msymbols/s but 4 bits/symbol; payload 38.6 Mbit/s
- Data transmitted according to a data frame
  —It contains, a first data field sync segment, 312 data segments, a second data field sync segment, 312 data segments. Each segment is 4 symbols for sync followed by 832 symbols of data
- Terrestrial Mode Segments
  —One segment corresponds to 1 MPEG-2 Transport packet. Number of bits plus FEC per symbol is 832 symbols $\times$ bits/symbol = 2496. MPEG-2 transport packet is 188 bytes, RS encoding adds 20 bytes, further trellis coding adds 1 bit for every 2 input bits increasing ratio by 3/2, for total of 312 bytes
- Symbols modulate a carrier using suppressed-carrier modulation
  —Before transmission most of lower sideband is removed, resulting in flat spectrum except at band edges. At receiver, a small pilot is used to achieve carrier lock is added 310 kHz above lower band edge

Next, in Table 14.4 we summarize[11,13] specific Transmission parameters suitable for GA-HDTV.

14.4 PERFORMANCE OF MPEG-2 VIDEO ON HDTV PICTURES

In the previous section we have covered the GA-HDTV system in detail. Since MPEG-2 video is a main component of this system we now evaluate the performance of various encoding options and tools on HDTV pictures. Simulations on several sequences are performed. Although our simulations[5,12] primarily report results on Peak-Signal to RMS-Noise Ratio (PSNR), we are aware that a subjective measure of performance is usually needed to truly evaluate the video coding quality. Also, the simulation results presented here are for illustration purposes and do not represent the results of an optimized MPEG-2 video encoder. Generally, they use the various macroblock mode decisions and rate control of the MPEG-2 Test Mode.[1]

In our simulations we employ three interlaced

Table 14.4 GA-HDTV Transmission Parameters Summary

Transmission Parameter	Terrestrial Mode	High Data Rate Cable Mode
Channel Bandwidth	6 MHz	6 MHz
Excess Bandwidth	11.5%	11.5%
Symbol Rate	10.76 Msymbols/s	10.76 Msymbols/s
Bits per Symbol	3	4
Trellis FEC	2/3 rate	None
Reed-Solomon FEC	(208, 188) T = 10	(208, 188) T = 10
Segment Length	836 Symbols	836 Symbols
Segment Sync	4 symbols/segment	4 symbols/segment
Frame Sync	1 per 313 segments	1 per 313 segments
Payload Data Rate	19.3 Mbit/s	38.6 Mbit/s
NTSC cochannel Rejection	NTSC Rejection Filter inReceiver	N/A
Pilot power contribution	0.3 dB	0.3 dB
C/N threshold	14.9 dB	26.3 dB

HDTV sequences of 1920 × 1024 active resolution, derived from a 1920 × 1035 resolution SMPTE 240M source. For each of these sequences, namely Marching in, Leaves, and Basketball, 60 frames were employed.

14.4.1 Group-of-Pictures Length $N = 9$ versus $N = 15$ Comparison for HDTV

We now show PSNR results[12] of experiments of frame-picture coding for P-picture prediction distances $M = 1$ and $M = 3$ and group-of-pictures of size $N = 9$ and 15 on HDTV sequences. We perform simulations with and without dual-prime motion compensation, but do not use concealment motion vectors. Adaptive motion compensation and adaptive DCT coding modes are allowed. Table 14.5 indicates that the use of $N = 9$ instead of $N = 15$ degrades the PSNR performance only by a small amount, up to 0.2 dB. Thus, $N = 9$ can be used for faster switching between TV channels without noticeable degradation in performance.

Table 14.6 shows that Dual-prime achieves a higher PSNR without the extra delay of B-pictures. However, as shown in Table 14.3 B-pictures achieve a higher PSNR than Dual-prime.

14.4.2 No B-Pictures versus with B-Picture Comparison for HDTV

With the adaptive motion compensation and adaptive DCT coding conditions for frame-pictures we conduct simulations on interlaced HDTV video to estimate performance tradeoffs with ($M = 3$ case) and without B-pictures ($M = 1$ case). The detailed PSNR statistics are provided for each picture types only for reference. Table 14.4 illustrates that $M = 3$ case outperforms $M = 1$ without dual prime by as much as 0.25 to 0.6 dB with the possibility of 0.3 to 0.6 additional dB improvement

Table 14.5 Luminance PSNR [dB] for interlaced HDTV: $M = 1$ versus 3 and $N = 9$ versus 15 comparison, bitrate = 18 Mbits/s

	$M = 1$ without Dual-prime MC		$M = 1$ with Dual-prime MC		$M = 3$	
Sequence	$N = 9$	$N = 15$	$N = 9$	$N = 15$	$N = 9$	$N = 15$
Marching in	25.40	25.51 (+0.11)	25.96	26.12 (+0.16)	25.70	25.75 (+0.05)

Table 14.6 Coding statistics for interlaced HDTV $M = 1$ with and without Dual prime, $N = 15$, bitrate = 18 Mbits/s. (a) "Marching in," (b) "Leaves," (c) "Basketball"

"Marching In"	$M = 1$ Without Dual-Prime MC			$M = 1$ with Dual-Prime MC		
Statistic	I-Picture	P-Picture	Avg. Picture	I-Picture	P-Picture	Avg. Picture
Bits	1,866,871	509,462	599,956	1,973,228	501,979	600,062
Avg. Qnt step	31.32	31.14	31.16	29.50	28.40	28.48
SNR, Y	26.89	25.41	25.51	27.10	26.05	26.12

"Leaves"	$M = 1$ Without Dual-Prime MC			$M = 1$ with Dual-Prime MC		
Statistic	I-Picture	P-Picture	Avg. Picture	I-Picture	P-Picture	Avg. Picture
Bits	1,888,359	508,093	600,111	1,964,643	502,588	600,059
Avg. Qnt step	43.36	43.36	43.36	41.00	39.50	39.60
SNR, Y	26.14	24.03	24.17	26.40	25.05	25.14

"Basketball"	$M = 1$ Without Dual-Prime MC			$M = 1$ with Dual-Prime MC		
Statistic	I-Picture	P-Picture	Avg. Picture	I-Picture	P-Picture	Avg. Picture
Bits	1,662,531	523,859	599,771	1,739,009	518,647	600,005
Avg. Qnt step	28.72	26.04	26.22	27.48	22.94	23.24
SNR, Y	29.97	28.69	28.78	30.14	29.56	29.60

Table 14.7 Coding statistics for interlaced HDTV sequence $M = 3$, $N = 15$, bitrate = 18 Mbits/s. (a) "Marching in," (b) "Leaves," (c) "Basketball"

Statistic	I-Picture	P-Picture	B-Picture	Avg. Picture
Avg. Bits	2,060,618	762,721	377,477	599,830
Avg. Qnt. step	28.46	29.16	40.92	36.82
SNR, Y	27.27	25.82	25.56	25.75

Statistic	I-Picture	P-Picture	B-Picture	Avg. Picture
Avg. Bits	2,140,284	930,871	297,573	599,360
Avg. Qnt. step	37.30	38.68	52.16	47.42
SNR, Y	26.89	24.72	24.54	24.75

Statistic	I-Picture	P-Picture	B-Picture	Avg. Picture
Avg. Bits	1,941,327	713,456	405,251	596,209
Avg. Qnt. Step	22.44	22.62	33.02	29.42
SNR, Y	30.78	29.35	28.76	29.07

Table 14.8 Luminance PSNR [dB] for interlaced HDTV: with B- versus without B-picture comparison, $N = 15$, bitrate = 18 Mbits/s

	P-Pictures Only			Average of All Pictures		
Sequence	$M = 1$, No Dual-Prime	$M = 1$ and Dual-Prime	$M = 3$	$M = 1$, No Dual-Prime	$M = 1$ and Dual-Prime	$M = 3$
Marching in	25.51	26.05 (+0.54)	25.82 (+0.31)	25.51	26.12 (+0.61)	25.75 (+0.24)
Leaves	24.03	25.05 (+1.02)	24.72 (+0.69)	24.17	25.14 (+0.97)	24.75 (+0.58)
Basketball	28.69	29.56 (+0.87)	29.35 (+0.66)	28.78	29.60 (+0.82)	29.07 (+0.29)

when dual-prime is included in adaptive motion compensation.

14.4.3 MPEG-1 Quantization versus MPEG-2 Nonlinear Quantization Scale Evaluation for HDTV

Using an interlaced HDTV video source we investigate[12] the effect of choosing linear or nonlinear quantization scale tables. In all the previous experiments, the linear quantization table was chosen. As expected, Tables 14.9 and 14.10 show that based on PSNR statistics it is not possible to draw any conclusion regarding which table is the best. The improvements are strictly subjective. The nonlinear table has an additional benefit for rate control in difficult sequences.

Table 14.9 Luminance PSNR [dB] for interlaced HDTV sequences for quantizer comparison: $M = 3$, $N = 15$, bitrate = 18 Mbits/s

Sequence	Linear Quant Scale	Nonlinear Quant Scale
Marching in	25.75	25.72
Leaves	24.75	24.71
Basketball	29.07	29.03

Table 14.10 Luminance PSNR [dB] for interlaced HDTV sequences for quantizer comparison: $M = 3$, $N = 15$, 45 Mbits/s coding

Sequence	Linear Quant Scale	Nonlinear Quant Scale
Marching in	28.23	28.24
Leaves	28.95	28.96
Basketball	32.29	32.33

14.4.4 Evaluation of HDTV Coding Performance at Various Bitrates

Tables 14.11 to 14.13 show the PSNR results[12] of coding interlaced HDTV source material at a variety of bitrates and under various motion com-

Table 14.11 Luminance PSNR [dB] for interlaced HDTV sequences: $M = 1$ without Dual-prime, $N = 15$, bitrates = 18, 30, and 45 Mbits/s

Sequence	18 Mbits/s	30 Mbits/s	45 Mbits/s
Marching in	25.51	26.95 (+1.44)	28.09 (+2.58)
Leaves	24.17	26.58 (+2.41)	28.70 (+4.53)
Basketball	28.78	30.58 (+1.80)	32.08 (+3.30)

Table 14.12 Luminance PSNR [dB] for interlaced HDTV sequences: $M = 1$ with Dual-prime, $N = 15$, bitrates = 18, 30, and 45 Mbits/s

Sequence	18 Mbits/s	30 Mbits/s	45 Mbits/s
Marching in	26.12	27.47 (+1.35)	28.62 (+2.50)
Leaves	25.14	27.61 (+2.47)	29.42 (+4.28)
Basketball	29.60	31.26 (+1.66)	32.66 (+3.06)

Table 14.13 Luminance PSNR [dB] for interlaced HDTV sequences: $M = 3$, $N = 15$, bitrates = 18, 30, and 45 Mbits/s

Sequence	18 Mbits/s	30 Mbits/s	45 Mbits/s
Marching in	25.75	27.10 (+1.35)	28.23 (+2.48)
Leaves	24.75	26.99 (+2.24)	28.95 (+4.20)
Basketball	29.07	30.84 (+1.77)	32.29 (+3.22)

pensation conditions. The coder uses adaptive frame/field DCT, zigzag scan, concealment motion vectors, and the MPEG-1 VLC table.

14.5 EVOLUTION OF HDTV

HDTV faces significant challenges to its widespread deployment. One line of thinking indicates that the current resolution of television is sufficient for most applications and that the television should evolve first by being digital and then by being more interactive. This would help consumers get the content they want with modern computer processing in a form (resolution, duration, etc.) that they choose. For example, better use of bandwidth could be made by filling it with a larger number of lower resolution programs and then interactively choosing between them. This is quite feasible since with 20 Mbits/s, which can be accommodated in the 6-MHz bandwidth, one can fit 4 to 10 compressed digital 525-line NTSC programs. Thus, a 60-channel cable television system would be converted into a 240- to 600-channel digital NTSC system. Viewers could then select between them with ease, with the help of a good user interface. However, good interoperability with progressive scan computer displays is foregone.

Many people also believe that the precious radio spectrum should not be used to transmit television to fixed locations. They argue that the terrestrial spectrum should be reserved exclusively for mobile services, which are in great demand as a result of cost reductions of wireless terminals owing to low power, compact electronics, and better batteries. Larger and larger fractions of modern industrialized countries are installing fiberoptic cables having enormous traffic capability. Television could reach many people through such fixed wired connections. However, the cost of "wiring" any country is substantial, and therefore it may take quite a while before television could be distributed entirely by land-based cables.

The high cost of HDTV sets may be another inhibiting factor, although with growing demand, both the electronics and displays should become cheaper over time. Previous experience from color television and VCRs in terms of market penetration, combined with cost projections from the electronics and display industries, give much hope for a low-cost HDTV by the year 2000. While the costs look more manageable than ever before, broadcasters are still unenthusiastic about the business value of HDTV. However, they are also concerned that the other media (tape, DBS, cable) might deliver HDTV to the public before them or that the FCC might assign the taboo channel spectrum to other services if the broadcasters do not use it.

HDTV may also grow by other means. In the United States, the National Information Infrastructure (NII) is being created. Digital HDTV, with its flexible data transport structure, is an immediate opportunity to be deployed as an important part of the interconnected web of information superhighways that make up the NII. GA-HDTV's interoperability with the computing and telecommunication platforms may allow many uses of HDTV for medicine, information access, and education. Thus, it is clear that many of the technical impediments to HDTV are being removed. Customers and the marketplace will decide which way it will evolve.

REFERENCES

1. Test Model Editing Committee, "MPEG-2 Video Test Model 5," ISO/IEC JTC1/SC29/WG11 Doc. N0400 (April 1993).
2. "Generic Coding of Moving Pictures and Associated Audio Information: Systems ISO/IEC 13818–1: Draft International Standard (November 1994).
3. "Generic Coding of Moving Pictures and Associated Audio Information: Video, ISO/IEC 13818–2: Draft International Standard (November 1994).
4. A.N. Netravali and B.G. Haskell, *Digital Pictures: Representation, Compression, and Standards,* Second Edition, Plenum Press, 1995.
5. A. Puri and B.G. Haskell, "Digital HDTV Coding with Motion Compensated Interpolation," *Proc. of the Fourth Int. Workshop on HDTV and Beyond,* (September 1991); *Signal Processing of HDTV,* III, Elsevier Science, 1992.
6. GI Corp., "Digicipher™ HDTV System Description," (August 1991).

7. MIT, "Channel Compatible Digicipher HDTV System," Amer. Television Alliance (April 1992).
8. Zenith Electronics Corp. "Digital Spectrum Compatible HDTV System," (November 1991).
9. Advanced Television Research Consortium, "Advanced Digital Television System Description," Jan 1992.
10. R. Hopkins, "Progress on HDTV Broadcasting Standards in the United States, Signal Processing: Image Commun., vol. 5, pp. 355–378, 1993.
11. R. Hopkins, "Digital Terrestrial HDTV for North America: The Grand Alliance HDTV System," *IEEE Transac. on Consumer Electronics,* vol. 40, No. 3, August 1994.
12. R.L. Schmidt, A. Puri, and B.G. Haskell, "Performance Evaluation of Nonscalable MPEG-2 Video Coding," *Proc. SPIE Visual Commun and Image Proc.,* (October 1994).
13. United States Advanced Television Systems Committee, "Digital Television Standard for HDTV Transmission," ATSC Standard Doc A/53 (April 1995).
14. The Grand Alliance, "The U.S. HDTV Standard," *IEEE Spectrum magazine,* pp. 36–45 (April 1995).
15. Y. Ninomiya, "High Definition Television Systems," *Proceedings of the IEEE,* vol. 83, No. 7, pp. 1086–1093 (July 1995).
16. E. Petajan, "The HDTV Grand Alliance System," *Proceedings of the IEEE,* vol. 83, No. 7, pp. 1094–1105 (July 1995).

17

MPEG-4 and the Future

The MPEG-1 and the MPEG-2 standards are two remarkable milestones, the overwhelming impact of which, in defining the next generation of digital multimedia, products, equipment, and services is already being felt. MPEG-1 decoders/players are becoming commonplace for multimedia on computers. MPEG-1 decoder plug-in hardware boards have been around for a few years, and now, software MPEG decoders are already available with release of new operating systems or multimedia extensions for PC and Mac™ platforms. MPEG-2 is well on its way to making a significant impact in a range of applications such as digital VCRs, digital satellite TV, digital cable TV, HDTV, and others. So, you may wonder, what is next?

As mentioned earlier, MPEG-1 was optimized for typical applications using noninterlaced video of 30 (25, in European format) frames/s at bitrates in the range of 1.2 to 1.5 Mbits/s, although it can certainly be used at higher bitrates and resolutions. MPEG-2 is more generic in the sense of picture resolutions and bitrates although it is optimized mainly for interlaced video of TV quality in a bitrate range of 4 to 9 Mbits/s; it also includes MPEG-1-like noninterlaced video coding, and also supports scalable coding. Getting back to the question of what's next, a possible reply is in the form of yet another question: What about coding of video at lower (than 1 Mbit/s) bitrates?

Next, yet another question arises: Is it possible for a standard to be even more generic than MPEG-2 and yet be efficient, flexible, and extendable in the future? If so, can it also be compatible in some way with the previous standards such as MPEG-1, MPEG-2, or other related standards. Further, can it also efficiently code or represent multiviewpoint scenes, graphics, and synthetic models, and allow functions such as interactivity and manipulation of scene contents? Indeed, we seem to have very high expectations from a new MPEG standard! To what extent this wish list may eventually be fullfilled is difficult to predict; however, it appears that ongoing work in MPEG-4 (the next MPEG standard) certainly seems to be addressing these issues. We will discuss this more a little later; for now we explore the relationship of MPEG-4 with other low-bitrate video standards.

During the early 1980s it was believed that for videoconferencing applications, reasonable quality was possible only at 384 kbits/s or higher and good quality was possible only at significantly higher bitrates, around 1 Mbit/s. Around 1984, the ITU-T (formerly CCITT) started an effort to standardize video coding primarily for videophone and video conferencing applications. This effort, although originally directed at video coding at 384 kbits/s and multiples of 384 kbits/s to about 2 Mbits/s, was refocused after a few years to address bitrates of 64 kbits/s and multiples of 64 kbits/s ($p \times 64$ kbits, where p can take values from 1 to

30). That standard was completed in late 1988 and is officially called the H.261 standard (the coding method is often referred to as $p \times 64$.). The H.261 standard, like the MPEG-1 and the MPEG-2 standards, is a decoder standard. In fact, the basic framework of the H.261 standard was used as a starting point in design of the MPEG-1 standard.

A few years ago, owing to a new generation of modems allowing bitrates of 28 kbits/s or so over PSTN, a new possibility of acceptable quality videophones at these bitrates arose. In early 1994, ITU-T launched a short-term effort to optimize and refine H.261 for videophone applications over PSTN; this effort is nearly complete and has resulted in a standard, called H.263.[5] Although, in this standard, video coding is optimized for the 10 to 24 kbits/s range, as is usual, the same coding methods have also been found to be quite efficient over a wider range of bitrates. Within ITU-T, effort is also in progress to come up with a version of this standard with more error resilience that would be robust for mobile applications as well. The ITU-T also has a plan for a long-term standard by 1998, capable of providing even higher coding efficiency; joint work is in progress with MPEG and it is expected that a subset of an ongoing MPEG-4 standard will satisfy this goal.

Besides videophone applications, at very low to low bitrates, several new applications have recently arisen. The requirements of such applications appear to be fairly diverse and not satisfactorily met by the H.263, the MPEG-1, the MPEG-2, or any other standards. To address such potential applications, in 1993, ISO started work toward the MPEG-4 standard, which is expected to reach a mature stage of Committee Draft (CD) by November 1997 and the final stage of International Standard (IS) by November 1998. Much of this chapter is an evaluation of the current status of MPEG-4 and what is foreseen beyond MPEG-4. However, before discussing MPEG-4, it is useful to have a good idea of the coding methods and the syntax of H.263, since H.263 has been found to offer a good starting basis for MPEG-4 (history repeats itself!). MPEG-4, as we shall see, is not only about increased coding efficiency but also about content-based functionalities, higher error resilience, and much much more.

17.1 H.263: THE ITU-T LOW-BITRATE STANDARD

In general, H.263, since it is derived from H.261 is based on the framework of block motion-compensated DCT coding. Both the H.261 and the H.263 standards, like the MPEG-1 and the MPEG-2 standards, specify bitstream syntax and decoding semantics. However, unlike the MPEG-1 and the MPEG-2 standards, these standards are video coding standards only and thus do not specify audio coding or systems multiplex, which can be chosen from a related family of ITU-T standards to develop applications requiring full systems for audio–visual coding. Also, unlike the MPEG standards, the ITU-T standards are primarily intended for conversational applications (low bitrates and low delay) and thus usually do not include coding tools needed for fully supporting applications requiring interactivity with stored data.

