conferencing and multimedia conference control. The subjective effects of end-to-end delay in real-time conferences were reviewed, and an analysis of the components of audio delay presented.

Multimedia and videoconferencing systems have so far achieved practical and commercial success in specific applications such as business meetings, telecommuting, and education. Consumer desire for video telephony is still mostly speculative, but at a minimum, issues of cost, end-to-end delay, video quality, and ease of use will have to be addressed before multimedia communication can hope to supplant traditional telephone service.

The standards described in this chapter, together with efforts by manufacturers and industry organizations like IMTC, seem likely to address the ease-of-use and interoperability issues. The cost of the necessary hardware is steadily declining and will reach consumer levels very soon. But major improvements in end-to-end delay and video quality may require higher bit rates than those available to consumers thus far. Happily, new high-bit-rate access technologies such as cable modems and xDSL seem to be coming quickly. These, possibly combined with effective Internet resource allocation mechanisms, may allow a real revolution in the communication tools we use every day. At the least, we can be sure multimedia standards and networks will continue to evolve rapidly for the foreseeable future.

The next chapter covers the MPEG-1 and MPEG-2 standards for audio and video compression. Unlike the ITU-T standards, the MPEG standards are primarily focused on storage, playback, and broadcast applications, rather than two-way real-time communication.

11 MPEG Compression

CHAPTER

11.1 Introduction

The Moving Pictures Experts Group (MPEG), an international standards committee, has produced a set of standards for audio and video compression that are good examples of sophisticated compression strategies and that are also commercially important. Starting from an initial focus on video compression at approximately 1.2 Mbps on CD-ROM, the MPEG standards have expanded to become the lingua franca of high-quality audio-video compression in the 1990s.

Formally, MPEG is Working Group 11 of Subcommittee 29 of Joint Technical Committee 1 of the International Standards Organization and the International Electrotechnical Commission. As an international standards body, MPEG follows a strict set of rules aimed at maximizing consensus decision making and international endorsement and minimizing the dominance of specific proprietary interests. The resulting process can be cumbersome, but with effective leadership, surprisingly good standards with widespread acceptance can be produced in a short time. Such has been the case with MPEG.

The focus of this chapter is the completed work embodied in the MPEG-1 and -2 standards. As of this writing, MPEG continues to work to produce new standards in several important areas, including advanced audio compression (AAC), digital storage media command and control, and MPEG-4, work targeted at very low-bit-rate (less than 64-kbps) audio-video communication. These efforts are described briefly at the end of the chapter. This chapter is not an exhaustive restatement of the standards (which run to several hundred pages); rather, it describes the major methods by which MPEG achieves compression and organizes information, in order to illuminate the potential effectiveness and applications of the standard.

11.2 The MPEG Model

A key to understanding MPEG is understanding both the problems that MPEG set out to address in developing MPEG-1 and -2 (although it is likely MPEG will be applied in many unanticipated places as well) and the fundamental models that underlie the algorithms and that are used to foster interoperability.

11.2.1 Key Applications and Problems

Some of the most important applications to drive the development of MPEG include disk-storage-based multimedia, broadcast of digital video, switched digital video, high-definition television, and networked multimedia.

Disk Storage

MPEG-1 had its genesis as the solution to a very specific compression problem: how to best compress an audio-video source to fit into the data rate of a medium (CD-ROM) originally designed to handle uncompressed audio alone. At the time MPEG started, this was considered a difficult goal. Using an uncompressed video rate for 8-bit active video samples (CCIR-601 chroma sampling) of approximately 210 Mbps, this requires a rather aggressive 200:1 compression ratio to achieve the slightly greater than 1 Mbps or so available after forward error correction and compressed audio on a typical CD-ROM.

Aside from the substantial compression requirement, another requirement of many video on CD-ROM applications is a reasonable random access capability, that is, the ability to start displaying compressed material at any point in a sequence with predictable and small delay. This is a key attribute, for example, of many interactive game and training materials.

More recently, a new optical disk format with much higher capacity than CD-ROM has been developed. Called digital versatile disc (DVD) (renamed from the original "digital video disc" to include data storage applications), its higher rates, combined with the use of variable-rate encoding, can provide video quality that surpasses that currently available to consumers through other video media (e.g., VHS tape, laserdisc, cable and off-air analog television), as well as cinematic audio and a number of unique access and format features.

Broadcast of Digital Video

Electromagnetic spectrum is a scarce and valuable resource. It allows widespread and inexpensive broadcast capabilities: a single transmitting site can illuminate

a city, country, or continent, without the need for digging trenches or stringing wires. But electromagnetic spectrum is also key to many existing and planned services, especially mobile radio applications of various types. For this reason, the electromagnetic spectrum available for distribution of audio-video material is limited. This has created a strong economic motivation to develop compression technology for digital broadcast applications. Already, digital broadcast of compressed video over satellite has been deployed in several countries. Application to terrestrial broadcast (typically at VHF and UHF frequencies) and point-to-multipoint microwave will emerge in the very near future.

Broadcast digital video is typically introduced within bands originally sized for analog video and capable of carrying 20–40 Mbps of digital material. Since this is sufficient for multiple channels of compressed audio and video, packet-based multiplexing is used to allow carriage of multiple programs within a single digital signal. Also, digital broadcast often occurs over channels with a high noise level and potentially with phenomena such as fading. This leads to a higher attention to the behavior of decoders in the presence of errors and interruptions in the data stream.

Broadcast video can also be carried over wired networks, such as the typical "hybrid fiber coax" networks used in cable television. These networks consist of a backbone of high-capacity fiber distributing analog modulated signals to hubs, which further distribute the signals over copper coaxial cables to multiple homes.

Switched Digital Video

Switched digital video involves establishment of a dedicated communication path between the source of the video (a real-time encoder or a storage system) and a decoder. Although such systems often contemplate carrying signals over very high-capacity fiber plant, the sheer number of such signals as well as the load on switching gear makes compression attractive. As important is the existence of a strong and complete standard allowing deployment of compatible equipment from multiple vendors at dispersed geographical and functional locations in a network, as well as additional work to detail how to transport material encapsulated within typical telecommunications protocols.