The H.263 standard[5] specifies decoding with the assumption of block motion-compensated DCT structure for encoding. This is similar to H.261, which also assumes a block motion-compensated DCT structure for encoding. There are, however, some significant differences in the H.263 decoding process as compared to the H.261 decoding process, which allow encoding to be performed with higher coding efficiency. For developing the H.263 standard, an encoding specification called the Test Model (TMN) was used for optimizations. TMNs progressed through various iterative refinements; the final test model was referred to as TMN5. Earlier, during the 1980s, when H.261 was developed, a similar encoder specification referred to as the Reference Model (RM) was used and went through iterative refinement leading to the final reference model, called RM8. The H.263 standard, although it is based on the H.261 standard, supports a structure that is significantly optimized for coding at lower bitrates such as several tens of kbit/s while maintaining good subjective picture quality at higher bitrates as well. We now discuss the syntax and semantics of the H.263 standard as well as how TMN5 encoding works.

17.1.1 Overview

The H.263 standard,[5] although intended for low bitrate video coding, does not specify a constraint on video bitrate; such constraints are given by the terminal or the network. The video coder generates a self-contained bitstream. The decoder performs the reverse operation. The codec can be used for either bidirectional or unidirectional visual communication.

The video coding algorithm is based on H.261 with refinements/modifications to enhance coding efficiency. Four negotiable options are supported to allow improved performance. The video coding algorithm, as in H.261, is designed to exploit both the spatial and temporal redundancies. The temporal redundancies are exploited by interpicture prediction, whereas the spatial redundancies are exploited by DCT coding. Again as in H.261, although the decoder supports motion compensation, the encoder may or may not use it. One difference with respect to H.261 is that instead of full-pixel motion compensation and loop filter, H.263 supports half-pixel motion compensation (as per MPEG-1/2), providing improved prediction. Another difference is in the Group-of-Block (GOB) structure, the header for which is now optional. Further, to allow improved performance, H.263 also supports four negotiable options which can be used together or separately, and are listed as follows:

- Unrestricted Motion Vector mode—This mode allows motion vectors to point outside a picture, with edge pixels used for prediction of nonexisting pixels.
- Syntax-based Arithmetic Coding mode—This mode allows use of arithmetic coding instead of variable length (huffman) coding.
- Advanced Prediction mode—This mode allows use of overlapped block motion compensation (OBMC) with four 8×8 motion vectors instead of 16×16 vectors per macroblock.
- PB-frames mode—In this mode, two pictures, one, a P-picture and the other a B-picture, are coded together as a single PB-picture unit.

17.1.1.1 Source Format

The video coder operates on noninterlaced pictures occurring 29.97 times per second (that is, 30000/1001, the same picture rate as NTSC). Pictures are represented by luminance, *Y*, and the two color difference signals (chrominance signals, *Cb* and *Cr*). Five picture formats are standardized and are referred to as sub-QCIF, QCIF, CIF, 4CIF, and 16CIF; the corresponding resolution of luminance signal for these formats is 128 × 96, 176 × 144, 352 × 288, 704 × 576, or 1408 × 1152. All H.263 decoders are mandated to operate using sub-QCIF and QCIF. Some decoders may also operate with CIF, 4CIF, or 16CIF. The information about which formats can be handled by decoders are signalled to encoder by external means. The two chrominance signals (Cb and Cr) have half the resolution of luminance in each direction. The chrominance samples are located at the center, equidistant from the four neighboring luminance samples.

17.1.1.2 Data Structures

Each picture is divided into GOBs. In H.263, a GOB consists of a row of $k \times 16$ lines with $k = 1$ for sub-QCIF, QCIF, and CIF, $k = 2$ for 4CIF and $k = 4$ for 16CIF; thus there are 6 GOBs for sub-QCIF, 9 for QCIF, and 18 for CIF, 4CIF, and 16CIF. Data for each GOB consists of a GOB header (which may be empty) followed by data for each of the macroblocks contained in a GOB. A macroblock in H.263 is the same as that in MPEG-1/2, that is, it consists of a 16 × 16 block of *Y*, and corresponding 8 × 8 blocks of each of the two chrominance components.

17.1.1.3 Coding and Decoding Basics

- Prediction—Motion-compensated interpicture prediction (as in H.261 and MPEG-1/2) is supported for high efficiency, although the use of motion compensation is optional at the encoder. If no prediction is applied, the coding mode is said to be Intra; otherwise it is Inter. There are two basic picture types, I-pictures and P-pictures. In I-pictures, macroblocks are coded in Intra mode only whereas in P-pictures, both Intra and Inter modes may be used as indicated by macroblock type. The optional PB-frames are always coded in the Inter mode. B-pictures are partly coded bidirectionally.

- Motion Compensation—The decoder normally accepts one motion vector per macroblock, or either one or four motion vectors per macroblock when Advanced Prediction is used. If the PB-frame mode is used, one additional delta motion vector is transmitted per macroblock of B-pictures and is used for refinement of motion vectors derived by scaling of motion vectors of the corresponding macroblock in P-pictures.
- Quantization—The number of quantizers is 1 for the first coefficient (DC coefficient) of Intra blocks and is 31 for all other coefficients of an Intra block as well as all coefficients in an Inter block. All coefficients except for the first coefficient use equally spaced reconstruction levels with a central dead zone around zero.
- Coding Control—A number of parameters can be varied to control the rate of generation of coded video data. These include preprocessing, quantizer selection, block coding mode selection, and temporal subsampling and are encoding issues and thus not standardized.

17.1.2 Syntax and Semantics

The video bitstream is arranged in a hierarchical structure composed of the following layers:

- Picture Layer
- Group of Blocks Layer
- Macroblock Layer
- Block Layer.

The syntax diagrams of each of these layers is shown in Fig. 17.1.

For each of the layers, the abbreviations used, their meaning, and semantics are discussed next.

17.1.2.1 Picture Layer

This represents the highest layer in bitstream; the structure of this layer is shown in Fig. 17.2. Each coded picture consists of a picture header followed by coded picture data, arranged as Group of Blocks (GOBs). After all pictures belonging to a sequence are sent, an End-of-Sequence (EOS)

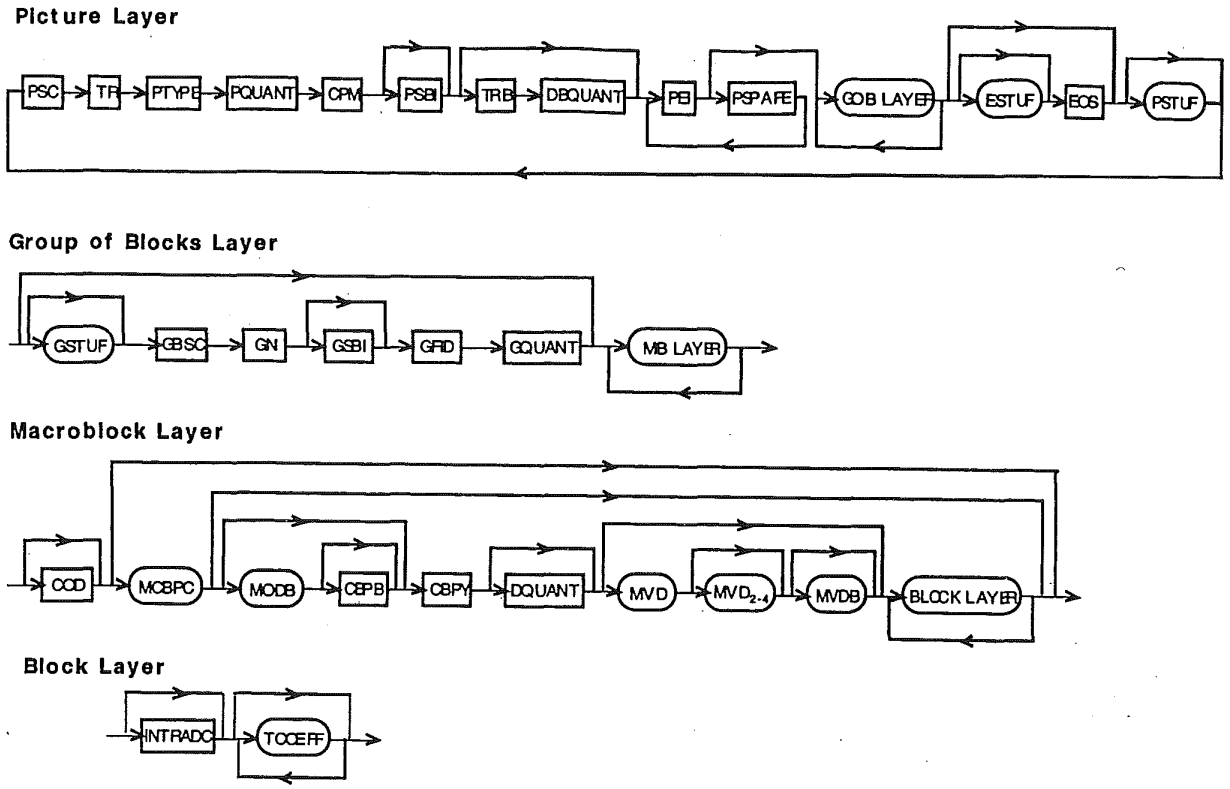


Fig. 17.1 Syntax diagram of various layers in the H.263 bitstream.

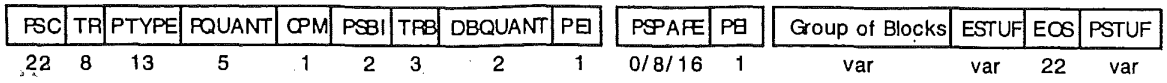


Fig. 17.2 Structure of Picture layer in the H.263 bitstream.

code and stuffing bits (ESTUF) is sent. The presence of some elements is conditional. For instance, if the Continuous Prescence Multipoint (CPM) indicates it, a Picture Sub-bitstream Indicator (PSBI) is sent. Likewise, if picture type (PTYPE) indicates a "PB-frame" Temporal Reference for B-pictures (TRB) and a B-picture Quantizer (DBQUANT) is sent. Combinations of Picture Extra Insertion Information (PEI) and Picture Spare Information (PSPARE) may not be present. EOS may not be present, while ESTUF may be present only if EOS is present. Further, if during the coding process some pictures are skipped, their picture headers are not sent. Some of the syntax elements in this layer are represented by variable length codes while others are of fixed length as shown in the figure.

The Picture Start Code (PSC) is 22 bits in length represented by 0000 0000 0000 0000 1 00000. All PSCs are byte aligned by using PSTUF bits before PSC.

Temporal Reference (TR) is 8 bits allowing 256 possible values. To generate TR for the current picture the TR of the previously transmitted picture is incremented by the number of nontransmitted pictures. When the optional PB-frame mode is employed, TR refers to P-pictures only, the TR of B-pictures is represented by TRB, a 3-bit code that represents the number of nontransmitted pictures since the last P- or I-picture and before the B-picture.

Picture type, PTYPE, is a 13-bit code, with bits 6 to 8 representing source format ("001" sub-QCIF, "010" QCIF, "011" CIF, "100" 4CIF, "101" 16CIF); bit 9 representing picture coding type ("0" I-picture, "1" P-picture); and bits 10, 11, 12, and 13 identifying the presence or absence of optional modes Unrestricted Motion Vector, Syntax-Based Arithmetic Coding, Advanced Prediction Mode, and PB-frames. Other bits in PTYPE are as follows: bit 1 is always "1" to avoid start code emulation; bit 2 is always "0" for distinction with H.261; bits 3, 4, and 5 respectively correspond to split screen indication, document camera indication, and freeze picture release indication.

The Picture Quantizer (PQUANT) is a 5-bit code that represents quantizer values in the range of 1 to 31. DBQUANT is a 2-bit code representing BQUANT ("00" 5 × QUANT/4, "01" 6 × QUANT/4, "10" 7 × QUANT/4, "11" 8 × QUANT/4) such that QUANT is used in the P-block and BQUANT in the corresponding B-block.

17.1.2.2 Group of Blocks Layer

Data for the GOB layer consists of a GOB header followed by data for macroblocks. The structure of the GOB layer is shown in Fig. 17.3. For the first GOB (GOB number 0) in each picture, the GOB header is not transmitted, whereas for other GOBs, depending on the encoder decision, the GOB header may be empty. A decoder, if it so desires, can signal using an external means, the encoder, to send nonempty GOB headers. Group stuffing (GSTUF) may be present when Group of Blocks Start Code (GBSC) is present. Group Number (GN), GOB Frame ID (GFID), and GOB quantizer (GQUANT) are present when GSBSC is present. Further, the GOB Sub-bitstream indicator (GSBI) is present if the Continuous Presence mode is indicated by CPM.

GBSC is a 17-bit code with value 0000 0000 0000 0000 1 and may be byte aligned by using GSTUF. The group number is a 5-bit code identifying the GOB number, which can take values from 0 to 17, while values from 18 to 30 are

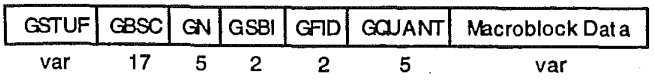


Fig. 17.3 Structure of Group-of-Blocks layer in H.263.

reserved for the future and a value of 31 indicates the end of sequence (EOS). The GOB number 0 is used in the PSC and for this value GOB header including GSTUF, GBSC, GN, GSBI, GFID, and GQUANT is empty. The length of GSBI and GFID codes is 2 bits each whereas for GQUANT it is 5 bits.

17.1.2.3 Macroblock Layer

Data for each macroblock consists of a macroblock header followed by data for blocks. The structure of the Macroblock layer is shown in Fig. 17.4. A coded macroblock indication (COD) is present for each macroblock in P-pictures. A macroblock type and Coded block pattern for chrominance (MCBPC) is present when indicated by COD or when PTYPE is an I-picture. A macroblock mode for B-pictures (MODB) is present for NonIntra MB types if PTYPE indicates a PB-frame. A coded block pattern for luminance (CBPY), codes for differential quantizer (DQUANT), motion vector data MVD, and MVD_{2-4} are present when indicated by MCBPC. The coded block pattern for B-blocks (CBPB) and motion vector data for the B-macroblock (MVDB) are present only if indicated by MODB. Block data are present when indicated by MCBPC and CBPY. MVD_{2-4} is present only in the Advanced Prediction mode. MODB, CBPB, and MVDB are present only in the PB-frame mode.

COD is a 1-bit codeword; its value when "0" indicates that a macroblock is coded and when "1" indicates that no further information is sent for that macroblock. A COD of "1" thus implies an Inter macroblock with zero motion vector and no DCT coefficient data. COD is present for each macroblock in pictures where PTYPE indicates a P-picture.

MCBPC is a variable length codeword that jointly represents macroblock type and coded block pattern for chrominance; it is always included for coded macroblocks. Macroblock type provides information about the coding mode and which data elements are present. The coded block pattern for chrominance (CBPC) indicates whether at least one NonIntraDC DCT coefficient is transmitted (IntraDC is dc coefficient for Intra blocks). The first and second bits of CBPC indicate the status of each of the chrominance blocks in a macroblock; either of the bits when set to "1" indicates the presence of NonIntraDC coefficients for that block. Owing to different macroblock types in I- and P-pictures, separate VLC tables representing MCBPC for these pictures are used. For I-pictures, macroblock types allowed are Intra and Intra + Q (Intra with quantizer update). For P-pictures the macroblock types are Inter, Inter + Q, Inter4V, Intra, and Intra + Q. The Inter4V mode allows use of four motion vectors per macroblock when the Advanced prediction option is employed.

For PB-frames, the macroblock types allowed are the same as that for P-pictures; however, for these frames, MODB is present for each macroblock and indicates if CBPB or MVDB is also present. CBPB is a 6-bit code with each bit signalling the status of a B-block in the macroblock; if a corresponding bit is set to "1" it indicates the presence of DCT coefficients for that block.

CBPY is a variable length code that represents a pattern of Y blocks in the macroblock for which at least one NonIntraDC DCT coefficient is transmitted.

DQUANT is a 2-bit code indicating four possible changes in values of quantizers, −1, −2, 1, and 2, over the value of QUANT which ranges from 1 to 31. After addition of differential values the values are clipped to lie in the range of 1 to 31.

Motion vector data (MVD) are included for all Inter macroblocks and consist of a variable length code for the horizontal component followed by a variable length code for the vertical component. Three additional codewords, MVD_{2-4}, are also included if indicated by PTYPE and by MCBPC and each consists of a variable length code for the horizontal component followed by a variable length code for the vertical component. MVD_{2-4}

COD	MCBPC	MODB	CBPB	CBPY	DQUANT	MVD	MVD_2	MVD_3	MVD_4	MVDB	Block Data
1	var	var	6	var	2	var	var	var	var	var	var

Fig. 17.4 Structure of the macroblock layer in H.263.

are present when in the Advanced Prediction mode.

For B-macroblocks, MVDB is present if indicated by MODB and also consists of a variable length code for the horizontal component followed by a variable length code for the vertical component. MVDB is present in PB-frames only if indicated by MODB.

17.1.2.4 Block Layer

In normal mode (not PB-frame mode), a macroblock consists of four luminance blocks and two chrominance blocks. The structure of the Block layer is shown in Fig. 17.5. IntraDC is present for every block of the macroblock if MCBPC indicates the MB type of Intra or Intra + Q. TCOEF is present if indicated by MCBPC or CBPY.

In PB-frames, the macroblock can be thought of as being composed of 12 blocks. First, the data for six P-blocks is transmitted followed by data for the next six B-blocks. An IntraDC is sent for every P-block of the macroblock if the MCBPC indicates an MB type of Intra or Intra + Q. IntraDC is not present for B-blocks. TCOEFF is present for P-blocks if indicated by MCBPC or CBPY; TCOEF is present for B-blocks as indicated by CBPB.

When the Syntax-Based Arithmetic Coding option is employed, the block layer is different and is explained later.

IntraDC is coded with 8 bits with values 0000 0000 and 1000 0000 not used. The reconstruction levels of 8, 16, 24, ... 1016 are represented by corresponding codewords 0000 0001, 0000 0010, 0000 0011, ... 0111 1111. The reconstruction levels of 1032, 1040, 1048, ... 2032 are represented by corresponding codewords 1000 0001, 1000 0010, 1000 0011, ... 1111 1110. The reconstruction level of 1024 is signalled by codeword 1111 1111.

The most frequently occurring EVENTs are variable length coded. An EVENT is a combination of a last nonzero coefficient indication (LAST; "0": there are more nonzero coefficients in this block, "1"; this is the last nonzero coefficient in this block), the number of successive zeros preceding the coded coefficient (RUN), and the nonzero value of the coded coefficient (LEVEL). In contrast, the EVENTs in VLC coding in MPEG-1 and MPEG-2 include the (RUN, LEVEL) combination and a separate code to signal end-of-block (EOB), instead of the (LAST, RUN, LEVEL) combination employed by the H.263. The remaining (less frequently occurring EVENTs) are coded by 22-bit codewords comprised of 7 bits of ESCAPE (0000 011), 1 bit of last (1 or a 0), 6 bits of RUN (0000 00 to 1111 11), and 8 bits of LEVEL. LEVELs of $-127, \ldots -1$ are represented by corresponding codewords 1000 0001, ... 1111 1111. LEVELs of $1, \ldots 127$ are represented by corresponding codewords 0000 0001, ... 0111 1111. Further, LEVELs of -128 and 0 are FORBIDDEN.

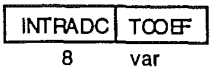


Fig. 17.5 Structure of the Block layer in H.263.

As a point of comparison, the dynamic range of LEVELs in MPEG-1 is -255 to $+255$ and in MPEG-2 it is $+2047$ to -2047 as compared to H.263 in which the range is only -127 to $+127$.

17.1.3 The Decoding Process

We now describe the decoding process which basically consists of motion compensation, coefficients decoding, and block reconstruction.

17.1.3.1 Motion Compensation

Motion compensation in H.263 may simply use the default prediction mode and in addition may use either one or both optional modes such as the Unrestricted Motion Vector mode or the Advanced Prediction mode. For now, we discuss the default prediction mode and will discuss the various optional modes a little later.

In default mode when coding P-pictures, each Inter macroblock has one motion vector. This vector is obtained by adding predictors to the vector differences indicated by MVD. Predictors for differential coding are taken from three surrounding macroblocks as shown in Fig. 17.6. The predictors are calculated separately for the horizontal and vertical components.