In some cases, switched digital video networks use a special modem technology, called *asymmetric digital subscriber line* (ADSL), to deliver the signal digitally over the last stretch to the home using existing copper twisted pair installed to provide basic telephone service. This technology imposes its own bit rate limit, typically between 1.5 and 6 Mbps, depending on range and the exact type of modulation used.

HDTV

High-definition television (HDTV) promises a substantial increase in the detail available in a video transmission. When combined with multichannel high-fidelity audio transmission and a wider screen aspect ratio (16:9 as opposed to the 4:3 ratio of horizontal to vertical length in current television standards), HDTV promises a near-cinematic viewing experience. However, HDTV must be delivered in the same frequency bands originally allocated for standard television (although in many cases unused to avoid interference between close-by transmitters). Substantial compression reduces the spectrum required to be comparable to that of conventional, uncompressed analog television transmission, making the introduction of HDTV achievable without a radical reallocation of the VHF and UHF frequency bands.

Networked Multimedia

The advent of a widely available and endorsed standard for representation of digital video in compressed form helps developers realize multimedia applications without a dedicated local storage device (such as CD-ROM). Should MPEG decoding become widely included in personal computers and workstations, as appears likely, delivery of multimedia including MPEG-compressed material over different computer networks is possible, although many current networks present their own problems for time-sensitive data such as audio and video. Given the sometimes high utilization on computer networks for nonvideo applications, such as file transfers, aggressive compression is often a necessity for achieving more than a trivial number of sessions over a network. Networked multimedia applications stress the ability of a technology to deliver multiple different types of data streams—some synchronized and some not—and to coexist with (encapsulating or being encapsulated in) a variety of existing and emerging computer networking protocols.

11.2.2 Strategy for Standardization

A key purpose of a standard is to facilitate interoperability. A good standard achieves this while maximizing support for current and as-yet-unknown applications and the development of a variety of implementations. For MPEG, this is achieved by focusing standardization on two related questions:

- What is a legal MPEG bitstream?
- What should be done with it to generate displayable audio-video material?

The first question is answered by a careful specification of the legal syntax of an MPEG bitstream, that is, rules that must be followed in constructing a bitstream. The second is answered by explaining how an idealized decoder would process a legal bitstream to produce decoded audio and video. The standardization process applies the principles to yield the following definitions:

- A *legal bitstream* is one that does not violate any syntactical or semantic rule of MPEG, including being decodable on an idealized decoder.
- A *legal encoder* never produces an illegal bitstream.
- A *legal decoder* successfully decodes all legal bitstreams.
- A *good encoder* generates bitstreams that use less bits for a given perceived video and audio quality level when decoded on an idealized decoder.
- A *good decoder* produces good audio and video quality on the final display device and is robust in the presence of errors in the received bitstream.

This particular strategy for standardization allows a great deal of latitude in the design of encoding systems because the range between truly efficient bitstreams (in terms of quality preserved versus bits spent) and inefficient but legal bitstreams can be quite large. There is less latitude in decoder design, since the decoder must not deviate from the idealized decoder in terms of the bitstreams it can process, but there is still room for different and clever implementations designed to reduce cost in specific applications, improve robustness to slightly damaged bitstreams, and provide different levels of quality in post-MPEG processing designed to prepare a signal for display (e.g., interpolation, digital-to-analog conversion, and composite modulation).

The actual details of what constitutes a legal bitstream are largely conveyed through the specification of the syntax and semantics of such a bitstream.

Syntax and Semantics

Syntax is specified in MPEG using a pseudocode-style notation based on the C programming language. The syntax explains how the individual defined data elements may be combined to produce a legal bitstream. The data elements themselves have specified types and lengths, with the length being either fixed or variable. The syntax is further organized by allowing a syntactical statement to include "subroutines," which are in fact syntactical statements expanded elsewhere, as well as a small number of special "functions" that describe specific local properties required of the bitstream. An example of such a function is `next_start_code()`. This function specifies that the bitstream consists of only

```
Syntax                                           Number of Bits   Mnemonic
slice() {
        slice_start_code                         32               bslbf
        quantizer_scale                          5                uimsbf
        while (nextbits() == '1') {
            extra_bit_slice                      1                "1"
            extra_information_slice              8
        }
        extra_bit_slice                          1                "0"
        do {
            macroblock()
        } while (nextbits() != '000 0000 0000 0000 0000 0000')
        next_start_code()
}
```

FIGURE 11.1 Slice Syntax

0 bits and bytes of at least 23 total bits (but possibly more) followed by a 1 bit and then 8 bits defining the particular start code found.

As an example, Figure 11.1 shows the syntactical specification for the slice layer of MPEG-1 video (ISO/IEC 11172-2 1993 (E), Section 2.4.2.6) (the *slice* is the smallest collection of compressed video data at which resynchronization can occur in case a channel error has corrupted the decoding process). It defines a `slice` as starting with a particular 32-bit pattern, the `slice_start_code`, which is a `bit string left bit first` (`bslbf`). This is followed by a 5-bit `unsigned integer most significant bit first` (`uimsbf`), the `quantizer_scale`. If the next bit (`extra_bit_slice`) is a 1, another byte (`extra_information_slice`) follows. Indeed several such combinations of an `extra_bit_slice` set to 1 and an `extra_information_slice` byte may follow until finally an `extra_bit_slice` of 0 occurs. The function `nextbits()` used here observes for comparison purposes the next bits in the stream. Once we have allowed for the possible occurrence of `extra_information_slice` bytes, we then have a sequence of `macroblock()` until a new start code is observed (the final `nextbits()` condition recognizes the 23 0 bits that can only occur at the start of a start code or the 0 bit and byte stuffing that may precede it). The syntactical definition of `macroblock()` is further defined elsewhere in the standard, as are the semantics (meaning) of each of the boldfaced data elements. In summary, a slice consists

`slice_start_code`—The `slice_start_code` is a string of 32-bits. The first 24-bits have the value 000001 in hexadecimal and the last 8 bits are the `slice_vertical_position` having a value in the range 01 through AF hexadecimal inclusive. $AF = 174_{10}$

`slice_vertical_position`—This is given by the last eight bits of the `slice_start_code`. It is an unsigned integer giving the vertical position in macroblock units of the first macroblock in the slice. The `slice_vertical_position` of the first row of macroblocks is one. Some slices may have the same `slice_vertical_position`, since slices may start or finish anywhere. Note that the `slice_vertical_position` is constrained by 2.4.1 to define non-overlapping slices with no gaps between them. The maximum value of `slice_vertical_position` is 175.