At the boundary of a GOB or picture, predictions are calculated according to a set of decision rules applied in the following sequence. The imme-

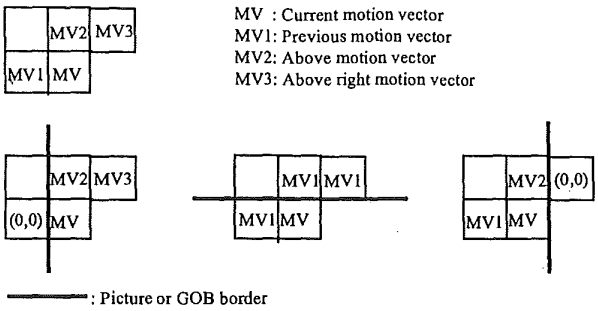


Fig. 17.6 Motion vector prediction in H.263.

diately previous candidate (left) predictor MV1 is set to zero if the corresponding macroblock is outside the picture. Then, the immediately above candidate (top) predictor MV2, and right of immediately above (top right) predictor MV3 are set to MV1 if the corresponding macroblocks are outside the picture (at the top) or outside the GOB (at the top) if the GOB header of the current GOB is nonempty. Next, the candidate predictor MV3 is set to zero if the corresponding macroblock is outside the picture (at the right side). Finally, if the corresponding macroblock is Intra coded or was not coded (COD = 1), the candidate predictor is set to zero.

Once the values of each of the three predictors MV1, MV2, and MV3 are determined, the actual predictor is obtained by taking median values of the three predictors for the horizontal and vertical components.

The value of each component of the motion vector is constrained to lie in the range of [−16, +15.5]. Each VLC word for MVD is used to represent a pair of difference values, only one of which is a valid value falling in the range [−16, +15.5] depending on the value of the predictor.

A positive value of the horizontal or vertical component of the motion vector means that the prediction is being formed from pixels in the previous pictures which are spatially to the right or below the pixels being predicted. The same motion vector is used for all four luminance blocks in a macroblock. Motion vectors for chrominance blocks are obtained by dividing each of the motion vector components by 2, since chrominance resolution is half of that of the luminance resolution in each direction. Since, after division, the chrominance vectors can now in fact be at quarter pixel resolution, they are truncated to the nearest half-pixel position.

Luminance or chrominance intensities at half-pixel position are calculated by bilinear interpolation as in MPEG-1/2.

17.1.3.2 Coefficient Decoding

If LEVEL = "0," the reconstructed value is given by REC = "0." The reconstructed values of IntraDC is simply 8 × LEVEL, except for decoded level of 255, which indicates REC = "1024." The reconstructed values of nonzero coefficients other than IntraDC are calculated as follows:

$$|REC| = (2 \times |LEVEL| + 1) \times QUANT \quad \text{when QUANT = "odd"}$$

$$|REC| = ((2 \times |LEVEL| + 1) \times QUANT) - 1 \quad \text{when QUANT = "even"}$$

The coefficient reconstruction process disallows even values at the output to prevent IDCT mismatch errors. The value of REC is calculated from |REC|as follows:

$$REC = \text{sign(LEVEL)} \times |REC|; \quad \text{where sign(LEVEL) is given by last bit of TCOEFF code.}$$

After inverse quantization, the reconstructed values of all coefficients other than IntraDC are clipped to lie in the range −2048 to 2047.

The reconstructed DCT coefficients are placed into an 8 × 8 block along the zigzag path, essentially undoing the zigzag scan performed at the encoder.

After inverse quantization and zigzag placement of coefficients, the resulting 8 × 8 block is inverse DCT transformed as in case of MPEG-2, discussed earlier in the book. The output of the inverse transform is 9 bits with a range of −255 to +255.

17.1.3.3 Block Reconstruction

After motion compensation and coefficients decoding, luminance and chrominance blocks are reconstructed. If the blocks belong to an Intra macroblock, the inverse transformation already

provides the reconstruction. If the macroblock is Inter, the reconstructed blocks are generated by adding prediction on a pixel basis to the result of the inverse transformation. A clipping operation is then performed to ensure that the reconstructed pixels lie in the range of 0 to 255.

17.1.4 Optional Modes

Next, we discuss the optional modes in H.263. As mentioned earlier, there are four such modes and the capability of such modes is signalled by the decoder to the encoder by external means (such as ITU-T Recommendation H.245). The H.263 bitstream is self sufficient and carries information about the optional modes in use via the PTYPE information.

17.1.4.1 Unrestricted Motion Vector Mode

The Unrestricted Motion Vector mode removes the restriction of the default prediction mode which forces all referenced pixels in prediction to be from inside the coded picture area. Thus, in this mode, motion vectors are allowed to point outside of the picture. When a pixel referenced by a motion vector is outside of the coded picture area, an edge pixel is used instead and is found by restricting the motion vector to the last full pixel inside the coded picture area. The restriction of motion vector is performed on a pixel basis and separately for each component of the motion vector.

Thus, for example, if a picture size of QCIF (picture size 176 × 144) is used, and if prediction needs to access a pixel at an x-coordinate larger than 175, an x-coordinate of 175 is used instead; or if prediction needs to access a pixel at x-coordinate of less than 0, then an x-coordinate of 0 is used. For the same example, if prediction needs to access a pixel at a y-coordinate larger than 143, a y-coordinate of 143 is used, or if prediction needs to access a pixel at a y-coordinate of less than 0, then a y-coordinate of 0 is used.

Earlier, we noted that in the default prediction mode, the value of each component of motion vector is limited to [−16, 15.5]. This range is clearly insufficient in many cases and thus in Unrestricted Motion Vector mode, this range is extended to [−31.5, 31.5] with the constraint that only values in the range of [−16, 15.5] around the predictor for each motion vector component can be reached if the predictor is in the range [−15.5, 16]. If the predictor is outside this range, all values in the range of [−31.5, 31.5] with the same sign as the predictor and the zero value can be reached.

With the unrestricted Motion Vector mode all motion vector data (MVD, MVD_{2-4}, MVDB) are interpreted as follows. If the predictor for the motion vector component is in the range [−15.5, 16] only the first of the pair of vector differences applies. If the predictor is outside the motion vector component range [−15.5, 16], the vector difference from pairs of vector differences will be used which results in a vector component in range [−31.5, 31.5] with the same sign as the predictor, including zero.

17.1.4.2 Syntax Based Arithmetic Codes

In, MPEG-1, MPEG-2, and the default coding mode of H.263, each symbol is VLC encoded using a specific table based on the syntax of the coder. The table typically stores lengths and values of VLC codewords. The symbol is mapped to an entry by table look-up and the binary codeword specified by the entry is sent to the buffer for transmitting to the receiver. In VLD decoding, the received codewords, depending on the context, are compared against appropriate tables based on the syntax of the coder to determine matching entries and these actual symbols resulting from VLD decoding are an interim step in the video decoding process. Of course, the tables used at the encoder and the decoder must be exactly the same. The VLC/VLD process implies that each symbol must be coded in integer number of bits and removing this restriction can result in increase in coding efficiency; this can be accomplished by arithmetic coding.

In Syntax-Based Arithmetic coding (SAC), all the corresponding variable length coding/decoding operations of H.263 are replaced with arithmetic coding/decoding operations. The capability of this mode is signalled by external means while the use of this mode is indicated by PTYPE.

In the SAC encoder, a symbol is encoded by a specific array of integers (a model) based on the

syntax of the coder and by calling a procedure called *encode_a_symbol (index, cumul_freq)*. A FIFO (first-in–first out) called PSC_FIFO is used for buffering the output bits from the arithmetic encoder. The model is specified through cumul_freq[ ], and the symbol is specified using its index in the model.

In the SAC decoder, a symbol is decoded by using a specific model based on the syntax and by calling a procedure called *decode_a_symbol (cumul_freq)*. As in the encoder, the model is specified via cumul_freq[ ]. The decoded symbol is returned through its index in the model. A FIFO similar to one at the encoder called PSC_FIFO is used for buffering the incoming bitstream. The decoder is initialized by calling a procedure called *decoder_reset ( )* to start decoding an arithmetic coded bitstream.

As discussed earlier, the H.263 syntax can be thought of as composed of four layers: Picture, Group-of-Blocks, Macroblock, and Block. The syntax of the top three layers, Picture, Group-of-Blocks, and Macroblock, remain exactly the same between variable length coding (default) or SAC mode. The syntax of the fourth layer, the Block layer, is slightly modified as shown in Fig. 17.7.

TCOEF1, TCOEF2, TCOEF3, and TCOEFr are symbols representing (LAST, RUN, LEVEL) EVENTs, and possibly can be the first, second, third, and the rest of the symbols respectively. Thus TCOEF1, TCOEF2, TCOEF3, and TCOEFr are present only when one, two, three, or more coefficients respectively are present in the block layer.

At the encoder or at the decoder, the PSC_FIFO, a FIFO of size >17 bits is used. In PSC_FIFO at the encoder, illegal emulations of PSC and GBSC are located and are avoided by stuffing a "1" after every successive appearance of 14 "0"s, which are not part of PSC or GBSC). In PSC_FIFO of the decoder, the first "1" after every string of 14 "0"s is deleted; if instead of a string of 14 "0"s is followed by a "0," it indicates legal PSC or GBSC is detected with the exact location of PSC or GBSC determined by the next "1" following the string of zeros.

Fixed Length Symbols form three possible strings PSC–TR–PTYPE–PQUANT–CPM–(PSBI)–(TRB–DSQUANT)–PEI–(PSPARE–PEI–. . .), (GSTUF)–GBSC–GN–(GSBI)–GFID–GQUANT, and (ESTUF)–(EOS)–(PSTUF). These strings are directly sent to PSC_FIFO as in the normal VLC mode of the H.263 encoder, and are directly sent out from PSC_FIFO in the decoder after a legal PSC or GBSC or EOS is detected. If a fixed length string is not the first in a video session, the arithmetic encoder is reset before sending the fixed length string by calling a procedure called *encoder_flush ( )*. This procedure is also called at the end of video session or before EOS. At the decoder side, after each fixed length symbol string, a procedure *decoder_reset ( )* is called.

Nonfixed Length Symbols employ precomputed models with a separate model for each symbol type. All models are stored as a one-dimensional array whose values are accessed by using the same indices as used in normal VLC coding. The models for COD and MCBPC in P-pictures are cumf_COD [ ] and cumf_MCBPC[ ]. The model for MCBPC in I-pictures is cumf_MCBPC_intra[ ]. The model for MODB is cumf_MODB[ ]. For CBPB, two models are used, one for luminance and the other for chrominance and are respectively

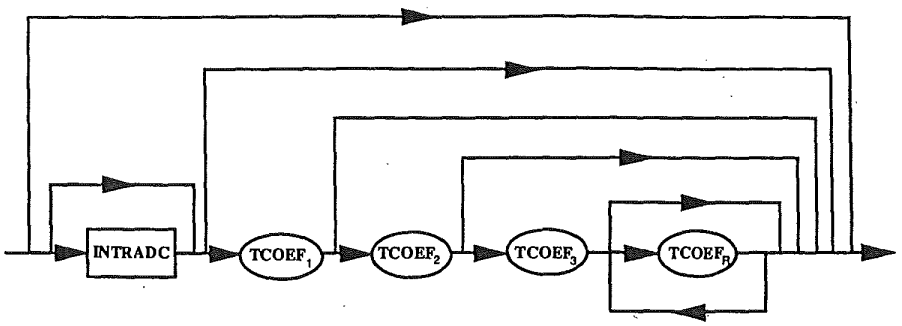


Fig. 17.7 Structure of SAC block layer in H.263.

called cumf_YCBPB [ ] and cumf_UVCBPB [ ]. The model for CBPY is cumf_CBPY [ ] in Inter macroblocks and cumf_CBPY_intra [ ] in Intra macroblocks. The model for DQUANT is cumf_DQUANT[ ]. For all MVDs, a single model, cumf_MVD[ ], is used. The model for INTRADC is cumf_INTRADC[ ]. The models for TCOEF1, TCOEF2, TCOEF3, and TCOEFr in Inter blocks are cumf_TCOEF1[ ], cumf_TCOEF2[ ], cumf_TCOEF3[ ], and cumf_TCOEFr[ ]; the corresponding models in Inter blocks are cumf_TCOEF1_intra[ ], cumf_TCOEF2_intra[ ], cumf_TCOEF3_intra[ ], and cumf_TCOEFr_intra[ ]. A separate model is used for coefficient sign and is cumf_SIGN[ ]. Finally, the models for LAST, RUN, LEVEL after ESCAPE in Inter blocks are cumf_LAST[ ], cumf_RUN [ ] and cumf_LEVEL [ ] and the corresponding models in Intra blocks are cumf_LAST_intra[ ], cumf_RUN_intra[ ], and cumf_LEVEL_intra[ ].

For details of procedures *encode_a_symbol (index, cumul_freq), decode_a_symbol (cumul_freq), encoder_flush ( ), decoder_reset ( ),* and the actual models used consult the H.263 standard.

17.1.4.3 Advanced Prediction Mode

H.263 allows yet another option to allow improved prediction; this option, like other options of H.263, is signalled through external means and when used is identified in the H.263 bitstream in the PTYPE information. This option is called the Advanced Prediction Mode and includes overlapped block motion compensation and the possibility of using four motion vectors per macroblock. In this mode, like in the Unrestricted Motion Vector Prediction mode, motion vectors can refer to an area outside of the picture for prediction. However, the extended motion vector range feature of the Unrestricted Motion Vector mode is active only if that mode is explicitly selected. If the Advanced Prediction mode is used in combination with the PB-frame mode, overlapped motion compensation is used only for prediction of P-pictures and not for B-pictures.

In the default mode of H.263, one motion vector is used per Inter macroblock in P-pictures. However, when the Advanced Prediction Mode is used, either one or four motion vectors can be used as indicated by the MCBPC codeword. If information about only one motion vector MVD is transmitted, this is interpreted as four vectors each with the same value. If MCBPC indicates that four motion vectors are transmitted, the first motion vector is transmitted as codeword MVD and information about three additional vectors is transmitted as MVD_{2-4}. The vectors are obtained by adding predictors to vector differences MVD and MVD_{2-4} in a similar manner when only one motion vector is present with the caveat that candidate predictors are defined for each 8 × 8 block of a macroblock as shown in Fig 17.8.

The motion vector for chrominance is computed for both chrominance blocks by the sum of the four luminance vectors and dividing the sum by 8, separately for the horizontal and the vertical components. This division can result in chrominance vectors to $^1/_{16}$ pixel accuracy and have to be rounded to the nearest half-pixel position. The luminance or chrominance values at half-pixel position are calculated by bilinear interpolation.

We now discuss *Overlapped Motion Compensation,* a very important feature of the Advanced Prediction mode. In overlapped motion compensation, each pixel in an 8 × 8 luminance prediction block is computed as a weighted sum of three predictions. The three predictions are calculated by using three motion vectors: a vector of current luminance block and two out of four "remote" motion vectors. The "remote" vector candidates are the motion vector of the block to the left or the right side of the current luminance block, and the

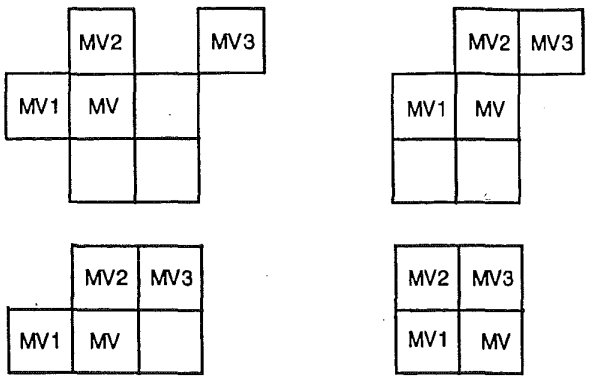


Fig. 17.8 Candidate predictors MV1, MV2, MV3 for each luminance block of a macroblock in H.263.

motion vector of the block above or below the current luminance block. Remote motion vectors from other GOBs are used in the same way as the motion vector within a GOB. For each pixel, remote motion vectors of the block corresponding to the two nearest block borders are used. Thus, for the upper half of the block, the motion vector corresponding to the block above the current block is used, while for the lower half of the block, the motion vector corresponding to the block below the current block is used. The weighting values used in conjunction with this type of prediction with motion vectors of luminance blocks are shown in Fig. 17.9b. Similarly, for the left half of the block the motion vector corresponding to the block on the left side of the current block is used, while for right half of the block the motion vector corresponding to the block on the right side of the current block is used. Again, the weighting values used in conjunction with this type of prediction with motion vectors of luminance blocks are shown in Fig. 17.9c.

Each pixel $p(i,j)$ in the 8×8 luminance prediction block is calculated as follows:

$$p(i,j) = (q(i,j) \times H_0(i,j) + r(i,j) \times H_1(i,j) + s(i,j) \times H_2(i,j) + 4)/8$$

where $q(i,j)$, $r(i,j)$, and $s(i,j)$ are pixels from referenced picture obtained using motion vectors MV^0, MV^1, and MV^2, such that MV^0 represents the motion vector of the current block, MV^1 represents the motion vector of the block either above or below, and MV^2 represents the motion vector either to the left or to the right of the current block. As always, each MV is in fact a pair (MV_x, MV_y) representing horizontal and vertical components.

If some motion vectors are not available such as when a surrounding block is not coded or it is coded as Intra, the corresponding remote motion vector is replaced by the motion vector of the current block (except in the PB-frame mode). If the current block is at the border of a picture and a surrounding block is not present, the corresponding remote motion vector is replaced by the current motion vector.

The weighting values in terms of matrices $H_0(i,j)$, $H_1(i,j)$, and $H_2(i,j)$ are shown in Figure 17.9a, b, and c, respectively.

17.1.4.4 PB-Frames Mode

The last optional mode in H.263 is called the PB-frames mode. Like other optional modes discussed thus far, its capability is signalled by external means and when this mode is enabled it is indicated by PTYPE information as discussed earlier.

A PB-frame consists of a pair of pictures coded together as a single unit. A PB-frame consist of a normal P-picture coded with prediction from a previously decoded reference picture which may be an I- or a P-picture, and a B-picture coded with bidirectional prediction from a decoded past reference picture and a decoded future reference picture. In fact, the B-picture in the PB-frame mode of H.263 is a simplified and overhead reduced version of MPEG-1 (or MPEG-2) B-pictures. The prediction process of H.263 B-pictures is illustrated in Fig. 17.10.

When using PB-frames, the interpretation of Intra macroblocks is revised to mean P-blocks are Intra coded and B-blocks are Inter coded with prediction as for an Inter block. For Intra macroblocks in P-pictures, motion vectors are included for use by corresponding B-blocks in B-pictures of PB-

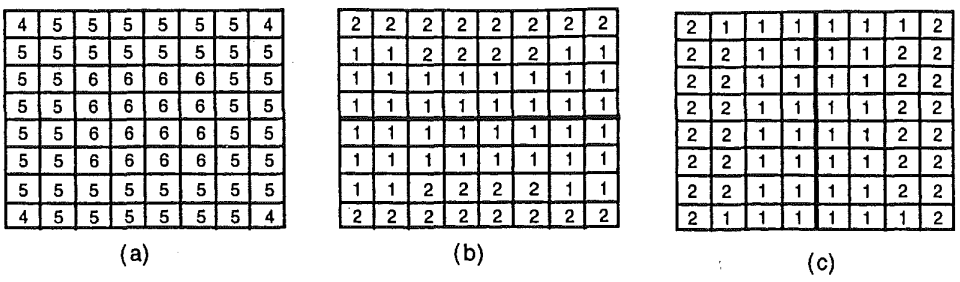


Fig. 17.9 Weighting values (a) H0, (b) H1, (c) H2, used for overlapped motion compensation in H.263.

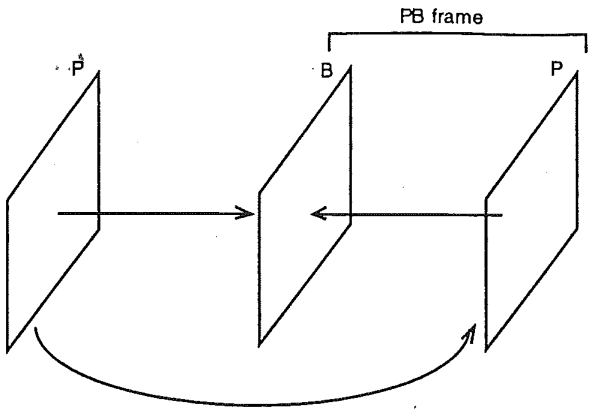


Fig. 17.10 Prediction in PB-frame mode of H.263.

frames. Further, when using the Advanced Prediction mode and the PB-frame mode, if one of the surrounding blocks is coded Intra, the corresponding remote motion vector is not replaced by the current block motion vector but uses the remote Intra motion vector instead.

As noted earlier, in the PB-frame mode a macroblock is considered to consist of 12 blocks. First the data for six P-blocks are transmitted, next followed by data for six B-blocks. If the macroblock is Intra in the P-picture, IntraDC is present for each P-block; however, IntraDC is not present for B-blocks. TCOEF is present for P-blocks if indicated by MCBPC or CBPY; TCOEF is present for B-blocks if indicated by CBPB.

The vectors for a B-picture are calculated using motion vectors available for the corresponding P-block in the P-picture. Assume the motion vector of a P-block is MV; then we calculate a forward motion vector MV_F and a backward motion vector MV_B from MV and enhance it by a delta motion vector given by MVDB. Further, let TR_D be the increment in temporal reference TR from the last picture header (if TR_D is negative, $TR_D = TR_D + 256$), and TR_B be the temporal reference for the B-picture and represents the number of non-transmitted pictures since the last I-or P-picture, just before the B-picture. Let MV_D be the delta motion vector for a B-block derived from MVDB and corresponds to the vector MV of a P-block. If MVDB is not present, MV_D is set to zero; otherwise, the same MV_D given by MVDB is used for each of the four luminance B-blocks in a macroblock.

The forward vector MV_F and the backward vector MV_B in half-pixel units are calculated as follows:

$$MV_F = (TR_B \times MV)/TR_D + MV_D$$

$$MV_B = ((TR_B - TR_D) \times MV)/TR_D \quad \text{if } MV_D \text{ is not equal to 0}$$

$$MV_B = MV_F - MV \quad \text{if } MV_D \text{ is not equal to 0}$$

Similar to that for MV, the range of values for MV_F is constrained. Also, a VLC word for MVDB represents a pair of difference values with only one value of MV_F falling in the default range of [−16, 15.5] or [−31.5, 31.5] in Unrestricted Motion Vector mode. For chrominance blocks, MV_F is calculated by sum of four luminance MV_F vectors, dividing the sum by 8, and truncating to nearest half-pixel; the MV is calculated similarly using four luminance MV_B vectors.

To calculate prediction for an 8 × 8 B-block in a PB-frame, first the forward and the backward vectors are calculated. It is assumed that the P-macroblock is first decoded and the reconstructed value P_{REC} is calculated. The prediction for B-block is calculated next and consists of two parts. First, for pixels where the backward vector, MV, points to inside P_{REC}, use bidirectional prediction which is computed as average of forward prediction obtained by MV_F and backward prediction obtained by MV_B. For all other pixels, forward prediction using MV_F relative to the previous decoded picture is used. Figure 17.11 illustrates the part of the block that is predicted bidirectionally and the remaining part that is predicted by forward prediction.

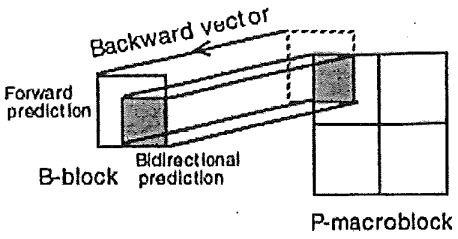


Fig. 17.11 Forward and bidirectional prediction for the B-block.

17.1.5 The Encoder

In Fig. 17.12 we show a high-level structure of a typical interframe encoder that can be used for the H.263 standard. This encoder is very similar to the encoder used for MPEG-1 encoding and uses motion estimation and compensation to generate an interframe prediction signal which is analyzed on a macroblock basis to decide if Intra or Inter coding should be employed. If Intra coding is employed, the prediction signal is ignored, and the block is transformed to a block of coefficients by DCT and quantized by Quant and Quant idices are sent along with the Inter/Intra decision and quantizer scale information to the video multiplex coder which generates the H.263 bistream. If Inter coding is employed, the motion-compensated prediction macroblock is differenced from the macroblock being coded and the prediction error signal is transformed by DCT, quantized by Quant, and results in Quant indices which along with motion vectors, Inter/Intra flag, transmit or not flag, and quantizer scale are sent to the video multiplex coder.

Details of motion estimation and compensation are not shown; depending on the coding options selected, motion estimation and compensation may involve computations specific to Unrestricted Motion Vectors, Advanced Prediction, or PB-frames. Also, the scaling of luminance motion vectors to generate chrominance motion vectors and calculation of bilinear interpolation for half-pixel motion compensation is not shown. Video multiplex uses various VLC tables and incoming flags to correctly format the bitstream to be H.263 compliant.

17.2 CODING ADVANCES

While various key standardization efforts related to image and video coding have been ongoing for the last ten years or so, outside of the standards arena, considerable progress has also been ongoing,[15] leading to a refinement of well-established coding schemes, gradual maturity of recent less established coding schemes, and invention of new promising coding schemes. Not only in terms of performance but also in terms of implementation complexity a number of coding schemes are beginning to be competitive with standards-based video coding. Thus, before proceeding with our discussion on MPEG-4, which is currently an ongoing standard, a brief review of the state of the art is necessary.

In reality, the MPEG standardization process has been fairly open such that whenever a new standard is initiated, a call for proposals is sent,

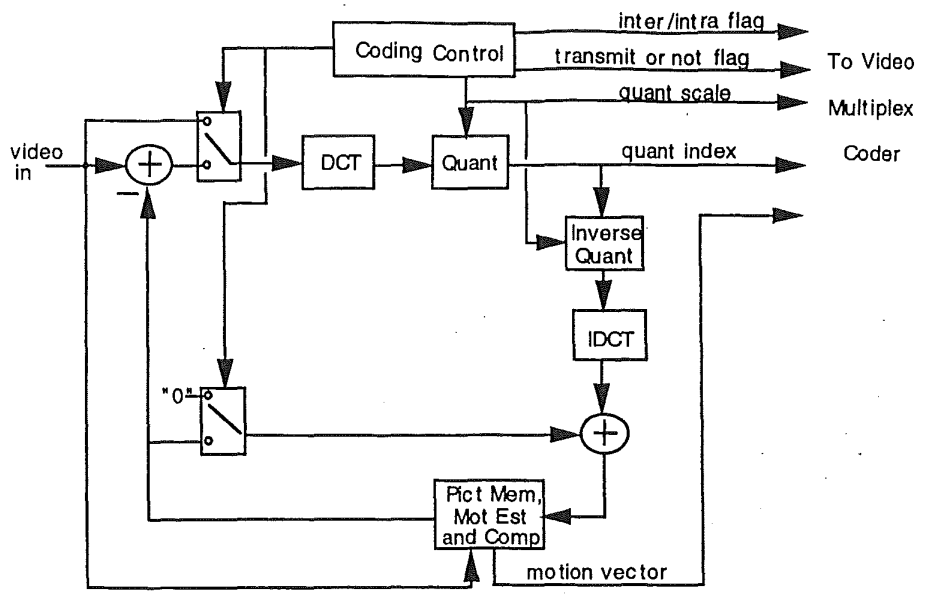


Fig. 17.12 Encoder structure.

inviting any candidate proposals to an open "contest" involving subjective testing of video quality, evaluation of functionalities it can achieve, and analysis of its implementation complexity. Further, even if a candidate proposal is not among the top rated proposals but consists of some promising components, as long as there are organizations willing to study them further, these components are evaluated and even improved via Core Experiments.

There are various ways of classifying image and video coding schemes. In one such technique for classification, terms such as first generation coding, second generation coding, and so forth have been used. The underlying implication of this technique for classification is that the second generation schemes analyze a scene and thus can code it more efficiently by using a contours (edges separating various regions) and texture (intensity variations in various regions) based representation, while the first generation schemes do not. Another technique for classification is based on clearly identifying the segmentation and motion models employed by a scheme. Various possibilities for segmentation models are: statistically dependent pixel blocks, 2D models based on irregular regions/deformable triangles/specific features, and 3D models such as wireframe representations. Likewise, possibilities for motion models are: 2D translation, sophisticated 2D models based on affine/bilinear/perspective transformations, and 3D models with global or local motion. Another technique for classification, besides coding efficiency, uses additional functionalities that a coding scheme can provide as a criterion for judging its value. Examples of functionalities are degree of scalability, error resilience, content based access, and others.

The performance, functionality, and complexity tradeoffs are different for different applications, and in general there are no easy answers. It is difficult for a single coding algorithm to be able to provide significantly higher compression than current standards while achieving desirable functionalities and somewhat reasonable complexity. Further, an important question is how general should a coding scheme be? Should it be very application specific, or should it be very generic? One way to answer these questions is in terms of coding efficiency/functionalities achieved. If very significant coding/functionality improvements can be obtained by being application specific, it should be application specific; otherwise it should be generic. In fact one of the motivations in studying various techniques for classification is to determine trends and predict which methods are likely to lead to significant improvements in the future.

In presenting a comparative overview of various recent coding advancements, one of the main difficulties is that one is judging the performance of an overall coding system, and not that of individual components. To be more concrete, a typical image or video coding system consists of a number of components such as analyzer, quantizer, and entropy coder, with a number of possibilities for each component. Further, analysis may itself be done on an entire signal or on portions of the signal, on the original signal, or on the prediction error (including motion-compensated prediction error) signal, and moreover may use DCT transform, Sub-band, Discrete Wavelet Transform, Fractals, or others. The next component is quantization which may be perceptual visibility based or not, fixed or adaptive, linear or nonlinear, scalar or vector. The third major component is entropy coding which may again be fixed or adaptive, Huffman (VLC), or arithmetic. Recognizing the various pitfalls, we briefly review the state of progress of various promising coding schemes.

17.2.1 Transform Coding

Transform coding and in particular DCT coding has enjoyed significant popular owing to its ability to effectively exploit high spatial correlations in image data. It has even proven to be effective for coding of motion-compensated prediction error data which typically have fairly low correlations. All major image and video coding standards use DCT transforms of 8×8 block size. Besides DCT, other orthogonal block transforms have also been experimented with but from a practical standpoint significant benefits of other transforms over DCT have not yet been proven. From a coding efficiency standpoint, on the average, a block size of 8×8 has been proven to provide a suitable tradeoff in exploiting correlations and providing

adaptations needed owing to local structures in images, and is thus commonly used in DCT coding.

Improvements in DCT coding have been reported by use of somewhat larger size transforms, which from the implementation standpoint are becoming feasible only now. Overall, to be sufficiently adaptive across images of different resolutions, variable block size (VBS) schemes[22] may be promising. Further, optimized VBS coding may involve a hybrid of DCT and pixel domain coding. By pixel domain coding, we mean schemes such as Vector Quantization (VQ) in which an approximation to the block being coded is obtained by look-up into a limited codebook, or by template coding (similar to VQ) but with a codebook satisfying specific properties. As an example of a VBS scheme, block sizes of 16 × 16, 8 × 8, and 4 × 4 may be used for DCT coding and block size of 4 × 4 may be used for pixel-domain template coding. The extent to which VBS coding can be successful depends on minimization of overhead, good codebook design, and use of optimized VLCs for different block sizes. Other directions for improvements have focused on adaptation of block DCT to coding of irregular shaped regions.

Some improvements have also been reported with VQ of DCT coefficients at the expense of loss of genericity over a wide range of bitrates. Before motion-compensated interframe coding became popular, 3D-DCT coding was also tried for coding of video but such attempts were not very successful. Recently, 3D-DCT of 8 × 8 × 4 block size has been proposed for motion-compensated prediction error coding for B-pictures.[16]

17.2.2 Wavelet Coding

Sub-band coding decomposes an image signal into its frequency components similar to transform coding but also manages to retain spatial relationships (present in pixel domain representation) in its frequency components. Sub-band coding achieves critical sampling, unlike pyramid coding in the pixel domain, which requires oversampling. Sub-band encoding typically consists of applying a series of analysis filters over the entire image, downsampling of each band by a factor, quantization, and entropy coding. Sub-band decoding consists of complementary operations of entropy decoding, inverse quantization, upsampling of each band, and applying synthesis filters to generate the reconstructed image. The analysis and synthesis filters are generally chosen to allow perfect reconstruction. In general, sub-band coding can be viewed as transform coding with overlapped blocks so that it can exploit correlation over a larger set of pixels. While transform coding has the potential of producing blocking artifacts, sub-band coding may typically produce ringing or contouring artifacts. An advantage of sub-band coding is in terms of the flexibility it offers to allocate different rates to different sub-bands.

Typically, wavelet transform coding[15,18] has been treated as sub-band coding because sub-band analysis filtering and the transform operation are essentially equivalent and wavelet transform can be achieved with the sub-band filter bank. Without going into detail, the wavelet transform operation can be identified by properties such as shorter basis for high frequencies, higher resolution for high frequencies, orthogonality between basis, and multiresolution representation. Despite its relationship to sub-band decomposition, there is some benefit to treating wavelet coding as transform coding instead of subband coding owing to nonstationarity properties of images; these properties can be best exploited by locally adaptive quantization, scanning, and use of end of block code, such as often done in transform coding.

It has often been reported that wavelet coding can provide intraframe image coding with significantly higher performance[25], however, it is not completely clear as to how it can provide improvements in video coding. If simple intraframe wavelet techniques are applied for video coding, it is difficult to obtain sufficient performance. To improve performance, several ways of performing interframe coding with wavelets are being investigated by researchers and include coding of differences in wavelet transforms of current image and a previous image, wavelet coding over the entire image of block motion-compensated error signal generated by use of overlapped block motion compensation, and wavelet coding adapted to each region where motion compensation is performed

on a region basis. Among the more promising techniques are those that employ overlapped block motion compensation and wavelet coding of the entire image, followed by arranging of coefficients into a number of blocks that can be quantized, scanned, and VLC (or arithmetic) coded much like transform coding. Some improvements have also been reported by vector quantization of wavelet coefficients instead of scalar quantization and scanning.

17.2.3 Fractal Coding

Fractal coding[15,17] exploits the property of self-similarities within an image; some of the self-similarities may exist at different resolution scales and may not be directly obvious. Fractal coding basically uses concepts such as affine transformations, iterated function systems, and the collage theorem. Affine transformations consist of mappings such as translation, rotation, and scaling that allows a part of source image to yield a target image. An iterated function system is a collection of contractive affine maps that when applied iteratively on a part of source image can generate a part of the target image using repeated parts (such as translations, rotations, and scales) of a source image. The collage theorem says that if an image can be described by a set of contractive affine transformations then these transformations specify an iterated function system that can be used to reconstruct an approximation after some number of iterations and that it will eventually converge.

Fractal coding and VQ seem to have distinct similarities owing to use of some codebooks and need for matching operations. Like VQ and transform coding practical fractal coding is also block based. In fractal transform coding, first an image is partitioned into nonoverlapping smaller blocks called domain blocks that cover the entire image. Next, range blocks are defined and are allowed to overlap and not necessarily cover the entire image. Affine transformations are then computed which allow mapping of range blocks to domain blocks. The compressed representation then involves affine transformation maps, domain bock geometry, and range block addresses. Fractal decoding then consists of generating two buffers, a domain buffer and a range buffer. Affine transforms are applied using the range buffer and filling the domain buffer. The range and domain block are identified in the opposite buffers and affine transforms are applied again; this process is repeated until differences in the two buffers are small and results in an image that is a close approximation to the original image.

Again, fractal coding is promising for coding of intraframes but has obvious difficulties in coding of video. Intraframe coding of video using fractals does not provide the desired compression. Fractal coding of difference images does not work well owing to the pulsive nature of the differential or motion-compensated differential signal. Fractal coding of video by using self-similarities in 3D blocks has also been investigated. Currently research is in progress to determine ways in which motion compensation and fractal coding can be combined to provide an effective solution for video coding.

17.2.4 Region-Object-Based Coding

Images and video of natural or artificial scenes are generally quite nonstationary. This is due to the presence of local structures (edges) that correspond to imaged objects in scenes. Earlier we discussed a classification of images in which second generation coding was meant to represent contours and textures in a scene. It has been believed for quite some time that to achieve very high compression, not only information theoretic (such as correlations) but also structural properties (such as organization of structures, edges, etc.) need to be exploited. Region-based coding can be derived as a generalization of contour/texture coding, where regions can be of arbitrary shapes. Consider a scene in which each region can be identified; then the image coding problem reduces to finding an efficient representation of region boundaries (shapes) and that for texture values (luminance and chrominance intensities) to fill in for each region. This can also be looked upon as generalization of block transform coding; however, in that case shape information is not needed since block sizes and locations are known a priori; the transform

coding provides a means to specify texture information.

One of the main challenge in region-based coding[17,19] is to derive a few meaningful homogeneous regions in a scene; this is referred to as the segmentation problem. It is always possible to perform edge detection and try to define regions; however, this may result in too many regions which may cost considerable coding overhead. Ideally, one would like to segment a scene into a few principal objects that have real-world significance (semantic meaning). Each object could then be further subdivided into as many homogeneous regions as may provide overall good coding quality. However, automatic segmentation of scenes, except for the most trivial scenes, is a rather difficult problem for which considerable effort has been put forth by the computer vision community.

Next, for regions/objects that have been located, an efficient way of specifying shapes is necessary, since accurate shape representation causes a significant overhead, in effect reducing the number of bits available for texture coding. This is even more critical for low-bitrate video coding, in which the number of coding bitrates is limited. One of the simplest ways of representing shapes is via coding of chains of pixels, such that at each pixel a decision is made from among eight (or four) possible directions; with the use of differential coding and variable length coding chain coding can be made effective. Another method of shape coding is based on the use of polygon approximations[20] to actual shape via straight lines; this method is usually effective when some errors in shapes can be tolerated and does provide an overhead-efficient method for shape representation. Yet another method of shape representation is to use variable block size type segmentation using quadtree structures; this method also allows control of accuracy of shape representation to trade off shape coding overhead with texture coding overhead.

Finally, the texture for each region needs to be coded. For region-based texture coding, pixel domain coding (average values of region), transform coding, or other techniques such as wavelets can be employed. Recently, solutions for transform coding of arbitrary shaped regions have been proposed in the form of shape-adaptive DCT (SA-DCT), projection onto a convex set and DCT (POCS-DCT), and other schemes have been proposed. Attempts have also been made to jointly solve shape and texture coding problems by designing the texture coding scheme to implicitly specify shapes. Yet another direction has been to use block transform coding in conjunction with chroma keying to implicitly represent shapes, transferring some shape overhead into texture coding. Many of the aforementioned techniques can be applied for intraframe or motion-compensated interframe coding. In addition to texture coding improvements, attempts are also being made to reduce motion-compensated prediction error signal by use deformable triangles or quadrilaterals to perform advanced motion compensation. Such schemes can be used in conjunction with normal or region-based coding.

17.2.5 Model-Based Coding

By model-based coding, we primarily mean 3D model based coding.[21,26] Schemes that use 2D models were already discussed within the context of region-based coding. One of the main applications scenario in which research is ongoing in 3D model based coding is for representing the human head. Some advancements have also been made to extend this to represent a head and shoulders view as would be typical in a videophone scene. A generic wireframe model is used to represent the human head and provide the structure for key facial features.

A model-based coder then consists of three main components: a 3D head (facial) model, an encoder, and a decoder. The encoder segments an object of interest (a person) from the background, estimates motion of the object (translation, rotation, etc.), and then transmits these analysis parameters to the receiver. Further, as new previously unseen areas are uncovered at the encoder, it updates and corrects the model. This generates output images by using the 3D facial model and the analysis parameters. To model a person's face in sufficient detail, a deformable wireframe model consisting of several hundred triangles can be used. This model is adapted to a person's face by use of feature point positions. Then the front view

of the person's face is mapped onto the wireframe model. Now this model can be manipulated and the resulting images appear natural. Research is ongoing to improve the accuracy of model by use of side views, modeling of muscles, and so forth. To regenerate natural looking images by animating this model, accurate synthesis of facial movements is necessary. Several methods to accomplish this exist, such as transmitting of fine details and pasting them, use of deformation rules to control vertices of the model by identifying the facial actions, and translating them into movement of a set of action units.

The problem of analysis is a lot more difficult as compared to the problem of synthesis and deserves separate mention. At present there is no system that can automatically work for modeling of objects and extraction of model parameters, even for head and shoulder images. Analysis actually includes a number of subproblems such as determining feature points, segmentation of objects, estimation of global and local motion, and so forth. A number of solutions are being investigated to allow better analysis by tracking marks plotted on the moving object and using that to compute global (head) motion as well as a number of local movements corresponding to important features. Researchers[27–32] are looking into ways by which additional information such as range data, two or more views, speech, facial muscle models etc can improve the overall performance and robustness of 3D model based video coding system.