Section of the standard

`quantizer_scale`—An unsigned integer in the range 1 to 31 used to scale the reconstruction level of the retrieved DCT coefficient levels. The decoder shall use this value until another `quantizer_scale` is encountered either at the slice layer or the macroblock layer. The value zero is forbidden.

`extra_bit_slice`—A bit indicates the presence of the following extra information. If `extra_bit_slice` is set to "1", `extra_information_slice` will follow it. If it is set to "0", there are no data following it.

`extra_information_slice`—Reserved.

FIGURE 11.2 Slice Semantics

of a unique start code, the possibility of some extra information bytes, and then a sequence of macroblocks, terminating with the next start code (which typically would be another slice start code or the start code that begins a picture, the collection of compressed video material from a single video frame or video field).

Corresponding to each syntactical definition is a semantical definition describing how a particular data element is to be interpreted by a decoder. For the slice layer in Figure 11.1, Figure 11.2 gives the semantical definitions (ISO/IEC 11172-2 1993(E), Section 2.4.3.5).

We see here that the MPEG committee is reserving itself the right in the future to extend the standard through the `extra_information_slice` bytes. This is in contrast to providing locations where a user can insert private information (defined by the user) while generating a legal syntax (which occur at several other places in the standard).

The definition of syntax and semantics as well as the tabular definition of various variable-length codes make up the bulk of the MPEG standards. In addition, there are some critical general constraints and requirements describing

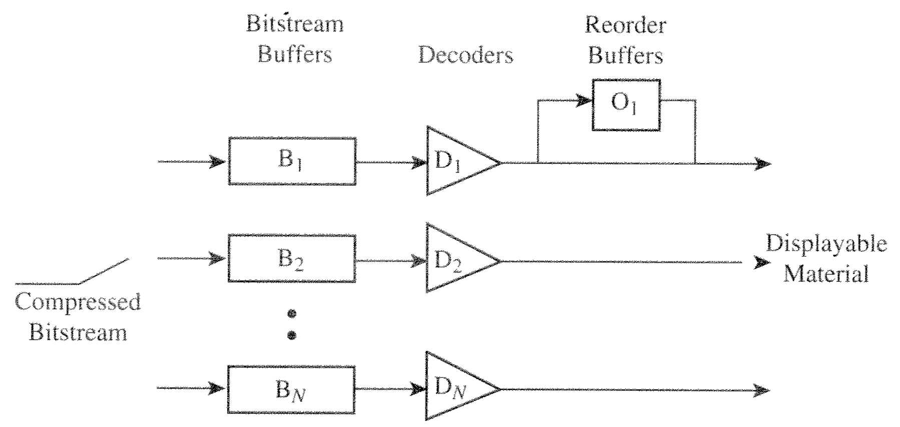

FIGURE 11.3 System Target Decoder

general timing and precision issues. A key such constraint is defined by the system target decoder.

The System Target Decoder

The *system target decoder* is a hypothetical target decoder used to provide precise definition for allowable synchronization and buffer states that can be produced by a legal bitstream (see Figure 11.3).

Conceptually, a bitstream to be decoded is demultiplexed as it arrives into elementary streams destined for various audio and video decoders, and placed in buffers $B_1, \ldots, B_N$ awaiting decoding. The decoders themselves, $D_1, \ldots, D_N$ have the idealized characteristic of instantaneously removing the bits required to decode a video or audio element at the time specified in the bitstream and decoding them ready for display. In some cases, the decoded material may wait in an additional buffer $O_1, \ldots, O_N$ until the display time, also encoded in the bitstream, occurs (such reordering buffers are often required for video decoding depending on the type of motion prediction allowed, but are not used in audio decoding).

The exact size required in the buffers $B_1, \ldots, B_N$ is specified in the bitstream (and limited by the standard). The system target decoder imposes the following key requirement on a legal bitstream: if the elements are decoded and displayed using the times given in the bitstream, the buffers $B_1, \ldots, B_N$ must never overflow or, except for the case of a special low-delay mode in MPEG-2, underflow. This requirement allows decoder designers to safely use a

specific amount of memory for these buffers. On the other hand, no real decoder can instantaneously remove all bits for a video or audio element and decode them. To the extent an actual decoder design varies from the system target decoder, it is up to the decoder system designer to insure that his or her design will be able to decode any bitstream that the system target decoder could decode.

11.2.3 Parts of the MPEG-1 and MPEG-2 Standards

The MPEG-1 and MPEG-2 standards are divided into several parts. In both cases, Part 1, "System," describes how various streams (video, audio, or generic data) are multiplexed and synchronized. Part 2, "Video," defines the video compression decoder, and Part 3, "Audio," defines the audio compression decoder. Part 4, "Conformance," defines a set of tests designed to aid in establishing that particular implementations conform to the design. Beyond these, MPEG is adding several new parts, discussed at the end of this chapter.

11.3 MPEG Video

The MPEG video algorithm is a highly refined version of a popular and effective class of video compression algorithms called motion-compensated discrete cosine transform (MC-DCT) algorithms. While not universally used for video compression, these algorithms have served as a basis for many proprietary and standard video compression solutions developed in the 1980s and 1990s. The algorithms use the same basic building blocks developed throughout this book and applied to many different signals, including

- temporal prediction—to exploit redundancy between video pictures
- frequency domain decomposition—the use of the DCT to decompose spatial blocks of image data in order to exploit statistical and perceptual spatial redundancy
- quantization—selective reduction in precision with which information is transmitted to reduce bit rate while minimizing loss of perceptual quality
- variable-length coding—to exploit statistical redundancy in the symbol sequence resulting from quantization as well as in various types of side information

These basic building blocks provide the bulk of the compression efficiency achieved by the MPEG video algorithm. They are enhanced, however, by a number of detailed special techniques designed to absolutely maximize efficiency and flexibility.

11.3.1 The Basic Algorithm

Figure 11.4 shows the basic encoding and decoding algorithms. The incoming video sequence is preprocessed (interpolated, filtered), then motion estimation is used to help form an effective predictor for the current picture from previously transmitted pictures. The motion vectors from motion estimation are sent as side information if used. The predictor for each block is subtracted, and the resulting prediction residual undergoes a DCT. The DCT coefficients are quantized, and the quantized coefficients are variable-length-coded for transmission. The quantized coefficients also undergo reconstruction, inverse DCT, and combination with the predictor, just as they will in the decoder, before forming reference pictures for future motion estimation and prediction.

The decoder decodes the variable-length codes, performs reconstruction of DCT coefficients, inverse DCT, formation of the predictor from previous reconstructed pictures, and summing to form the current reconstructed picture (which may itself serve to predict future received pictures). Postprocessing interpolates and filters the resulting video pictures for display.

— the relation [illegible] chroma subsampling

Representation of Video

The video that MPEG expects to process is composed of a sequence of frames or fields of luma and chroma.

FRAME-BASED REPRESENTATION. MPEG-1 is restricted to representing video as a sequence of frames. Each frame consists of three rectangular arrays of pixels, one for the luma (Y, black and white) component, and one each for the chroma (Cr and Cb, color difference) components. The luma and chroma definitions are taken from the CCIR-601 standard for representation of uncompressed digital video.

The chroma arrays in MPEG-1 are subsampled by a factor of two both vertically and horizontally relative to the luma array. While MPEG does not specify exactly how the subsampling is to be performed, it does make clear that the decoder will assume subsampling was designed so as to spatially locate the subsampled pixels according to Figure 11.5 and will design its interpolation of chroma samples accordingly.