17.3 INTRODUCTION TO MPEG-4

The MPEG-4 standard is the next generation MPEG standard which was started in 1993 and is currently in progress. It was originally intended for coding of audio–visual information with very high compression at very low bitrates of 64 kbits/s or under. When MPEG-4 video was started, it was anticipated that with continuing advances in advanced (non-block-based) coding schemes, for example, in region-based and model-based coding, a scheme capable of achieving very high compression, emerge. By mid 1994, two things became clear. First, video coding schemes that were likely to be mature within the timeframe of MPEG were likely to offer only moderate increase in compression (say, up to 2 at the most) over existing methods as compared to the original goal of MPEG-4. Second, a new class of multimedia applications was emerging that required greater levels of functionality than that provided by any other video standard at bitrates in the range of 10 kbits/s to 1024 kbits/s. This led to broadening of the original scope of MPEG-4 to larger range of bitrates and important new functionalities.[3]

17.3.1 Focus of MPEG-4

The mission and the focus statements of MPEG-4 explain the trends leading up to MPEG-4 and what can be expected in the future and are directly stated here from the MPEG-4 Proposal Package Description (PPD) document.[1]

The traditional boundaries between the telecommunications, computer, and TV/film industries are blurring. Video, sound, and communications are being added to computers; interactivity is being added to television; and video and interactivity are being added to telecommunications. What seems to be convergence in reality is not. Each of the industries approach audio–visual applications from different technological perspectives, with each industry providing its own, often in compatible solutions for similar applications.

Three important trends can be identified:

- The trend toward wireless communications
- The trend toward interactive computer applications
- The trend toward integration of audio–visual data into a number of applications.

At the intersection of the traditionally separate industries these trends must be considered in combination; new expectations and requirements arise that are not adequately addressed by current or emerging standards.

Therein lies the focus of MPEG-4, the convergence of common applications of the three industries. MPEG-4 will address these new expectations and requirements by providing audio–visual coding solutions to allow interactivity, high compression, or universal accessibility.

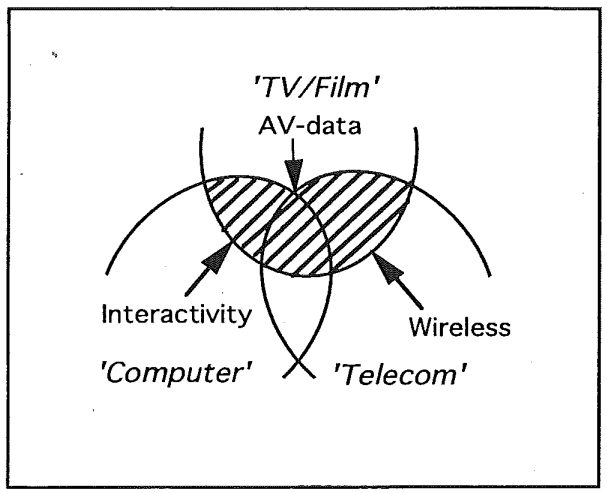


Fig. 17.13 Applications area addressed by MPEG-4 (shaded region).

Figure 17.13 shows the applications of interest to MPEG-4 arising at the intersection of different industries.

17.3.2 MPEG-4 Applications

We now provide a few concrete examples of applications or application classes that the MPEG-4 video standard may be most suitable for:

- Video on LANs, Internet video
- Wireless video
- Video databases
- Interactive home shopping
- Video e-mail, home movies
- Virtual reality games, flight simulation, multi-viewpoint training.

17.3.3 Functionality Classes

Now that we have some idea of the type of applications MPEG-4 is aimed for we clarify the three basic functionality classes[1] that the MPEG-4 standard is addressing. They are as follows:

- *Content-Based Interactivity* allows the ability to interact with important objects in a scene. Currently such interaction is possible only for synthetic objects; extending such interaction to natural and hybrid synthetic/natural objects is important to enable new audio–visual applications.
- *High Compression* is needed to allow increase in efficiency in transmission or the amount of storage required. For low-bitrate applications, high compression is very important to enable new applications.
- *Universal Accessibility* means the ability to access audio–visual data over a diverse range of storage and transmission media. Owing to an increasing trend toward mobile communications, it is important that access be available to applications via wireless networks. This acceptable performance is needed over error-prone environments and at low bitrates.

17.3.4 MPEG-4 Work Organization

Currently the MPEG-4 work is subdivided into five parts:

- Video
- Audio
- Integration
  —Requirements
  —MPEG-4 Syntax Description Language (MSDL)
  —Synthetic and Natural Hybrid Coding (SNHC)
- Implementation studies
- Test.

Each part is handled by a separate subgroup, so in fact there are five parts. Further, some parts are divided into a number of subparts; for example, Integration work consists of Requirements, MSDL, and SNHC. Each subgroup, depending on the number of technical issues that need to be addressed, forms a number of ad hoc groups, which by definition are temporary in nature to ensure maximum flexibility.

17.4 MPEG-4 REQUIREMENTS

The process of collection of requirements for MPEG-4, although it was started in late 1993, is continuing at present,[8] in parallel with other work items. This is so because the process of development of an MPEG standard is intricate, tedious, thorough, and thus time intensive (about 3 to 5 years per standard with overlap between stan-

dards). To keep up with marketplace needs for practical standards at the right time and follow evolving trends, the requirements collection process is kept flexible. This ensures the ability to modify requirements as they evolve over time as well as to be able to add new ones when absolutely necessary. The work for various parts (or subparts) of MPEG-4, Video, Audio, MSDL, and SNHC is kept synchronized with the requirements collected at any given point in time, to the maximum extent possible. The major restructuring of the MPEG-4 effort in July 1994 to expand its scope to a number of useful functionalities and not to be limited to very low bitrates only was in part due to an exhaustive effort in requirements analysis which indicated changing marketplace and new trends.

Before we specify the new functionalities of MPEG-4, the standard functionalities it must support, and some general requirements, we need to clarify the terminology used in MPEG-4. Some of the terms used have evolved from MPEG-1 and MPEG-2 while other terms are new.

17.4.1 Terminology

Tool: A tool is a technique that is accessible or described by the MSDL. Tools may themselves consist of other tools. Motion compensation, transform filters, audio–visual synchronization, and so forth are some examples of tools.

Algorithm: An algorithm is an organized collection of tools that provides one or more functionalities. An algorithm may be composed of a number of tools tools, or even of other algorithms. DCT image coding, Code Excited Linear Prediction, Speech-Driven Image Coding, and so forth are some examples of algorithms.

Flexibility: Flexibility is the degree of programmability provided in an MPEG-4 implementation. Three degrees of flexibility can be defined: fixed set of tools and algorithms (Flex_0); configurable but fixed tools (Flex_1); and downloadable tools, algorithms, and configuration (Flex_2).

Profile: A profile addresses a cluster of functionalities that can enable one or more applications. At each flexibility, one or more profiles can be defined. Thus, at Flex_0, a profile is a set of tools configured into an algorithm; at Flex_1, a profile is a set of tools; and at Flex_2, a profile enables the ability to download and configure tools and algorithms. MPEG-1 audio layer 3, MPEG-2 Video Profile, MPEG-2 System, and H.263 are all examples of profiles at Flex_0.

Level: A level is a specification of the constraints and performance criteria needed to satisfy one or more applications. Each profile is classified by a degree of flexibility and a value of level. At Flex_0, for a profile, a level simply indicates performance achieved. At Flex_1, for a profile profile, a level is the capability of a system to configure a set of tools and execute the resulting algorithms. At Flex_2, for a profile, a level specifies system capability relative to downloading, configuration of tools, and execution of resulting algorithms.

Conformance Points: Conformance points are specifications of particular Flexibility/Profile/Level combinations at which conformance may be tested. Like MPEG-2, MPEG-4 expects to standardize a set of conformance points.

17.4.2 MPEG-4 Functionalities

Earlier, we had discussed the various classes of functionalities being addressed by MPEG-4. In Table 17.1 we show a detailed list of eight key functionalities[1,2,8] and how they are clustered into three functionality classes. These functionalities are to be supported through definition of a set of coding tools and a mechanism (MSDL) to access, download, combine, and execute these coding tools.

17.4.3 Basic Functionalities

Besides the new functionalities that the ongoing work in MPEG-4 expects to support, a set of basic functionalities are also required and can be listed as follows.

Synchronization: The ability to synchronize audio, video, and other content for presentation.

Auxillary data capability: The ability to allocate channels for auxillary data streams.

Virtual channel allocation: The ability to dynamically partition video, audio, and system channels.

Table 17.1 Functionalities Supported by MPEG-4 and Their Explanation

Content Based Interactivity
Content-Based Multimedia Data Access Tools: The access of multimedia data with tools such as indexing, hyperlinking, browsing, uploading, downloading, etc.
Content-Based Manipulation and Bitstream Editing: The ability to provide manipulation of contents and editing of audio–visual bitstreams without the requirement for transcoding.
Hybrid Natural and Synthetic Data Coding: The ability to code and manipulate natural and synthetic objects in a scene including decoder controllable methods of compositing of synthetic data with ordinary video and audio, allowing for interactivity.
Improved Temporal Random Access: The ability to efficiently access randomly in a limited time and with fine resolution parts (frames or objects) within an audio–visual sequence. This also includes the requirement for conventional random access.
Compression
Improved Coding Efficiency: The ability to provide subjectively better audio–visual quality at bitrates compared to existing or emerging video coding standards.
Coding of Multiple Concurrent Data Streams: The ability to code multiple views/soundtracks of a scene efficiently and to provide sufficient synchronization between the elementary streams. To obtain high coding efficiency techniques for exploiting redundancies that exist between multiviewpoint scenes are required.
Universal Access
Robustness in Error-Prone Environments: The capability to allow robust access to applications over a variety of wireless and wired networks and storage media. Sufficient robustness is required, especially for low-bitrate applications under severe error conditions.
Content-Based Scalability: The ability to achieve scalability with fine granularity in spatial, temporal, or amplitude resolution; quality; or complexity. Content-based scaling of audio–visual information requires these scalabilities.

Low delay mode: The ability for system, audio, and video codecs to operate with low delay.

User controls: The ability to support user control for interactive operation.

Transmission media adaptibility: The ability to operate in a number of different media.

Interoperability with other systems: The ability to interoperate with other audio–visual systems.

Security: The ability to provide encryption, authentication, and key management.

Multipoint capability: The ability to have multiple sources or destinations.

Coding of audio types: The ability to deal with a range of audio and speech data such as wideband, narrowband, intelligible, synthetic speech, and synthetic audio.

Quality: The ability to provide audio–visual services at sufficient quality level.

Bitrates: The ability to efficiently operate over range of bitrates from 9.6 to 1024 kbits/s.

Low complexity mode: The ability to operate with low complexity (in hardware, software, or firmware).

17.4.4 Evolving Requirements

The various aforementioned advanced as well basic functionalities give rise to certain requirements that an MPEG-4 coding system must satisfy. Further, it is not surprising that several functionalities sometimes have common requirements and often, one or more such functionalities is simultaneously needed in an application. One way to derive specific requirements is to choose a number of representative applications and provide numerical bounds or characteristics for every single parameter; however, such an exercise is feasible for applications that are already in use and are well understood enough to allow full characterization. Another solution is to look for functional commonalities between applications to generate several clusters and specify parameter ranges for these clusters. Generating requirements for MPEG-4 is an ongoing exercise and is using a little bit of both approaches.

The requirements process is complicated further by the fact that requirements for MPEG-4 coding need to be translated into a set of requirements for Video, Audio, MSDL, and SNHC. The current approach is to specify requirements for Flexibility/Profile/Level combinations. In this manner, for each degree of flexibility, the requirements for one or more profiles are being described. Further, for each profile, requirements for one or more levels are also being specified.

17.5 MPEG-4 VIDEO

MPEG-4 video was originally intended for very high compression coding at bitrates of under 64 kbits/s. As mentioned earlier, in July 1994, the scope of MPEG-4 was modified to include a new set of functionalities, and as a consequence, it was no longer meaningful to restrict bitrates to 64 kbits/s. Consistent with the functionalities it intends to support, the MPEG-4 video effort is now aimed at very efficient coding optimized at (but not limited to) a bitrate range of 10 kbits/s to about 1 Mbit/s, while seeking a number of content-based and other functionalities.

Following the tradition of previous MPEG video standards, the MPEG-4 video effort is currently in its collaborative phase after undergoing a competitive phase. The competitive phase consisted of issuing an open call for proposals in November 1994 to invite candidate schemes for testing in October/November 1995. A proposal package description (PPD) was developed describing the focus of MPEG-4, the functionalities being addressed, general applications MPEG-4 was aimed at, the expected workplan, planned phases of testing, how Verification Models (VMs) would be used, and the time iternary of MPEG-4 development. The MPEG-4 PPD which was started in November 1994 underwent successive refinements until July 1995. In parallel to the PPD development effort, a document describing the MPEG-4 Test/Evaluation Procedures started in March 1995 underwent several iterations and was subsequently finalized by July 1995.

17.5.1 Test Conditions for First Evaluation

For the first evaluation tests,[2] proposers were required to submit algorithm proposals for formal subjective testing or tools proposals. Since not all functionalities were tested in the first evaluation, for the untested functionalities, tools submissions were invited. It turned out that some tools submissions were made by proposers who were unable to complete an entire coding algorithm because of limited time or resources. In a few cases, proposers also used tools submissions as an opportunity to identify and separately submit the most promising components of their coding proposals. Since tools were not formally tested they were evaluated by a panel of experts and this process was referred to as *evaluation by experts.*

The framework of the first evaluation involved standardizing test material to be used in the first evaluation. Toward that end, video scenes are classified from relatively simple to more complex by categorizing them into three classes: Class A, Class B, and Class C. Two other classes of scenes, Class D and Class E, were defined; Class D contained stereoscopic video scenes and Class E contained hybrids of natural and synthetic scenes.

Since MPEG-4 is addressing many types of functionalities and different classes of scenes, it was found necessary to devise three types of test methods. The first type of test method was called the Single Stimulus (SS) Method and involved rating the quality of coded scene on an 11-point scale from 0 to 10. The second type of test method was called the Double Stimulus Impairment Scale (DSIS) and involved presenting to assessors a reference scene (coded by a known standard) and after a 2-second gap, a scene coded by a candidate algorithm, with impairment of the candidate algorithm compared to reference using a five-level impairment scale. The third test method was called the Double Stimulus Continuous Quality Scale (DSCQS) and involved presenting two sequences with a gap of 2 seconds in between. One of the two sequences was coded by the reference and the other was coded by the candidate algorithm; however, the assessors were not informed of the order in which the reference scene

Table 17.2 List of MPEG-4 First Evaluation Formal Tests and Their Explanation

Compression
Class A Sequences at 10, 24, and 48 kbits/s: Coding to achieve the highest compression efficiency. Input video resolution is CCIR-601 and although any spatial and temporal resolution can be used for coding, the display format is CIF on a windowed display. The test method employed is SS.
Class B sequences at 24, 48, and 112 kbits/s: Coding to achieve the highest compression efficiency. The input video resolution is CCIR-601 and although any combination of spatial and temporal resolutions can be used for coding, the display format is CIF on a windowed display. The test method employed is SSM.
Class C Sequences at 320, 512, and 1024 kbits/s: Coding to achieve the highest compression efficiency. The input video resolution is CCIR-601 and although any combination of spatial and temporal resolution can be used for coding, the display format is CCIR-601 on a full display. The test method employed is DSCQS.
Scalability
Object Scalability at 48 kbit/s for Class A, 320 kbit/s for Class E, and 1024 kbits/s for Class B/C Sequences: Coding to permit dropping of specified objects resulting in remaining scene at lower then total bitrate; each object and the remaining scene is evaluated separately by experts. The display format for Class A is CIF on a windowed display and for Class B/C and Class E is CCIR-601 on a full display. The test method employed for Class A is SSM, for Class B/C DSCQS, and for Class E DSIS.
Spatial Scalability at 48 kbit/s for Class A, and 1024 kbits/s for Class B/C/E Sequences: Coding of a scene as two spatial layers with each layer using half of the total bitrate; however, full flexibility in choice of spatial resolution of objects in each layer is allowed. The display format for Class A is CIF on a windowed display and that for Class B/C/E is CCIR-601 on a full display. The test method employed for Class A is SSM, and that for Class B/C/E is DSCQS.
Temporal Scalability at 48 kbits/s for Class A, and 1024 kbits/s for Class B/C/E Sequences: Coding of a scene as two temporal layers with each layer using half of the total bitrate; however, full flexibility in choice of temporal resolution of objects in each layer is allowed. The display format for Class A is CIF on a windowed display and that for Class B/C/E is CCIR-601 on a full display.
Error Robustness
Error Resilience at 24 kbits/s for Class A, 48 kbits/s for Class B, and 512 kbits/s for Class C: Test with high random bit error rate (BER) of 10^{-3}, multiple burst errors with three bursts of errors with 50% BER within a burst, and a combination of high random bit errors and multiple burst errors. The display format for Class A and Class B sequences is CIF on a windowed display and for Class C sequences it is CCIR-601 on full display. The test method employed for Class A and Class B is SSM and that for Class C DSCQS.
Error Recovery at 24 kbits/s for Class A, 48 kbits/s for Class B, and 512 kbits/s for Class C: Test with long burst errors of 50% BER within a burst and a burst length of 1 to 2 seconds. Display format for Class A and Class B is CIF on a windowed display and for Class C it is CCIR-601 on full display. The test method employed for Class A and Class B is SSM and that for Class C DSCQS.

and the coded scene under test were presented to them. In the DSCQS method, a graphical continuous quality scale was used and was latter mapped to a discrete representation on a scale of 0 to 100.

17.5.2 First Evaluation and Results

Table 17.2 summarizes the list of informal tests and gives explanations of each test and the type of method employed for each test.

To ensure that the MPEG-4 video subjective testing process would provide a means of comparison of performance of new proposals to known standards it was decided to use existing standards as anchors. In tests involving Class A and B sequences, it was decided to use the H.263 standard (with TMN5 based encoding results generated by a volunteer organization) as the anchor. The anchors for Class A and B used test sequences downsampled for coding and upsampled for display using filters that were standardized.

Likewise, in tests involving Class C sequences, it was decided to use the MPEG-1 standard (with encoding results generated by a volunteer organization) as the anchor. The anchor for Class C, however, used test sequences downsampled for coding and upsampled for display using proprietary nonstandardized filters. To facilitate scalability tests (object scalability, spatial scalability, and

temporal scalability), standardized segmentation masks were generated so that all proposers could use the same segmentation, with the expectation of making the comparison of results easier.

The proposers were allowed to address one or more of the tests listed in the table. By the registration deadline of mid September 1995, more than 34 proposers had registered. By the end of the first week of October 1995, for the proposals being subjectively tested, proposers submitted D1 tapes with their results, the description of their proposals, coded bitstreams, and executable code of their software decoders. By the third week of October 1995, proposers of tools had also submitted demonstration D1 tapes and a description of their tool submissions. About 40 video tools submissions were received. The formal subjective tests of algorithm proposals as well as an informal evaluation by experts of tools were started by the end of October 1995. The results of the individual tests[23] and a thorough analysis of the trends were made available during the November MPEG meeting.

The results of compression tests in various categories revealed that the anchors performed quite well, in many cases among the top three or four proposals. In some cases, the differences between many top performing proposals were found to be statistically insignificant. In some categories there were, however, innovative proposals that beat the anchors or provided as good a picture quality as the anchors while providing additional (untested) functionalities. It was also found that since spatial and temporal resolutions were not prefixed, there was some difficulty in comparing subjective and objective (SNR) results, owing to differences in the choices made by each proposer. It seemed that subjective viewers had preferred very low temporal resolutions as long as the spatial quality looked good, over a more balanced tradeoff of spatial quality and temporal resolution. Besides compression, the proposers had also managed to successfully achieve other functionalities tested.

Besides the results from subjective tests, the tools evaluation experts presented their results,[24] about 16 tools were judged as promising for further study.

17.5.3 Core Experiments Formulation

In the period from November 1995 through January 1996, the process of definition of core experiments was initiated. A total of 36 core experiments were defined prior to January 96 MPEG meeting;[7] another five experiments were added at the meeting itself, bringing the total number of experiments to about 41. Since the total number of experiments is rather large, these experiments are classified into a number of topics:

1. Prediction
2. Frame Texture Coding
3. Quantization and Rate Control
4. Shape and alpha Channel Coding
5. Object/Region Texture Coding
6. Error Resilience and Error Correction
7. Bandwidth and Complexity Scalability
8. Multiview and Model Manipulation
9. Pre-, Mid-, and Postprocessing

A set of common experimental conditions was also agreed to. A number of ad hoc groups were initiated each focusing on one or more topics as follows:

- Coding Efficiency—topics, 1, 2, and 3
- Content-Based Coding—topics 4 and 5
- Robust Coding—topic 6
- Multifunctional Coding—topics 7, 8, and 9.

Between January 1996 and March 1996 MPEG meetings these ad hoc groups undertook the responsibility of producing an improved description of each core experiment consistent with the VM 1.0, seeking volunteer organizations for performing core experiments and finalizing experiment conditions. In the meantime, a few new experiments were also proposed. At the MPEG meeting in March 1996, the ad hoc group on content-based coding was subdivided into two groups. At the same meeting, there were also minor changes in rearrangement of experiments and an addition of three or four experiments.