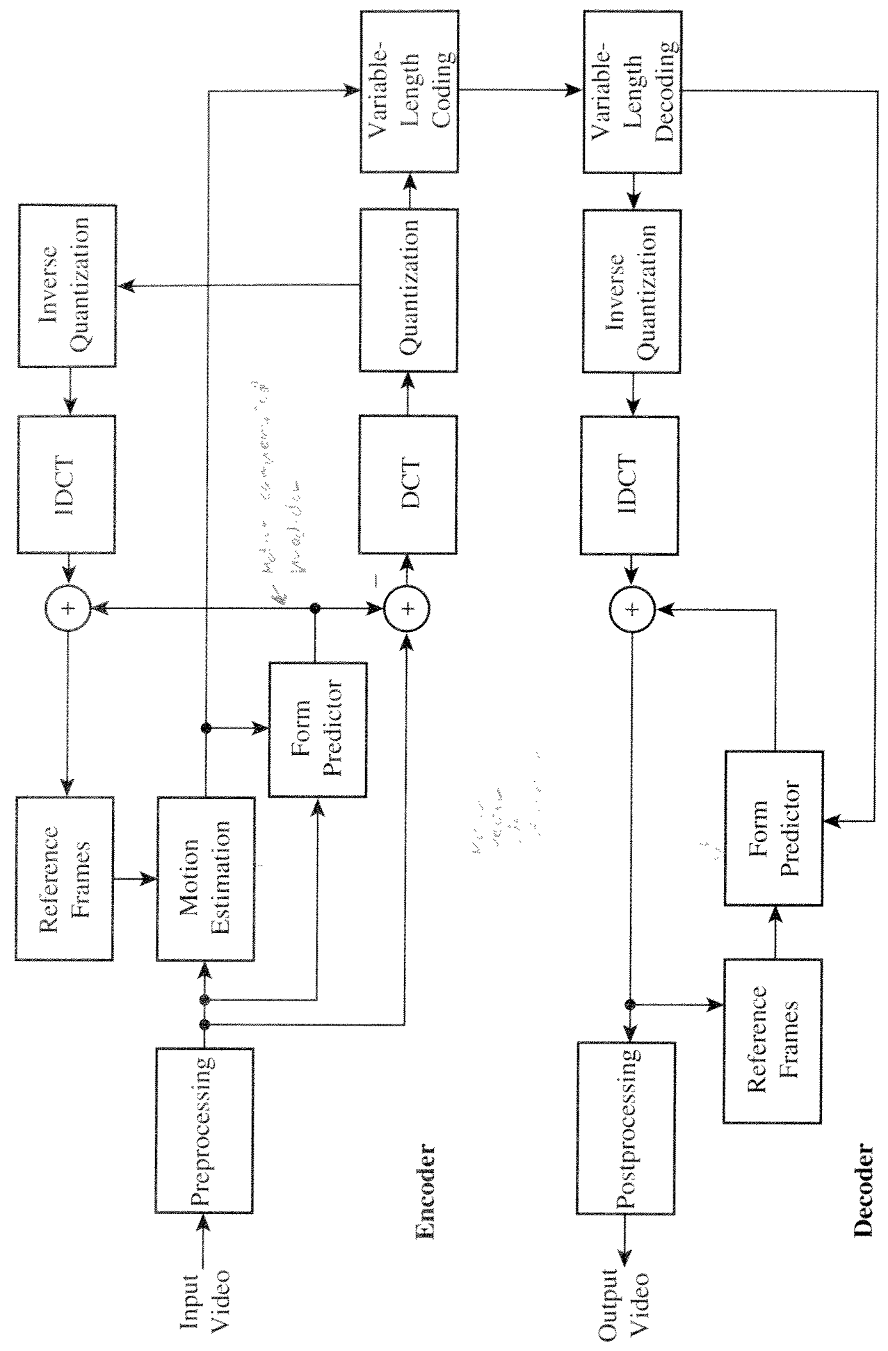

FIGURE 11.4 Block Diagram of MPEG Encoding and Decoding

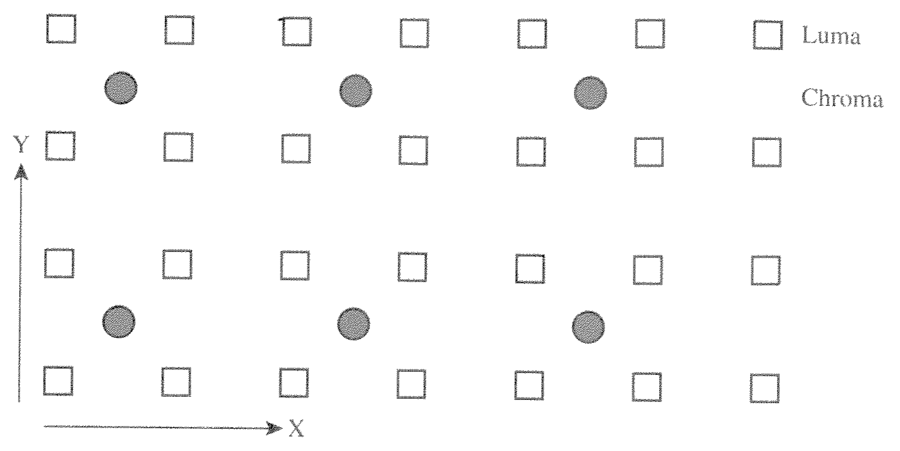

FIGURE 11.5 Relationship between Luma and Chroma Subsampling for MPEG-1

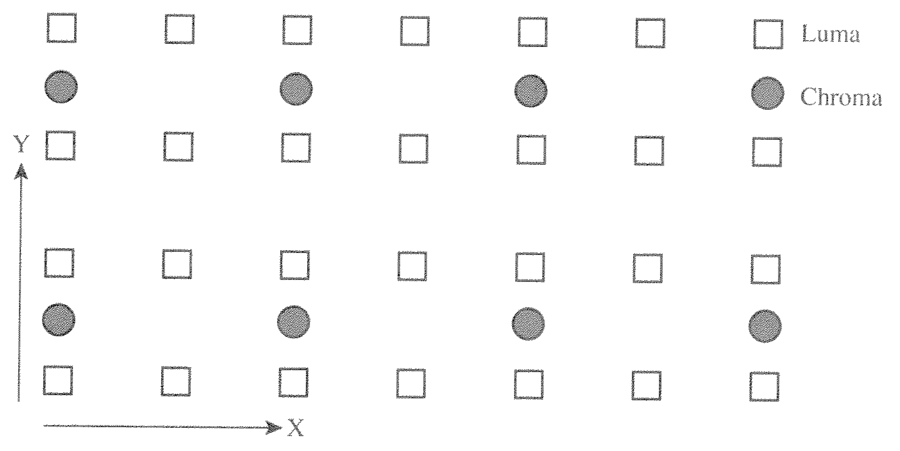

FIGURE 11.6 Relationship between Luma and Chroma Subsampling for MPEG-2

Typically, MPEG-2 expects chroma subsampling to be consistent with CCIR-601 prescribed horizontal subsampling. Spatially, this implies the chroma subsampling pattern shown in Figure 11.6, termed 4:2:0 sampling.

Field-Based Representation. MPEG-2 is optimized for a wider class of video representations, including, most importantly, field-based sequences. *Fields* are created by dividing each frame into a set of two interlaced fields, with odd lines from the frame belonging to one field and even lines to the other. The fields are transmitted in interlaced video one after the other, separated by half a frame time. Interlacing of video is in fact a simple form of compression by subsampling. It exploits the fact that the human visual system is least sensitive to scene content that has both high spatial and temporal frequencies (such as a fast-moving item with much detail). An interlaced source cannot represent such scenes effectively, but can build up the full detail of a frame (within two field times) and can also update low-resolution items that are changing every field time; these latter types of material are the most visually important.

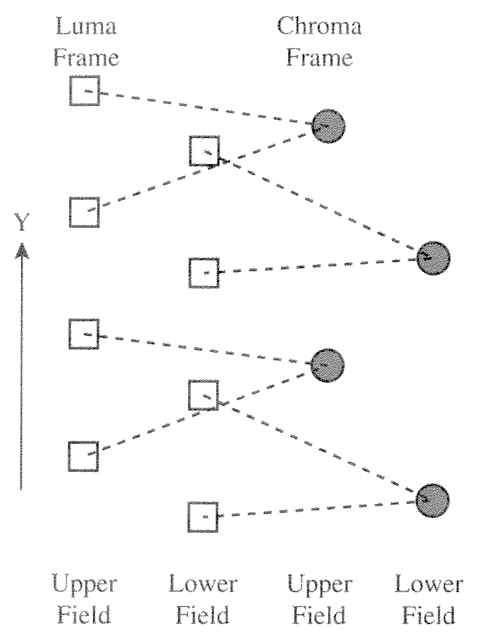

FIGURE 11.7 Relationship between Luma and Chroma Samples Vertically and in Time for Field-Based Video Material

Unfortunately, field-based representations are somewhat awkward for digital compression. Most contemporary compression scientists would prefer to work with a rectangular sampled array in both space and time and apply more sophisticated techniques than simple subsampling to achieve compression. However, a great deal of material has been, and will continue to be, produced in interlaced form, so efficient compression is necessary.

For field-based sequences, MPEG-2 expects the chroma associated with each field to be vertically subsampled within the field, yet maintain an expected alignment consistent with frame-based sequences. This leads to a vertical resampling pattern as shown in Figure 11.7.

11.3.2 Temporal Prediction

Temporal prediction exploits the intrinsic redundancy between video pictures. Often in video sequences, temporally adjacent pictures have a great deal of similarity. For instance, the background of a scene may be unchanged, with only disturbances in the foreground. In such a case, application of the classic differential pulse code modulation (DPCM) compression strategy (see Section 5.3) can provide substantial compression, with the prediction being applied to entire video pictures. The typical application of this strategy uses the last picture to predict the current picture and is called *picture differencing*.