17.5.4 Verification Model (VM)

It was agreed that MPEG-4 only have a single verification model for development of the standard to address the various functionalities.[6] The first MPEG-4 Video VM (VM 1.0) was released on January 24, 1996. It supports the following features:

- VOP structure
- Motion/Texture coding derived from H.263
- Separate Motion/Texture syntax as an alternative for error resilience
- Binary and Grayscale Shape Coding
- Padding.

The second MPEG-4 Video VM (VM 2.0)[9] was released on March 29, 1996. It refines some existing features of VM 1.0 and adds a few new features as follows:

- B-VOPs derived from H.263 B-pictures and MPEG-1/2 B-pictures
- DC coefficients prediction for Intra-Macroblocks as per MPEG-1/2
- Extended Motion Vector Range
- Quantization Visibility Matrices as per MPEG-1/2

17.5.4.1 VM Description

We now describe the important elements of the latest MPEG-4 Video VM.[9] An input video sequence consists of a sequence of related pictures separated in time. Further, each picture can be considered as consisting of a set of flexible objects, that from one picture to the next picture undergo a variety of changes such as translations, rotations, scaling, brightness and color variations, and so forth. Moreover, new objects enter a scene and existing objects depart, leading to the presence of certain objects only in certain pictures. Sometimes, scene changes occur, and thus the entire scene may either get reorganized or initialized.

Many of MPEG-4 functionalities require access not only to individual pictures but also to regions or objects within a picture. On a very coarse level, access to individual objects can be thought of as generalization of the slice structure of MPEG-1/2 or that of the GOB structure of H.263, although such structures do not have a semantic meaning in these standards and although potentially there is a way of accessing them, they are not intended for individual access or display. MPEG-4, to allow access to contents of a picture, has introduced the concept of Video Object planes (VOPs).

A VOP can be a semantic object that is represented by texture variations (a set of luminance and chrominance values) and (explicit or implicit) shape information. In natural scenes, VOPs are obtained by semiautomatic or automatic segmentation, and the resulting shape information can be represented as a binary mask. On the other hand, for hybrid (of natural and synthetic) scenes generated by blue screen composition, shape information is represented by an 8-bit component.

In Fig. 17.14 we show the decomposition of a natural scene into a number of VOPs. The scene consists of two objects (head and shoulders view of a human, and a logo) and the background. The objects are segmented by semiautomatic or automatic means and are referred to as VOP1 and VOP2, while the background without these objects is referred to as VOP0. Each picture of interest is segmented into VOPs in this manner. The same object in a different picture is refered to by the VOP number assigned to it when it first appeared in the scene. The number of VOPs selected in a scene is dependent on the total available bitrate for coding, the degree of interactivity desired, and of course, the number and size of unique objects in the scene.

The VOPs are encoded separately and multiplexed to form a bitstream that users can access and manipulate (cut, paste, . .). The encoder sends, together with VOPs, information about scene composition to indicate where and when VOPs are to be displayed. This information is however optional and may be ignored at the decoder which may use user-specified information about composition.

In Fig. 17.15 we show a high-level structure of a VOP-based Codec. Its main components are VOP Formatter, VOP Encoders, Mux, Demux, and VOP Decoders and VOP Compositor. The VOP Formatter segments the input scene into VOPs and outputs formatted VOPs for encoding. Although separate encoders (and decoders) are shown for

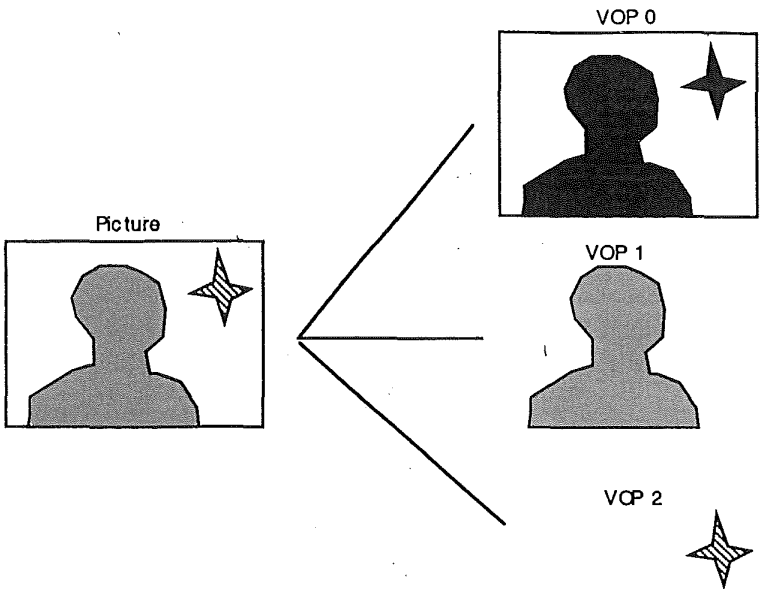


Fig. 17.14 Semantic segmentation of a picture into VOPs.

each VOP, this separation is mainly a logical one rather than a physical one. In principle even though different VOPs can use different methods of coding, it is anticipated that this will mainly be the case when VOPs contain different data types such as natural data, synthetic data, etc. Thus, although there may be many VOPs, there may be only a few (typically, one or two) coders. The multiplexer, Mux, multiplexes coded data from different VOPs into a single bitstream for transmission or storage; the demultiplexer, DeMux, performs the inverse operation. After decoding, individual VOPs are fed to a VOP compositor which either utilizes the composition information in the bitstream or composes VOPs under user control.

In Fig. 17.16 we show a block diagram structure of a VOP coder; this coder is meant for coding of natural video data. Its main components are: Texture Coder, Motion Coder, and Shape Coder.

The Texture Coder codes the luminance and chrominance variations of the region bounded by the VOP. This region may either be subdivided into macroblocks and blocks and coded by DCT coding, or other non-block-based techniques such as region-oriented DCT or wavelet coding may be used. Further, the coded signal may in fact be either an Intra- or motion-compensated Inter-frame signal. The Texture Coder is currently assumed to be a block-based DCT coder involving block DCT, quantizer, scan (and complementary operations of inverse

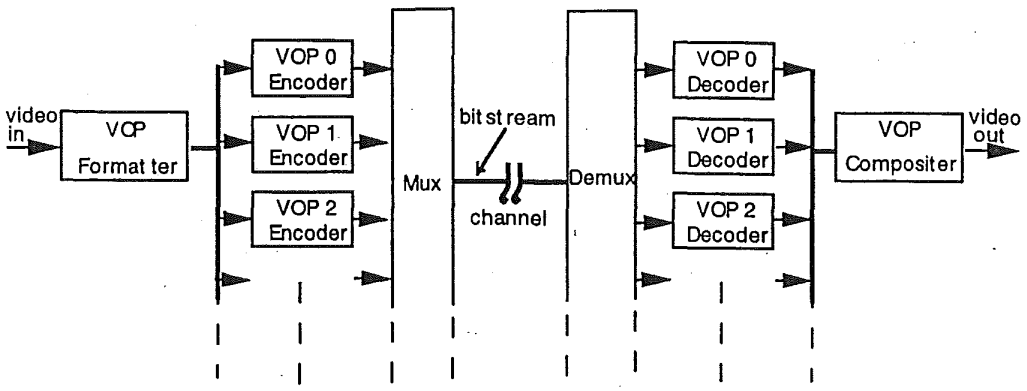


Fig. 17.15 Structure of VOP-based Codec for MPEG-4 Video.

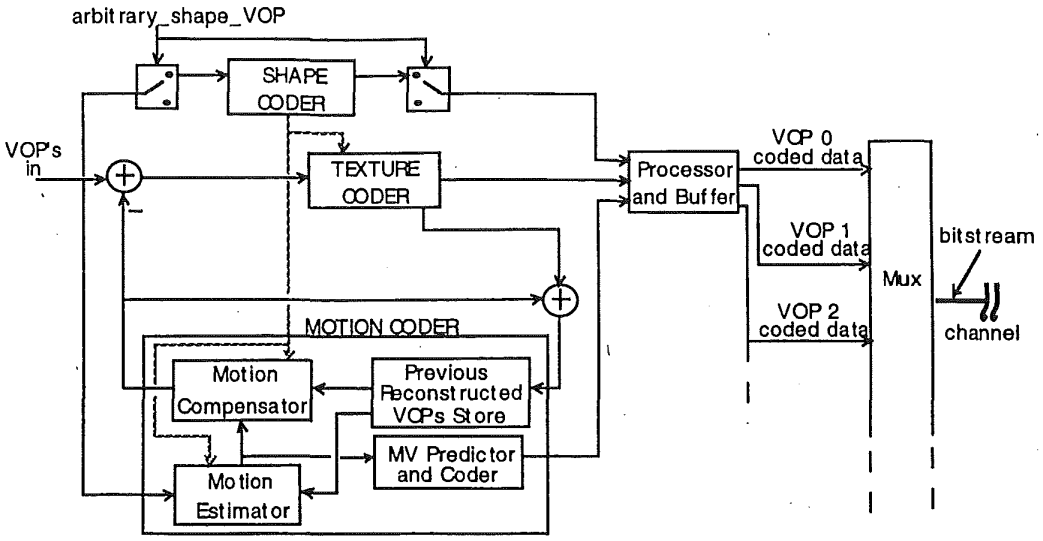


Fig. 17.16 Detailed structure of VOP Encoder.

DCT, inverse quantizer, and inverse scan) as per H.263, MPEG-1, and MPEG-2 standards. Since VOPs can be of irregular shape, padding of VOPs (see discussion of Motion Compensation) is needed.

The Motion Coder consists of a Motion Estimator, Motion Compensator, Previous VOPs Store, Motion Vector (MV) Predictor, and Coder. The Motion Estimator computes motion vectors using the current VOP and temporally previous reconstructed version of the same VOP available from the Previous Reconstructed VOPs Store. The Motion Compensator uses these motion vectors to compute motion-compensated prediction signal using the temporally previous reconstructed version of the same VOP (reference VOP). The MV Predictor and Coder generates prediction for the MV to be coded using H.263-like median prediction of MVs. Currently, the motion estimation and compensation in the VM is block based, basically quite similar to that in H.263. The current MPEG-4 VM employs not only default motion mode but also Unrestricted Motion Vector mode and Advanced Prediction modes of H.263; these modes are however not options. In the current VM, at the VOP borders, owing to irregular shape of VOPs, polygon matching instead of block matching is employed. This requires use of repetitive padding on the reference VOP for performing motion compensation and for texture coding when using block DCT.

In repetitive padding each pixel outside the object boundary is considered as zero pixel, each line is scanned horizontally for zero pixel segments, and if the zero pixels occur between nonzero pixel segments they are filled by the average of pixels at endpoints; otherwise they are filled by pixels at endpoints of the nonzero segment. Each line is then scanned vertically, repeating the filling operation as for horizontal scanning. If zero pixels can be filled by both horizontal and vertical scanning an average of the two values is used, and finally the remaining zero pixels are filled by the average of horizontally and vertically closest nonzero pixels.

While we are still on the subject of motion estimation and compensation, it is worth clarifying that the MPEG-4 VM now supports B-pictures. B-pictures currently in VM are derived by combining B-picture modes from H.263 and MPEG-1/2. Thus motion estimation and compensation also have to be performed for B-pictures. In the current MPEG-4 B-pictures, a choice of one out of four modes that differ mainly in the type of motion compensation is allowed on a macroblock basis. The motion compensation modes allowed are Direct, Forward, Backward, and Interpolated. The Direct mode is derived from H.263 and uses scaled motion vectors and a delta update vector. The Forward, Backward, and Interpolated are modes from MPEG-1/2. with one vector each in Foreward and Backward modes and two vectors in Interpolated mode.

The Shape Coder codes the shape information of a VOP and may involve a quadtree representation, a chain representation, a polygonal representation, or some other representation. If a VOP is of the size of a full picture, no shape information is sent. Thus, inclusion of the shape information is optional depending on the value of the binary signal, arbitary_shape_VOP, which controls the associated switches. The Shape Coder in the current version of the VM is assumed to be Quadtree based. Efforts are in progress to replace it by a Chain or a Polygon Approximation based technique.

The complete VOP Coder just described allows either combined coding of motion and texture data as in H.263, MPEG-1/2, or separate coding of motion and texture data for increased error resilience. The main difference is whether different types of data can be combined for entropy coding; this is not possible for separate motion and texture coding which assumes separate entropy coding to belong to individual Motion, Texture, and Shape coders rather than to a single entropy coder. Further, since in our discussion we have assumed VOPs derived from natural image or video data, all VOPs are coded by the same VOP Coder and the coded data of each VOP is output (by Processor and Buffer) for multiplexing to Mux which generates the bitstream for storage or transmission.

Earlier, although we mentioned the quadtree based shape coding method currently in the MPEG-4 VM, a few words of explanation are in order. The shape information may be either binary or grey scale and is correspondingly referred to as binary alpha plane or grey scale alpha plane. Currently in VM, binary alpha planes are coded with quadtree (without vector quantization) and grey scale alpha planes are coded by quadtree with vector quantization. An alpha plane is bounded by the tightest rectangle that includes the shape of a VOP. The bounding rectangle is extended on the right-bottom side to multiples of 16 × 16 samples, extended alpha samples set to zero, and the extended alpha plane is partitioned to blocks of 16 × 16 samples and encoding/decoding is done per 16 × 16 block. We now briefly explain binary alpha plane coding.

Figure 17.17 shows the quadtree structure employed for binary alpha plane coding. At the bottom level (level 3) of quadtree, a 16 × 16 alpha block is partitioned into 64 sub-blocks of 2 × 2 samples. Each higher level of quadtree is then formed by grouping 4 sub-blocks at the lower level as shown. The following sequence of steps is then employed.

1. Rounding process for bitrate control

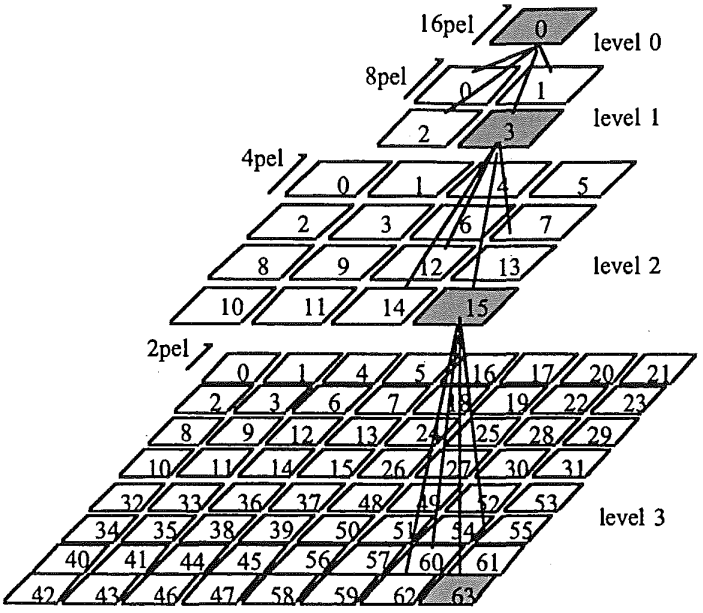


Fig. 17.17 Quadtree structure for binary shape (alpha plane) coding.

2. Indexing of sub-blocks at Level 3
3. Grouping process for higher levels
   - grouping of sub-blocks from Level 3 to Level 2
   - grouping of sub-blocks from Level 2 to Level 1
   - grouping of sub-blocks from Level 1 to Level 0
4. Encoding process.

The rounding process is needed only for lossy coding of binary alpha planes and can be ignored for loss-less coding.

The indexing of sub-blocks at level 3 consists of assigning an index to each 2 × 2 sub-block by first performing a swapping operation on alpha pixels of each sub-block, comparing their values against values of upper-block alpha pixels and left-block alpha pixels, except for sub-blocks belonging to the uppermost row of 2 × 2 sub-blocks or left-most column of 2 × 2 sub-blocks in Level 3 block comprised of a 8 × 8 array of sub-blocks (each of size 2 × 2). After swapping, a numerical index is calculated for the 2 × 2 sub-block as follows:

$$\text{index} = 27 \times b[0] + 9 \times b[1] + 3 \times b[2] + b[3]$$

where b[i] = 2 if the sample value = 255; b[i] = 0 if the sample value = 0.

The current sub-block is then reconstructed and used as a reference when processing subsequent sub-blocks.

The grouping process of blocks at the higher level first starts at Level 2 where four sub-blocks from Level 3 are grouped to form a new sub-block. The grouping process involves swapping and indexing similar to that discussed for Level 3. The current sub-block is then reconstructed and used as a reference when processing subsequent sub-blocks. At the decoder swapping is done following a reverse sequence of steps as at the encoder. The grouping process is also performed similarly for level 1 where four sub-blocks from Level 2 are grouped to form a new sub-block. The swapping, indexing, and reconstruction of sub-block follows grouping, the same as that for other levels.

The encoding process involves use of results from the grouping process which produces a total of 85 (= 1 = 4 + 16 + 64) indices for a 16 × 16 alpha block. Each index is encoded from topmost level (Level 0). At each level, the order for encoding and transmission of indices is shown by numbers in Fig. 17.17. Indices are encoded by variable length codes.

As a final note on the current Video VM, it now supports a larger range of motion vectors (H.263 range was too limited even when using the Unrestricted Motion Vector mode), MPEG-1/2 style quantization visibility matrices for Intra- and Inter-macroblocks, MPEG-1/2 style DC prediction for Intra-macroblocks, and of course, B-VOPs. The MPEG-4 Video VM is currently undergoing an iterative process which is eventually expected to lead to convergence as core experiments get resolved and the successful techniques get adopted.

17.5.5 Video Directions

From our discussion of the MPEG-4 Video VM development process, it may appear that the goal of VM is solely to address all functionalities via a single coding algorithm, the various parts of which are selected based on best results of core experiments. In reality, the goal of MPEG-4 is to select a set of coding tools (and thus the resulting algorithms) to address the specified functionalities, in the best possible way. Tools are expected to be selected to minimize redundancy in problems they solve; however, at times a tool may be selected if it significantly outperforms another tool, which was originally selected for solving a different problem.

Although it is a bit difficult to completely predict which tools will eventually be selected in the MPEG-4 Video standard, the trends are quite clear. Block motion-compensation and DCT-based schemes still remain fairly competitive and in many cases outperform any other alternative schemes. The continuing core experiments will most likely be able to achieve up to about 40% to 50% improvement over existing standards in specific bitrate ranges of interest. A number of useful tools on content-based scalability, error-resilient coding, synthetic model based coding, multiviewpoint coding, and others will offer functionalities not offered by any other standard in an integrated manner.

The MPEG-4 Video effort is expected to contribute to tools and algorithms for part 2 of the MPEG-4 Working Draft-1 scheduled for November 1996. After the first revision, Working Draft-2

is scheduled for release in March 1997, and after the second revision, Working Draft Version-3 is scheduled for release in July 1997; both these revisions will involve refinements in tools and algorithms contributed to part 2. Also, in July 1997, the final MPEG-4 Video VM will undergo Verification Tests for its validation; any new candidate proposals at that time are also allowed to participate in those tests. The MPEG-4 effort (MSDL, Video, and Audio) is expected to lead to a stable stage of Committee Draft by November 1997 and be finally approved as the International Standard in November 1998.

17.6 MPEG-4 AUDIO

The MPEG-4 Audio coding effort is also in progress in parallel with the MPEG-2 NBC Audio coding effort which is now reaching a mature stage. The MPEG-4 Audio effort is targeting bitrates of 64 kbits/s or under. Besides coding efficiency, content-based coding of audio objects, and scalability are also being investigated.

The MPEG-4 Audio effort also underwent subjective testing recently. Three classes of test sequences—Class A, B, and C—were identified. Class A sequences were single-source sequences consisting of a clean recording of a solo instrument. Class B sequences were single-source with background sequences consisting of a person speaking with background noise. Class C sequences were complex sequences consisting of an orchestral recording. All sequences were originally sampled at 48 kHz with 16 bits/sample and were monophonic in nature. For generating reference formats filters were specified to downsample them to 24, 16, and 8 kHz. A number of bitrates such as 2, 6, 16, 24, 40, and 64 kbits/s were selected for testing of audio/speech. The first three bitrates are obviously suitable only for speech material. The audio test procedures used were as defined in ITU-R Recommendation 814.

The submissions to tests included variants of MPEG-2 NBC Audio coding, improvements on MPEG-1 coding, and new coding schemes. For specific bitrates, improvements over existing coding solutions were found. The collaborative work has been started and an initial MPEG-4 Audio VM has been developed. MPEG-4 Audio development is expected to undergo a core experiment process similar to the MPEG-4 Video development process. The MPEG-4 Audio effort is expected to contribute to tools and algorithms for part 3 of the MPEG-4 standard. The schedule of various versions of the Working Draft, the Committee Draft, and the International Standard is the same as that mentioned while discussing MPEG-4 Video.

17.7 MPEG-4 SYNTAX DESCRIPTION LANGUAGE (MSDL)

Owing to the diverse nature of the various functionalities to be supported by MPEG-4 and the flexibilities offered by software based processing, it was envisaged that the MPEG-4 standard would be basically different from traditional audio–visual coding standards such as MPEG-1, MPEG-2, or others. While the traditional standards are fairly fixed in the sense of the bitstream syntax, and the decoding process they support, the MPEG-4 standard expects to relax these constaints and offer more flexibility and extensibility.

The MPEG-4 Syntax Description Language, MSDL, was originally conceived to deliver flexibility and extensibility by providing a flexible means to describe algorithms (including new ones in the future) and related syntax. With time, however, the MSDL[10] is evolving into the role of systems layer, addressing system capabilities needed to support MPEG-4 functionality classes such as content-based manipulation, efficient compression, universal accessibility, and basic functionalities, in addition to its original role.

17.7.1 MSDL Scope

To support flexibility and extensibility, MSDL[10] defines three types of decoder programmability as follows:

- Level 0 (nonprogrammable) decoder incorporates a prespecified set of standardized algo-

rithms which must be agreed upon by the decoder during the negotiation phase.

- Level 1 (flexible) decoder incorporates a prespecified set of standardized tools which can be flexibly configured into an algorithm by the encoder during the scripting phase.
- Level 2 (extensible) decoder provides a mechanism for the encoder to download new tools as well as algorithms.

The aforementioned definitions of various levels are nested, that is, Level 2 programmability assumes Level 1 capabilities and Level 1 programmability assumes Level 0 capabilities. Currently MPEG-4 is planning to address Level 0 and Level 1 capabilities only and Level 2 may be addressed later.

To support MPEG-4 functionalities to the fullest, MSDL deals with presentable *audiovisual objects* (audio frames, video frames, sprites, 3D objects, natural or synthetic objects, . .) and their *representation methods* (waveform, splines, models, . .). The decoder decodes each object and either based on prespecified format or under user control *composes* and *renders* the scene. All audio–visual objects are assumed to have the necessary interfaces.

In Fig. 17.18 we show the position of MSDL in a vertical stack of protocols as was envisaged earlier. Then, the MSDL was considered as in between the channel coding protocols and the presentation protocols, with some overlap with these protocols.

Owing to MSDL recently taking over the role of systems layer also, it is expected that the overlap with presentation protocols will be even more significant.

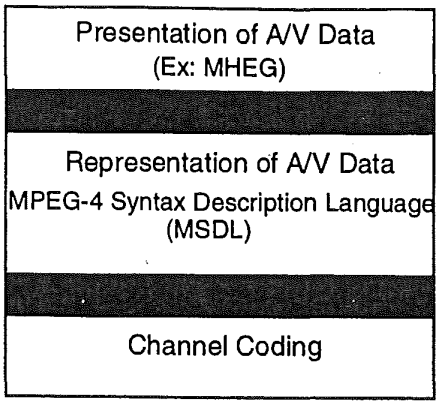


Fig. 17.18 Position of MSDL in a vertical stack of protocols.

Earlier, while discussing MPEG-4 requirements, we mentioned one role of MSDL as the glue between Tools, Algorithms, and Profiles. Since MSDL uses object-oriented methodology, we can represent Tools, Algorithms, and Profiles as objects, each with a clearly defined interface for input/output. In the context of MPEG-4, a number of Tools when connected together in a meaningful manner result in an Algorithm, and a Profile contains a number of related Algorithms. Figure 17.19 more clearly illustrates the role of MSDL as a glue between Tools, Algorithms, and Profiles.

17.7.2 MSDL Requirements

Our discussion of the scope of MSDL, although quite helpful in clarifying the range of functions MSDL is addressing, is still not specific enough. We now list various categories of requirements[10] that MSDL is expected to satisfy and list detailed requirements in each category:

- *General Requirements*
- *Structural Requirements*
- *Interface Requirements*
- *Construction Requirements*
- *Downloading Requirements.*

17.7.2.1 General Requirements

1. The MSDL should allow for genericity up to Level 2

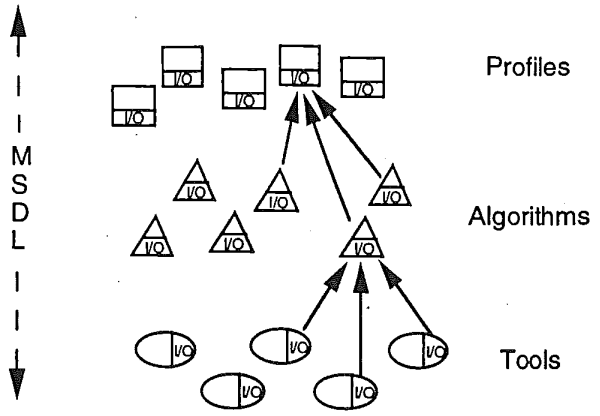


Fig. 17.19 Role of MSDL in binding Tools, Algorithms, and Profiles.

2. The MSDL shall provide rules for parsing the bitstream. This will define the procedure for understanding the ensuing data.
3. The MSDL should define a Structure Module, the Interface Description Module, an Object Construction Module, a Downloading Description Module, and a Configuration Module.
4. The MSDL should provide solutions for a configuration phase, a learning phase, and a transmission phase.
5. The MSDL should provide means for synchronization.
6. The MSDL should provide a means for error-free transmission of its data files (which contain set-up information between encoder and decoder, downloading of new tools/logarithms, etc.)

17.7.2.2 Structural Requirements

1. The MSDL should provide a structural library for the objects (tools, algorithms, and profiles).
2. The MSDL should provide different ranges of programmability for different applications.
3. The MSDL objects should be extensible with a mechanism to add in the future, ISO standardized data.
4. The MSDL structure shall use constructs, notations, and formats in a consistent manner.
5. The MSDL should provide for efficient coding of object identifiers.

17.7.2.3 Interface Requirements

1. The MSDL should provide ways to describe objects, this means describing their interfaces, what information fields exist, and which fields have to be set up each time the object is used and which ones do not.
2. The MSDL shall use constructs, notation, and formats in a consistent manner.
3. The MSDL shall code data with maximum efficiency. This means that dynamic and static parameters shall be coded with maximum efficiency. Further, there shall be an efficient mechanism to default such parameters when their coding is necessary.

17.7.2.4 Construction Requirements

1. The MSDL should provide a language to link elementary objects or already built complex objects to create new complex objects.
2. The MSDL via OCM should allow for easy mapping from the description of the objects used and connections among these objects to a real coding/decoding processor.

17.7.2.5 Downloading Requirements

1. The MSDL should be a language that has a flexibility and expressiveness similar to that of other general-purpose programming languages.
2. The MSDL should have as a goal to construct a machine-independent downloading language.
3. The MSDL via DDM should support dynamic binding of objects during the run-time to allow adaptation and adjustment during communication.
4. The MSDL should provide for mechanisms to ensure security and preservation of intellectual property rights.

17.7.3 MSDL Parts

Having reviewed the requirements that the MSDL must satisfy, it is clear that the MSDL effort needs to be clearly partitioned into a number of areas.[10] Currently, the ongoing MSDL work expects to address the following areas:

Architecture (MSDL-A): Specifies the global architecture of the MPEG-4 system. This includes the role of MPEG-4 system to support complete audio–visual applications as well as conceptual objects and their data content exchanged between encoder and decoder.

Class Hierarchy Definition (MSDL-O): Specifies particular classes of objects that are useful for specific audio–visual applications. It includes the definition of class libraries of MSDL.

Readable Language Specification (MSDL-R): Describes a readable format for transmission of decoder scripts.

Binary Language Specification (MSDL-B): This is a specification of a binary executable format for scripts or descriptions. This is the executable binary language understood by the decoder.

Syntactic Description Language (MSDL-S): This is a specification of a language that can be used to describe the bitstream syntax. It is expected to be

an extension of the MPEG-2 syntax description method into a formal well-defined language that can be interpreted by machines.

Multiplex Specification (MSDL-M): Describes a procedure for multiplexing of encoded information.

Work on all the aforementioned parts of MSDL is currently in progress and thus it is best to discuss only the basic concepts behind these methods rather than any details.

17.7.4 MSDL-A

The MSDL-A specifies the global architecture of an MPEG-4 system; an MPEG-4 system is a system for communicating audio–visual objects. An audiovisual (AV) object may be aural or visual, 2D or 3D, static or time varying, natural or synthetic, or combinations of these types. The architecture for communicating these objects is as follows. The coded AV objects and their spatiotemporal relationships if any are transmitted by an MPEG-4 encoder to an MPEG-4 decoder. At the encoder, the AV objects are compressed, error protected, multiplexed, and transmitted (downstream) to the decoder. Actual transmission may take place over multiple channels each with different service qualities. At the decoder, AV objects are demultiplexed, error corrected, decompressed, composited, and presented to the user. The user may interact with the presentation; the user interaction information may be used locally at the decoder or transmitted back (upstream) to the encoder. Figure 17.20 shows the architecture of the MPEG-4 system from the MSDL aspect.

The encoder and the decoder exchange configuration information prior to the encoder transmitting AV objects. The encoder determines the classes of algorithms, tools, and other objects that the decoder must have to process coded AV objects it intends to send to the decoder. Each class of objects can be defined by a data structure and the executable code. The definitions of missing classes are downloaded to the decoder, where they supplement or override existing or predefined classes at the decoder. If the decoder needs new class definitions in response to the user interaction it can request these from the encoder to download additional class definitions, in parallel with transmitted data. Figure 17.20 clarifies these various aspects just discussed.

For the new classes that the encoder communicates to the decoder, class definition can be considered as the header and the data component as the

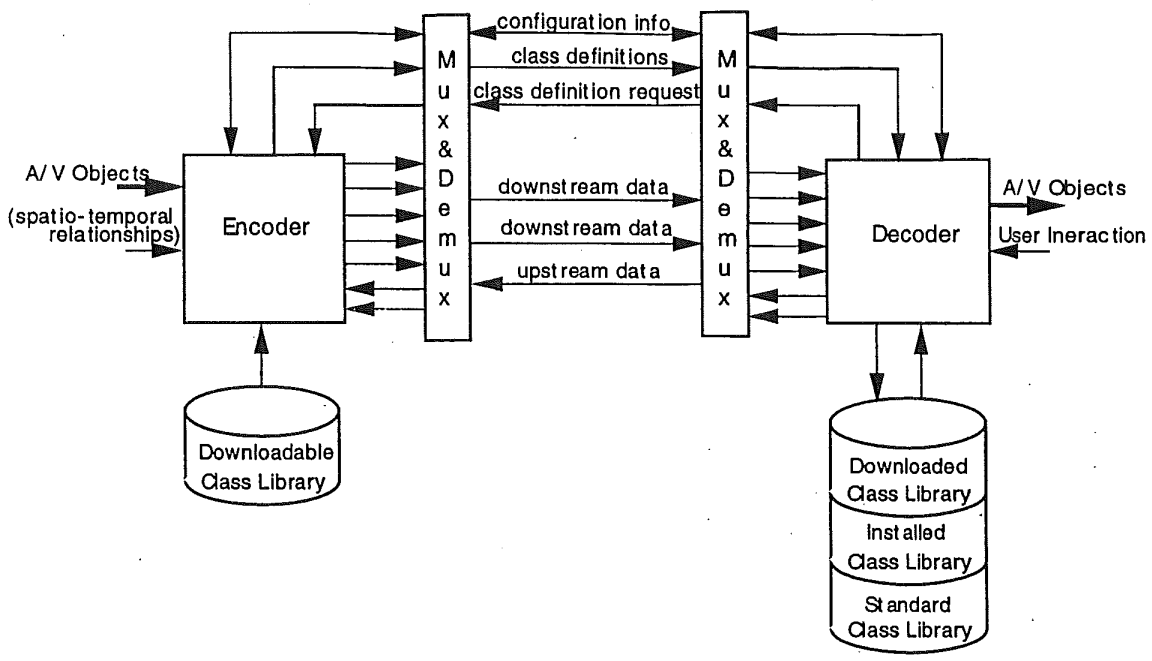


Fig. 17.20 Overall System architecture from the MSDL aspect.

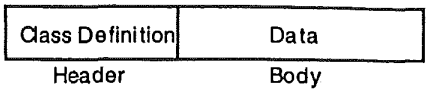


Fig. 17.21 Audio–visual objects specification.

body of a transmitted A/V objects. The class definition specifies the data structure and the methods that are used to process the data component. The specification of an AV object is illustrated in Fig. 17.21.

Not all AV objects need to transmit their class definition along with the data, for example, in the case of an MPEG-1 movie, its class definition can be found in a standard library. Likewise, if an AV object uses the same class definition as one already sent, it does not need to be sent again. Data are used to control the behavior of objects. For some AV objects, there may be no data component, as the behavior of the object may be predetermined. It is expected that all class definitions may be transmitted lumped separate from the data components.

17.7.5 MSDL-O

The MSDL-O specifies classes of MPEG-4 objects needed in specific applications. In particular it addresses classes of objects that would need to be constructed during decoding to render and manipulate the audio–visual scene.

Audio–visual objects correspond to data structures that can actually be rendered by the compositor. In a coded bitstream audio–visual objects are represented as a sequence of content objects of various types.

Process Objects correspond to a process that will be applied on content or presentable objects and will return a new presentable object. Hence, each process object contains an *Apply* method.

A partial class hierarchy for MSDL-O is as follows:

- Audio–visual and Content Objects
  - —Audio–visual Scene Model
  - —Hierarchical Model for Spatiotemporal Scene
  - —Modeling 2D Objects
  - —Texture Information
  - —Shape Information
  - —Motion Information
  - —Depth and Opacity Information
  - —Location Information
- Process Objects
  - —Quantizer Object
  - —Transformer Object
  - —Compensator Object.

17.7.6 MSDL-R

The MSDL-R specifies a readable format or a textual representation of the algorithm. It uses a subset of (C and) C++ as the starting point; however, it is envisaged that new capabilities not present in C++ such as multithreading and message passing will need to be added at some point.

The MSDL-R features include the following.

- C variable names, expressions, and function calls
- C constructs: if–else, for, while, for which conditions are limited to constants, variables, and function calls.
- Return statements.

The MSDL-R excludes the following:

- Definition of functions, procedures, and global variables
- Preprocessor directives
- Statements such as goto, break, and continue
- Virtual functions
- Structures or Classes and access to them
- Type casts.

Overall, for security reasons, the MSDL-R is strongly typed, that is, no casts, function parameter type checking, no implicit type change.

An MSDL-R program is a procedure. It has local parameters, local variables, global variables, and instructions. No new data structure can be defined in MSDL-R; only the standard elementary types and decoder known types are allowed.

17.7.7 MSDL-B

The MSDL-B is the specification of the binary executable format and can be generated from MSDL-R. Two approaches have been identified. In the first approach, MSDL-B is a coded version

of the high-level language MSDL-R; for example, using Huffman coding of MSDL-R keywords. The second approach involves compilation of MSDL-R to a platform-independent low-level representation. Currently both approaches are being evaluated and there are concrete proposals for each.

In the first approach, MSDL-B is obtained by direct compression of the MSDL-R code. MSDL-R keywords, arithmetic operators, and variables are mapped to predefined codewords. The decoder can reconstruct the original MSDL-R code, but without the symbolic names used for class instances, variables, and methods. MSDL-B can be executed directly or translated to native machine code.

In the second approach, MSDL-B is obtained by compilation at the encoder. Compilation can be done based on "lex" or "yacc" which are textual syntax analysis tools. Initially, the resulting MSDL-B code may be suboptimal in terms of efficiency but its efficiency can be improved. Instructions are 32 bits long and are interpreted by Forth (programming language) like stack-based interpreters. The interpreter and tools are written in C++ and linked together. If required, a direct compilation to the native machine language is also possible.

17.7.8 MSDL-S

The goal of MSDL-S is to provide a language that can be used to describe the bitstream syntax. It is intended to somewhat separate the definition of bitstream syntax from the decoding/rendering process. Since MPEG-4 intends to provide flexibility and programmability it becomes necessary to separate syntax from processing to allow content developers to disclose only the bitstream structures they may use but not the details of any processing methods. Another advantage of this separation is that the task of the bitstream architect is greatly simplified, allowing focus on decoding and preparing it for display rather than the mundane task of extracting bits from the bitstream. Further, this separation also eases the process of compliance with the overall bitstream architecture.

Some explanation is in order regarding the relationship of MSDL-S and MSDL-R. The MSDL-R, as discussed earlier, addresses the general programming facilities of MSDL, and is the language in which decoding and generic processing operations are described. On the other hand, MSDL-S is an orthogonal subset of MSDL-R; orthogonality implies that the two are independent, that is, the specification of MSDL-S does not affect the specification of MSDL-R. There is, however, an assumption of commonality at the level of their capability to define object hierarchies. Currently, MSDL-S assumes a C++ or Java like approach to be employed for MSDL-R. For example, MSDL-S can be thought of as generalizing the concept of declaring constants with hard coded values to declaring constants that derive values from the bitstream. An MSDL-R programmer can assume that a constant is parsed from the bitstream before it is accessed, similar to the way a programmer is not concerned about how the initialization of a constant occurs, as long as it does.

We now briefly introduce the main elements of Syntax Description Language, the work on which is currently in progress.

- Elementary Data Types
  - —Constant-length direct representation bit-fields
  - —Variable-length direct representation bit-fields
  - —Constant length indirect representation bit-fields
  - —Variable length indirect representation bit-fields
- Composite Data Types
- Arrays
- Arithmetic and Logical Expressions
- Temporary Variables
- Control Flow Structures
- Parsing Modes.

17.7.9 MSDL-M

The goal of MSDL-M is to specify multiplexing and demultiplexing procedures for coded audio–visual information. In particular this goal can be broken down into a number of functions, such as interleaving of multiple compressed streams into a single stream, recovery of a system time base, managing of a decoder buffer, synchronization of multiple compressed data streams on

decoding, and others. Owing to a large variety of anticipated MPEG-4 applications, such as broadcast, real-time bidirectional communication, database retrieval, and others, substantial flexibility needs to be provided.

The starting point for MSDL-M is the MPEG-2 systems specification (13818–1), the ITU-T multiplex specification H.223, and the evolving ITU-T multiplex specification H.22M. This approach is intended to maintain commonality with various current and evolving standards. To proceed further with the design of MSDL-M, a number of categories of requirements, each with its own list of requirements, are taken into account. The main categories of requirements are as follows:

- General Requirements
- Application-Dependent Requirements
- Timing/Synchronization Requirements
- Error Resilience Requirements
- Network Adaptation Requirements
- Compatibility Requirements.

The specification of MSDL-M is currently being developed taking into consideration the detailed requirements in each of these categories and the system functionalities already in existence in other standards. A partial list of specific issues that are being addressed is as follows:

- Clock Synchronization
- Stream Synchronization
- Error Resilience
- Configuration
- Periodic Retransmission
- Buffer Management Models.

17.7.10 MSDL Directions

The MSDL effort is expected to result in part 1 of the MPEG-4 Working Draft-1 by November 1996, coincident with the schedule for parts 2 and 3. The schedule for following iterations of the Working Draft (Working Draft-2 and Working Draft-3), the Committee Draft, and the International Standard was discussed earlier while discussing the schedule for Video.

17.8 SYNTHETIC AND NATURAL HYBRID CODING (SNHC)

In recent years, with advances in digital video technology and fueled by ever increasing demand for sophisticated multimedia, traditionally separate fields of natural images/video and synthetic images/animations have been merging at a breathtaking speed. For example, several recent movies have included composites of natural and quite realistic looking synthetically generated scenes, with composition performed using chroma keying (Chapter 5). The key to even more sophisticated multimedia is not only to produce more sophisticated composites but also to provide the ability to interact with audio–visual objects in these scenes; this ability is severely limited at the present time. Traditional audio–visual presentation has simply consisted of displaying a frame of video and playing with its accompanying audio soundtrack/s. The increasing popularity of the World Wide Web, which offers the opportunity to interact with and thus control the presentation of data in a highly nonlinear fashion, implies that increasingly in the future, multimedia data will also be interacted with and presented in the same way. Thus, coding of natural and synthetic image data, ability to interact with and manipulate objects in the coded domain, and the ability to control the order of presentation are related important functionalities that MPEG-4 is addressing via the Synthetic and Natural Hybrid Coding (SNHC)[12] effort.

The most obvious way of providing access to individual objects in the coded domain is by coding each object individually. In addition, it is anticipated that by coding objects separately some gain in overall compression efficiency may also be obtained as different coding strategies can be more easily used for different objects, as appropriate. A normal way to proceed would then simply appear to be to separately encode segmented or prerendered objects. However, there are certain applications such as video games, CAD, medical, and geographical where many different types of image data are employed and no universal standards exist, the volume of data generated is usually too high, and user interactivity places additional synchronization requirements. In such cases, a differ-

ent coding paradigm is better suited and involves sending coded representation of parameters to the decoder followed by rendering at the decoder prior to display. In general, when dealing with both natural and synthetic data, a combination of the two paradigms appears to offer a practical solution.