We can go beyond picture differencing, however, by noting that when there are changes in the current picture, many times they are caused by the motion of objects. Another common occurrence is motion of the entire picture caused by panning of a camera. Such observations motivate the inclusion of a motion model in the prediction to further reduce the amount of spatial information that must be encoded as a prediction residual. Motion compensation, then, is the application of a motion model to prediction.

Temporal prediction with motion compensation is not universally effective, however. For instance, a scene change involves an abrupt discontinuity from one video picture to another. No form of temporal prediction can be particularly effective across such a discontinuity, and the compression algorithm must rely on other capabilities to adequately process such changes. Also, types of changes in the video scene that are not well modeled by the particular motion model chosen may reduce the efficiency of motion compensation. For instance, a motion model based on translation will be less effective in handling rotations or zooms.

Motion Compensation

The motion model used by MPEG is blockwise translation. For simplicity, the blocks are chosen to be a single fixed size: 16 × 16 pixels in the luma component. Since for typical 4:2:0 chroma subsampling, the chroma components are vertically subsampled by a factor of two in both dimensions relative to the luma, each chroma component includes an 8 × 8 block of pixels corresponding to the same spatial region as a 16 × 16 block from the luma picture. The collection of the 16 × 16 region from the luma and the two corresponding 8 × 8 regions from the chroma components is called a *macroblock*.

Motion compensation consists of taking a macroblock from the current picture and determining a spatial offset in the reference picture at which a good prediction of the current macroblock can be found. The offset is called

a *motion vector*. The mechanics of motion compensation are to use the vector to extract the predicting block from the reference picture, subtract it, and pass the difference on for further compression. The vector chosen for a macroblock is applied directly to determine the luma predictor, but is scaled by half in both dimensions (corresponding to their subsampled size) before being applied to find the chroma predictors.

Motion estimation consists of finding the best vector to be used in predicting the current macroblock. This is typically the most expensive activity in an MPEG encoder, but can have a substantial impact on compression efficiency. The straightforward approach to motion estimation is to evaluate each individual motion vector from the allowed range of possible vectors and select the best one. A reasonable criterion for which is best would be the number of bits consumed in coding the block using that specified offset. In practice, the criterion is often simplified to measuring the mean absolute error (MAE) between the current macroblock and the macroblock at the specified offset (often evaluated between the luma components only), and the search itself is structured in some manner to reduce the total number of comparisons to be made. The most common strategy for such motion estimation is a pyramid-based search. Pyramid searches rely on forming a collection of increasingly low-pass and subsampled versions of the image (a low-pass image pyramid) for the current and reference picture. The search begins by comparing regions between the lowest-resolution versions of the current and reference images; the result of this search initializes a local search at the next higher resolution, which again initializes a search at the next higher resolution, and so on. Such pyramidal searches may require many fewer comparisons since the local searches may be quite restricted. They are effective for most image sequence types, but can be less effective if the difference between a good and bad motion estimation match is restricted to detail that is not apparent in the lower resolution levels of the pyramid.

The effectiveness of motion compensation can be enhanced by allowing the search for an effective prediction region in the reference picture to include not only positions at integral pixel offsets but also fractional pixel offsets. For a fractional pixel offset, the predicting macroblock is constructed by linearly interpolating pixel values relative to the nearest actual pixels. An example of a $\frac{1}{2}$, $\frac{1}{2}$ pixel offset prediction block is shown in Figure 11.8. The dark circle locations have pixel values x calculated as

$$x = (a + b + c + d)/4 \tag{11.1}$$

where a, b, c, and d are the closest pixels in the original reference picture.

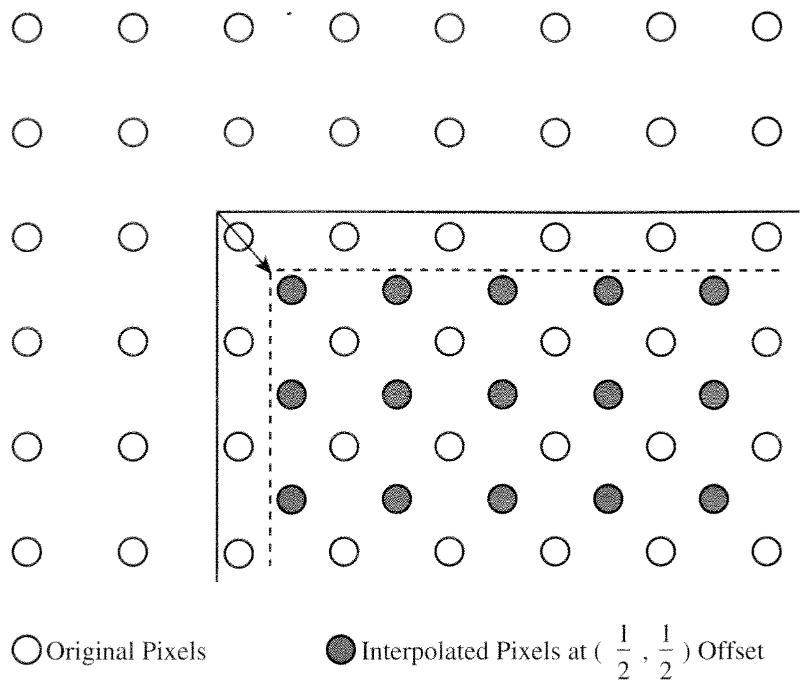

FIGURE 11.8 Example of Half-Pixel Interpolation in Motion Compensation

MPEG allows half-pixel interpolation vertically, horizontally, and both (the previous example being both). The crudeness of linear interpolation does not provide perfect motion tracking at higher resolution, but does bring the benefit of effectively low-pass filtering the image content at interpolated locations. This sometimes removes substantial noise from a reference image and provides a better predictor than might otherwise be expected.

This 32 × 32 region size motion compensation provides a reasonably good compromise between a calculus style approximation to arbitrary motion fields (which becomes more and more accurate as block size decreases) and the bit rate absorbed as overhead in transmitting motion vectors. It will not be effective, though, on complicated motion fields involving substantial rotation, zoom, or deformation. In such cases, as with scene cuts, we must rely on the lower sensitivity of the human visual system to distortion in such complicated material as well as the compression capabilities of other elements of the standard.

Picture and Macroblock Prediction Types

One of the key requirements of MPEG is reasonable support for random access. Random access into an arbitrary motion-compensated sequence is difficult for the same reason that starting decoding in a delta modulation or DPCM sequence is difficult: the received sequence consists of a lot of information about changes from one picture to another, but not necessarily the starting point or original reference. This is a dangerous situation in any case from the point of view of dealing with transmission imperfections; the effect of such imperfections on the decoded images can persist indefinitely. These are usually managed through one of two main strategies: refresh and leaky prediction.

Refresh involves periodically sending an entire picture (picture-based refresh) or portion of a picture (region-based refresh) without any prediction, allowing the decoder to resynchronize at that point. Region-based refresh is achieved by guaranteeing a particular subset of macroblocks are encoded without prediction and rotating this subset through subsequent pictures used in prediction until all macroblocks have been forced to be encoded without prediction. This mechanism reduces the burst of bits required for a picture compressed entirely without prediction, but does not provide a guaranteed random access entry point nor a deterministic time to complete refresh (since nonrefreshed material may be shifted into a just refreshed area by motion compensation before the refresh cycle is complete).

Leaky prediction involves slightly reducing the effectiveness of the prediction in order that the decoder will progressively "forget" any perturbations in the decoded sequence caused by transmission errors. Both encoder and decoder predictors are multiplied by an agreed upon leak factor less than one. Since this reduces the accuracy of the predictor, it causes an increase in the bits required to transmit the compressed residual. However, any discrepancy between decoder and encoder predictor memories is reduced during each iteration of prediction, resulting in an exponential decay in the difference.

Only picture-based refresh aids in random access, however, as it allows a predictable location to begin decoding and a bounded time to decode to any target picture. For instance, if we want to decode picture 303 and we know every 10th picture is not predicted starting at picture 0, then we can decode picture 300, followed by picture 301, 302, and 303 (all of which may consist mainly of differences relative to the previous picture). We will never need to decode more than 10 pictures to get to any desired picture, and we know where to start decoding. To facilitate this strategy, MPEG defines an *I-picture* as being composed entirely of macroblocks compressed without the use of temporal prediction.

TABLE 11.1 Allowed Prediction Modes by Macroblock Type

Macroblock Type	Prediction
Nonpredicted macroblock	none
Backward-predicted macroblock	references temporally nearest subsequent anchor picture
Forward-predicted macroblock	references temporally nearest previous anchor picture
Bidirectionally predicted macroblock	averages predictions from temporally nearest previous and subsequent anchor pictures

[illegible]

TABLE 11.2 Allowed Macroblock Types and Anchor Picture Definition by Picture Type

Picture Type	Anchor Picture	Macroblock Types
I-picture	yes	nonpredicted
P-picture	yes	nonpredicted forward predicted
B-picture	no	nonpredicted forward predicted backward predicted bidirectionally predicted

[illegible]

Given the likelihood that streams would be generated with the regular use of nonpredicted pictures, the MPEG committee decided to investigate some new prediction strategies that might further improve compression efficiency and developed the concept of forward and backward prediction. We first define an *anchor picture* as one that can be used for prediction. Then we define four different prediction strategies that can be used for a particular macroblock, as shown in Table 11.1. Each picture has a declared type that limits the type of macroblock allowed in that picture, as shown in Table 11.2.

Figure 11.9 uses arrows to show which pictures can be referenced for motion estimation in a typical sequence of pictures. In fact, while not required by the standard, the set of sequences in which there is a fixed spacing between I-pictures and between anchor pictures (I- and P-pictures) is widely used, so widely that a pair of parameters are commonly used to describe these spacings: N is the number of pictures from one I-picture (inclusive) to the next (exclusive), and M is the number of pictures from one anchor picture (inclusive) to the next

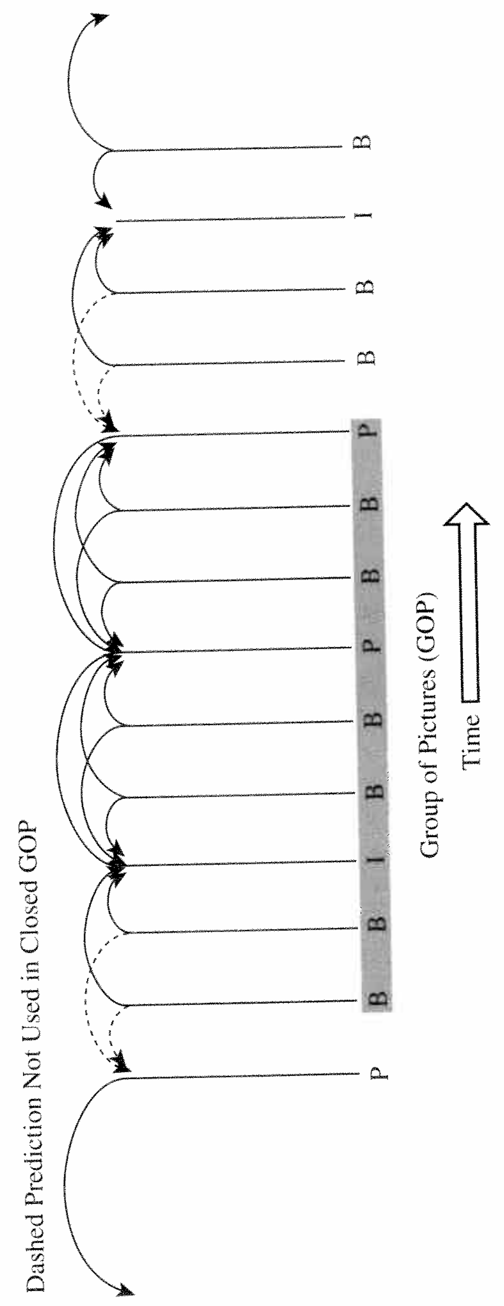

FIGURE 11.9 Typical Group of Pictures Collection of Pictures and Allowed Predictors for Pictures

(exclusive). So, the pattern shown in Figure 11.9 would be called an $N = 9$, $M = 3$ pattern.