The MPEG-4 SNHC effort is aimed at addressing the needs of sophisticated multimedia applications involving synthetic and natural, audio and video data in an efficient, yet practical way. The SNHC effort is expected to undergo two major phases, a competitive phase followed by a collaborative phase, much like the ongoing effort in MPEG-4 Video and Audio. MPEG-4 is currently seeking candidate proposals for consideration for standardization and is currently in its competitive phase.

17.8.1 SNHC Goal and Motivation

A more precise goal of MPEG-4 SNHC[12] work is to establish a standard for efficient coding, representation, and integration of 2D/3D, synthetic–natural hybrid data of an audio–visual nature.

This standard is intended to support a wide variety of multimedia experiences on a range of current and future platforms. SNHC is expected to be employed in a variety of applications such as video games, multimedia entertainment, educational, medical surgery, industrial design and operations, geographical visualizations, and so forth.

17.8.2 SNHC Architecture and Requirements

The functional architecture for the SNHC Decoding System[12] is shown in Fig. 17.22. It consists of System Layer Decoder, a Video Decoder, an Image Synthesizer, an Audio Decoder, an Audio Synthesizer, a Cache, and a Display Processor. The Display Processor is controlled by the user to control compositing of audio, video, and graphics objects for presentation. A Cache is used to store objects in repeated use such as 3D geometry, texture, audio clips, and segmentation masks.

The SNHC effort, by defining what is referred to as a media model, is expected to focus on providing building blocks for efficient coding and rendering in interactive environments, which synchronize 3D embedding of various objects such as images, real-time video, audio, and static or animated shapes. These objects may be either natural or synthetic in nature. We now list the various issues that need to be resolved in completely defining a media model:

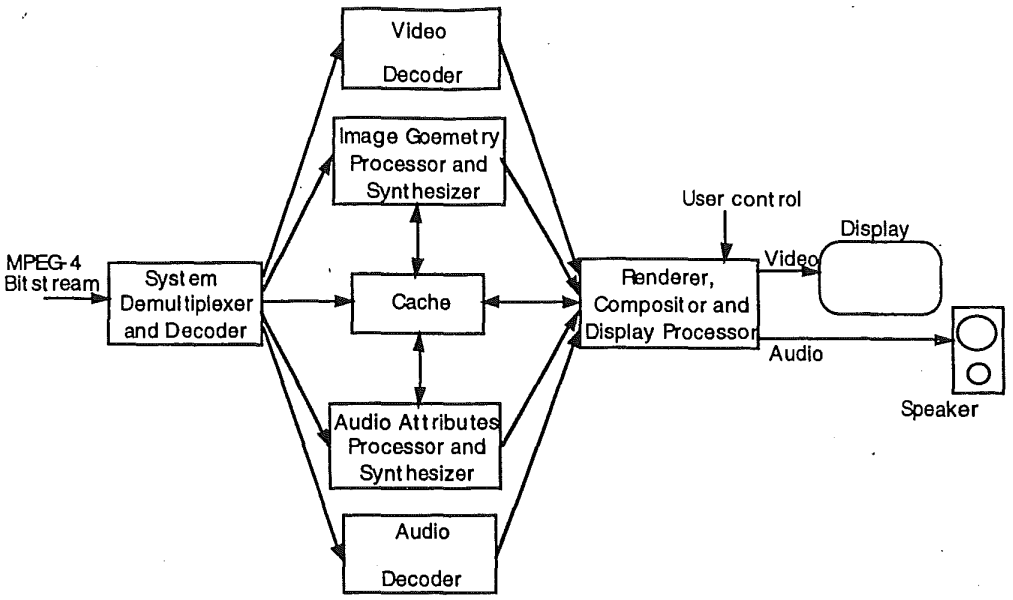


Fig. 17.22 Overall Decoding System Architectures from the SNHC aspect.

- Basic visual and audio primitives (polygons, sprites, textures, wavetables)
- 2D and 3D spatial models including geometric and structural properties
- Surface and material properties relative to scene illuminants
- Texture mapping attributes for still and moving images
- Combining natural and synthetic, static and moving, images and audio
- Combining synthetic spatial structures with traditional audio and video
- Structures for compositing scenes from objects
- Conventions for coordinate systems
- Parameterized models and free-form deformations
- Compressed and uncompressed data storage formats
- Static and dynamic behavior of objects
- Representation of viewers, cameras, and sound sources for rendering
- Naming and hyperlinking conventions for servers and libraries
- Modeling for different class of terminal resources and scene degradation.

17.8.3 SNHC Test Conditions

SNHC currently is in the process of establishing a standardized digital 2D/3D data set referred to as the Virtual Playground (VP). The VP will include critical data sets needed to discern performance differences between various competing approaches expected to be submitted to MPEG-4 for SNHC evaluation.

The VP is a collection of data sets representing agreed upon objects. These objects will be stored in popular formats such as VRML 1.0 for geometric objects, JPEG for image textures, and AIFF for audio clips. These objects, when selected, will be downloadable from MPEG World Wide Web home page. Various compositions of these VP objects will be used to demonstrate applications of MPEG-4 as well as uncover areas for study and standardization.

The VP is expected to include objects with features such as animations, complex geometry, texture mapping, and various sound sources to test complexity and synchronization issues. CAD models of various items such as furniture and vehicles may be used to represent complex geometry; talking synthetic characters may exercise the use of parametrized models, audio synchronization, and 3D localization.

17.8.4 SNHC Evaluation

SNHC focuses on coding for storage and communication of 2D and 3D scenes involving synthetic–natural images, sounds, and animated geometry. Further, the coded representation should facilitate various forms of interactions. The SNHC group is seeking algorithm and tool proposals for efficient coding and interactivity in the coded domain; the following is a partial list[11] of topics of potential interest:

- Compression and simplification of synthetic data representations—synthetic and natural texture, panoramic views, mapping geometry, mapping photometry, animation, and deformation
- Parametrized animated models—encoding of parameterized models and encoding of parameter streams
- New primitive operations for compositing of natural and hybrid objects
- Scalability—extraction of subsets of data for time-critical use and time-critical rendering
- Real-time interactivity with hybrid environments
- Modeling of timing and synchronization
- Synthetic Audio

The MPEG-4 SNHC effort, in the competitive phase, similar to the MPEG-4 video effort, has made an official call for proposals to undergo evaluation. The evaluation of proposals is likely to take place later in 1996. These proposals are expected to be evaluated by a group of experts. The evaluation criteria are likely to be based on functionality addressed, coding efficiency, quality of decoded model, real-time interactivity, anticipated performance in the future, and implementation cost. Each proposer is required to submit the following items:

- Technical Description detailing scope, advantages, description, and statistics
- Coded Bitstream of test data set and an executable decoder capable of decoding submitted streams

- A D1 tape showing results of compression/decompression, and simplification or any other process that modifies the data.

17.8.5 SNHC Directions

The SNHC proposers are expected to provide algorithms and tools to address certain functionalities, and provide associated media models and bitstream format. After the evaluation of SNHC proposals, the selected proposals will undergo further testing by core experiments, similar to that being used for MPEG-4 video. These experiments are expected to cover many categories such as compression, scalability, new media primitives, timing and synchronization, and real-time interactivity. A collection of algorithms, tools, and associated media models and bitstream formats is expected to result from the SNHC standardization effort. It is also expected that during the convergence phase there will be a need to harmonize SNHC, MSDL, MPEG-4 Video, and MPEG-4 Audio. In response to the recent SNHC Call for Proposals,[11] the evaluation of proposals is expected to take place in September 1996.

The MPEG-4 SNHC effort is expected to contribute to tools and algorithms for part 2 and part 3 of the MPEG-4 Working Draft-1. Part 2 of MPEG-4 deals with synthetic and natural visual representation and coding tools and algorithms, while part 3 of MPEG-4 deals with synthetic and natural aural representation and coding tools and algorithms. The MSDL is expected to provide the framework and glue for uniformly addressing natural and synthetic representations. The SNHC effort is thus expected to contribute to tools and algorithms in parts 2 and 3 of the MPEG-4 Working Draft-1 by November 1996, coincident with the tools and algorithm contributions from natural video to part 2 and natural audio to part 3. The schedule for following iterations of the Working Draft (Working Draft-2 and Working Draft-3), the Committee Draft, and the International Standard was discussed earlier while discussing the schedule for Video.

17.9 THE FUTURE

The MPEG-4 standard, owing to its flexibility and extensibility, is expected to remain a very useful standard for many years to come. Although MPEG-4, when completed, is expected to support many functionalities, only the very basic functionalities may be applied initially for specific applications. This may occur on next generation video signal processors or on custom VLSI chips. The more advanced functionalities are likely to be deployed only at a much later stage when very powerful and flexible video processors that can take full benefit of the MSDL capabilities become possible.

In the very near term, a number of applications, products, and services are expected to be based on MPEG-1, MPEG-2, and the MPEG-4 standards. In the United States, it appear that even before the introduction of HDTV, digital TV may be introduced; currently the FCC Advisory Committee on Advanced Television Standards (ACATS) is finishing up specification of digital TV standard, the video and systems part of which are based on the MPEG-2 standard. A number of types of devices (e.g., set-top boxes) allowing access and interactivity to a new generation of services are expected to be deployed. Relating to these devices, besides MPEG standards, the work done by DAVIC (Digital Audio and Video Council) and MHEG (Multimedia and Hypermedia Experts Group) is quite relevant.

17.9.1 Future Standards

The MPEG-4 standard is promising unparalleled flexibility and extensibility. This would seem to imply that if it succeeds in achieving its goals, there would not be a need for a future audio–visual standard. In reality, there will still be a need for audio–visual standardization activity, although it may not be mainly related to standardization of coding methods. We briefly discuss one such topic on which there may be a need for future MPEG standards activity.

17.9.1.1 Man-Multimedia Service Interface

Man–Multimedia Service Interface (MMSI)[14] can be defined as a set of high-level functions that enable user to access, utilize, and manage multimedia services. Examples of specific operations may be selection, browsing, navigation, customization,

and so forth. This activity may use existing and emerging multimedia standards and may provide solutions for a wide range of multimedia access devices and mechanisms including minimal resource devices.

Multimedia services are typically employed by a number of types of industries such as content providers, service operators, network operators, platform manufacturers, and end users. Each industry has its own set of requirements; however, taking into account a common set of requirements across various industries the following trends seem clear.

- The need for standardized navigation and browsing facilities
- The need for clear separation between contents and applications on one hand, and actual platform and configurations on the other hand
- The need for standardization of multimedia information, balanced by the need for differentiation between competing systems.

It is envisaged that multimedia services under MMSI may be categorized as follows:

- *Composition-Oriented Services:* These services include providing access to decoded representations of objects or actual decoding.
- *Distribution-Oriented Services:* These services may provide distribution of objects, scripts, content data, and so forth in a distributed environment.
- *Presentation-Oriented Services:* These services support the development of a dedicated environment where published and distributed multimedia information encoded according to a given representation will be perceived as an application-specific man–machine communication.
- *Utility Services:* This implies the need for user service groups to regulate access to multimedia information. This includes accounting, billing, security, and so forth.

This future work item[14] has recently been brought to MPEG for consideration and will undergo discussion of its relevance to MPEG and possible reinterpretation. There appears to be some commonality between this topic and topics addressed by MPEG-4, which can provide a smooth transition to the next MPEG standard.

17.9.2 Example Applications

Now it is time for some predictions about the future of audio–visual applications. This is where we the authors get to look into our crystal ball and with a fairly straight face try to make sense of where the technology is heading (a generally impossible task!). And yes, even knowing well in advance that we are going to get a good-natured ribbing from our colleagues and friends, we still dare to stick our necks out!! So, here goes. . .

17.9.2.1 Short Term (Within 3 to 5 years)

- Digital TV to home via terrestrial broadcast and cable. Interactive features allowing viewer participation
- Multimedia of high quality via Internet and other means. Variety of multimedia services such as multimedia e-mail, shared real-time multimedia work spaces, VOD (video on demand), MOD (movies on demand), variety of databases, multimedia information access, and retrieval tools
- Wireless multimedia on a number of devices such as personal phones that can be used anywhere, cellular phones, Personal Digital Assistants (PDAs), wireless Internet access terminals, and so forth
- Portable multimedia terminals that can be used for advanced networked games, video phones, or for database access

17.9.2.2 Medium Term (Within 5 to 12 years)

- Digital HDTV via satellite, terrestrial and cable, for reception by home TV, PC-TVs etc. Advanced interactive features allowing viewer participation
- Specialized services offering customized multimedia newspapers, customized magazines, customized movies, and special events
- Advanced, personal wireless multimedia communicators providing seamless access to a variety of services and systems locally, regionally, nationally, and globally.
- High-quality wireless multimedia headsets bringing capabilities of networked video game playing, video database access, Internet access, telepresence and others via functionalities such as

stereoscopic viewing, high-end graphics, 3D sound, and others
- PCs and high end game/Internet systems offering high-quality stereoscopic/multiviewpoint access, graphics capabilities, and interactivity.

17.9.2.3 Long Term (Beyond 15 years)

- User configurable personalized information multimedia services, devices, and networks
- Virtual entertainment gear (suits) combining advanced multimedia and variety of sensors to provide advanced communication, networking, games, and experiences
- Holographic TV/display systems offering surreal interactivity and immersion experiences.

17.10 SUMMARY

In this chapter we have taken a step beyond MPEG-2 and introduced the next MPEG (MPEG-4) standard currently in progress. The necessary background information leading up to the MPEG-4 standard and the multiple facets of this standard are introduced showing relationships to the MPEG-1 and the MPEG-2 standards. In particular we have discussed the following points:

- The bitstream syntax, decoding process, optional modes, and encoder configuration of the low-bitrate ITU-T H.263 standard which is the starting basis for the MPEG-4 standard
- Recent advances in Transform Coding, Wavelet Coding, Fractal Coding, Region-/Object-Based Coding, and Model-Based Coding that may offer promising solutions for MPEG-4 Video coding
- An introduction to MPEG-4 including a discussion of its focus, applications, functionality classes, and how it is organized
- The requirements for MPEG-4 including terminology, advanced functionalities, basic functionalities, and analysis of evolving requirements
- The MPEG4 Video development process including test conditions, first evaluation, results of evaluation, formulation of core experiments, definition of the Verification Model, and directions of ongoing work
- A brief introduction to the MPEG-4 Audio development process
- Discussion about the MPEG-4 Synax Description Language including its scope, requirements, its various parts, architecture, details of its parts, and directions of ongoing work.
- The MPEG-4 Synthetic and Natural Hybrid Coding effort, its goals, architecture and requirements, test conditions, evaluation process, and directions of ongoing work
- The Future—what are the implications of MPEG-4 and what lies beyond? Example applications involving audio–visual coding that seem promising for the short term, the medium term, and the long term.

REFERENCES

1. AOE Group, "MPEG-4 Proposal Package Description (PPD)—Rev. 3," ISO/IEC JTC1/SC29/WG11 N0998, Tokyo (July 1995).
2. AOE Group, "MPEG-4 Testing and Evaluation Procedures Document," ISO/IEC JTC1/SC29/WG11 N0999, Tokyo (July 1995).
3. L. Chiariglione, "MPEG-4 Call for Proposals," ISO/IEC JTC1/SC29/WG11 N0997, Tokyo (July 1995).
4. A. Puri, "Status and Direction of the MPEG-4 Standard," *International Symposium on Multimedia and Video Coding*, New York (October 1995).
5. ITU-T, "Draft ITU-T Recommendation H.263: Video Coding for Low Bitrate Communication" (December 1995).
6. T. Ebrahimi, "Report of Ad Hoc Group on Definition of VMs for Content Based Video Representation," ISO/IEC JTC1/SC29/WG11 MPEG 96/0642, Munich (January 1996).
7. A. Puri, "Report of Ad Hoc Group on Coordination of Future Core Experiments in MPEG-4 Video," ISO/IEC JTC1/SC29/WG11 MPEG 96/0669, Munich (January 1996).
8. MPEG-4 Requirements Ad Hoc Group, "Draft of MPEG-4 Requirements," ISO/IEC JTC1/SC29/WG11 N1238, Florence (March 1996).
9. MPEG-4 Video Group, "MPEG-4 Video Verification Model Version 2.0," ISO/IEC JTC1/SC29/WG11 N1260, Florence (March 1996).
10. MSDL Ad Hoc Group, "MSDL Specification V1.1," ISO/IEC JTC1/SC29/WG11 N1246, Florence (March 1996).

11. MPEG-4 Integration Group, "MPEG-4 SNHC Call for Proposals," ISO/IEC JTC1/SC29/WG11 N1195, Florence (March 1996).
12. MPEG-4 Integration Group, "MPEG-4 SNHC Proposal Package Description," ISO/IEC JTC1/SC29/WG11 N1199, Florence (March 1996).
13. L. Chiariglione, "Resolutions of 34th WG11 Meeting," ISO/IEC JTC1/SC29/WG11 N1186, Florence (March 1996).
14. ISO/IEC JTC1/SC29, "Proposal for a New Work Item: Information Technology—Man–Multimedia Service Interface," ISO/IEC JTC1 N3333 (December 1994).
15. Delta Information Systems, "Technical Work in the Area of Video Teleconferencing: Advanced Coding Techniques," Final Report Contract DCA100–91-C-0031 (December 1994).
16. A. Puri, R. L. Schmidt, and B. G. Haskell, "SBASIC Video Coding and Its 3D-DCT Extension for MPEG-4 Multimedia," *Proc. SPIE Visual Commun. Image Proc.,* pp. 1331–1341 (March 1996).
17. T. Ebrahimi, E. Reusens, and W. Li, "New Trends in Very Low Bitrate Video Coding," *Proc.* IEEE 83 (June 1995).
18. J. Katto et al., "A Wavelet Codec with Overlapped Motion Compensation for Very Low Bit-Rate Environment," *IEEE Trans. Circuits Syst. Video Technol.,* pp. 328–338 (June 1994).
19. H. Chen, C. Horne, and B. G. Haskell, "A Region Based Approach to Very Low Bitrate Video Coding," AT&T Internal Technical Memorandum (December 1993).
20. J. Osterman, "Object-Based Analysis–Synthesis Coding Based on the Source Model of Moving Rigid 3D Objects," *Signal Proc. Image Commun.,* pp. 143–161 (May 1994).
21. K. Aizawa, "Model-Based Image Coding," *Proc. SPIE Conf.* 2308: 1035–1049 (1994).
22. A. Puri, "Efficient Motion-Compensated Coding for Low Bit-rate Video Applications," Ph.D. Thesis, City Univ. of New York, 1988.
23. H. Peterson, "Report of the Ad Hoc group on MPEG-4 Video Testing Logistics," ISO/IEC JTC1/SC29/WG11 Doc. MPEG95/0532 (November 1995).
24. J. Osterman, "Report of the Ad Hoc group on Evaluation of Tools for non tested Functionalities of Video submissions," ISO/IEC JTC1/SC29/WG11 Doc. MPEG95/0488 (November 1995).
25. W. Li, Y.-Q. Zhang, "Vector-Based Signal Processing and Quantization for Image and Video Compression," Proc. of the IEEE, vol. 83, No. 2, pp. 317–335 (February 1995).
26. D. E. Pearson, "Developments in Model-Based Video Coding," Proc. of the IEEE, vol. 83, No. 6, pp. 892–906 (June 1995).
27. M. Li and R. Forchheimer, "Two-View Facial Movement Estimation," IEEE Transac. on Circuits and Systems for Video Technology, vol. 4, No. 3 (June 1994).
28. F. Lavagetto and S. Curinga, "Object-oriented Scene Modelling for Interpersonal Video Communication at Very Low Bit-rate," Signal Processing: Image Commun., pp. 380–395 (Oct. 1994).
29. T. Chen et al., "Speech Assisted Lip Synchronization in Audio-Visual Communications," Int. Conf. on Image Proc. (Oct. 1995).
30. G. Bozdagi, A. M. Tekalp, and L. Onural, "3-D Motion Estimation and Wireframe Adaptation Including Photometric Effects for Model-Based Coding of Facial Image Sequences," IEEE Transac. on Circuits and Systems for Video Technology, vol. 4, No. 3, pp. 246–256 (June 1994).
31. O. Lee, and Y. Wang, "A Region-Based Video Coder Using an Active Mesh Representation," Proc. IEEE Workshop on Vis. Signal Proc & Commun., pp. 168–172, Sept. 1994.
32. T. Chen and R. Rao, "Audio-Visual Interaction in Multimedia: From Lip Synchronization to Joint Audio-Video Coding," to appear IEEE Circuits and Devices magazine.