There are several implications to this set of possible predictions:

- Using bidirectionally predicted blocks allows effective prediction of uncovered background, areas of the current picture that were not visible in the past but are visible in the future.
- Bidirectional prediction can provide for interpolation equivalent to an even finer degree than the half-pixel interpolation for motion compensation already allowed. Imagine an object located at offset 0 in a temporally adjacent preceding anchor picture and at offset $\frac{1}{2}$ in a temporally adjacent subsequent anchor picture. By using bidirectional prediction, we can provide the equivalent to a $\frac{1}{4}$-pixel linear interpolation of the object.
- Bidirectional prediction can reduce noise in the predictor. If a good prediction is available in both preceding and subsequent anchor pictures, then averaging the two predictors reduces noise and hence increases prediction efficiency.
- Bidirectional prediction increases motion estimation complexity in two ways: first, it obviously requires twice as much work to perform motion estimation in two different anchor pictures. More significantly, though, since anchor pictures are now possibly separated by several picture times, we need a larger overall motion estimation range in order to track objects of a given velocity. For instance, if an object is moving 10 pixels per picture, then a motion estimation range of 10 pixels is adequate to find a good predictor for the object in the absence of B-pictures. With B-pictures, however, the required range to track the object increases to $10M$, where M is the distance from one anchor picture (inclusive) to the next (exclusive).
- Since B-pictures cannot themselves be used to predict other pictures, the quality of compression is solely determined by the visual acceptability of the result. For P- and I-pictures, since the reconstruction will also be used for other predictions, quality must reflect both perceptual fidelity and prediction efficiency. This allows a substantial reduction in the bits allocated to B-pictures relative to I- and P-pictures. A ratio of 5:3:1 in bits spent on I-, P-, and B-pictures is not uncommon. These ratios can and should vary dynamically, though. For instance, an unchanging scene would tend to put almost all the bits in the I-pictures (since the others would be extremely predictable); a scene that changes so thoroughly as to render motion estimation ineffective would tend to spread bits out evenly between all three picture types (indeed, almost all blocks in all pictures would be nonpredicted). Also, since B-pictures are not used as predictors, bit errors in a received bitstream that occur within a B-picture can often have their effects limited to that picture, reducing error propagation.

Picture Time	−3	−2	−1	0	1	2	3	4	5	6	7	8	9
Encoder Order	P_{-3}	B_{-2}	B_{-1}	I_0	B_1	B_2	P_3	B_4	B_5	P_6	B_7	B_8	I_9
Transmit Order				I_0	B_{-2}	B_{-1}	P_3	B_1	B_2	P_6	B_4	B_5	I_9
Decoder Memory 1			. . .	P_{-3}	P_{-3}	P_{-3}	P_{-3}	P_3	P_3	P_3	P_3	P_3	I_9
Decoder Memory 2			. . .	I_0	I_0	I_0	I_0	I_0	I_0	P_6	P_6	P_6	P_6
Decoder Scratch Memory			. . .	—	B_{-2}	B_{-1}	—	B_1	B_2	—	B_4	B_5	—
Displayed Picture			. . .	P_{-3}	B_{-2}	B_{-1}	I_0	B_1	B_2	P_3	B_4	B_5	P_6

FIGURE 11.10 Pictures in Encode and Transmit Order and Decoder Memory Usage

Display and Transmit Order

Combining the use of B-pictures with a desire to economize on the use of memory in high-volume decoders leads to a difference in the order in which pictures are transmitted versus the order in which they are displayed. The reordering is intended to ensure that, regardless of the spacing between anchor pictures, a decoder need only have enough memory to store two anchor pictures and the scratch memory for the picture currently being decoded. This is achieved by transmitting the anchor picture's compressed version before those of the B-pictures that will precede it in display. Hence the anchor picture will arrive ahead of the time it is needed and be available for reconstructing the B-pictures for immediate display. An example is shown in Figure 11.10. The boxed area shows a single Group of Pictures (GOP), defined as a contiguous sequence of pictures beginning with an I-picture (inclusive) and ending with an I-picture (exclusive) in transmit order.

Although the decoder escapes the requirement for additional buffers as the spacing between anchor pictures increases, the encoder does not: the brunt of providing memory for reordering falls on it.

The first set of B-pictures following the I-picture that starts a GOP will in general require the last anchor picture from the previous GOP in order to decode (the examples are B_{-1} and B_{-2} in Figure 11.10). So if we try to start decoding from the beginning of a GOP, although we can decode and display the I-picture, we can't decode and display the next set of B-pictures and must instead wait until the next anchor picture to be able to decode and display continuously. MPEG allows the specification of a closed GOP, which requires that the first B-pictures following an I-picture in transmit order cannot use motion references to the last anchor picture in the previous GOP (e.g., B_{-1} and B_{-2} cannot use P_{-3} as a reference). In this case, display can begin as soon as B_{-2} is decoded.

The actual memory required in a specific decoder implementation can vary depending on how tightly the decoder synchronizes decoding and display processes. For instance, by advocating a "display while decoding" strategy yet dealing with interlaced source, it is possible to drive the total memory required from 3 frames to 2.75 or even below.

Field and Frame Prediction

While MPEG-1 does not consider the possibility of field-based video representations, MPEG-2 specifically does allow for increased compression efficiency on interlaced material. One way in which this is accommodated is by allowing either field- or frame-based prediction for each macroblock (often termed *adaptive field/frame compression*). In the case of field-based prediction, the macroblock is divided into two field macroblocks. The top field macroblock is predicted from the top fields in one or two anchor pictures using the modes appropriate for the picture type. The lower field macroblock is predicted from the lower fields in one or two anchor pictures. Note that the subdivision of chroma here corresponds to the chroma subsampling constraints for field representation described previously.

Field-based prediction can be highly effective in instances of substantial horizontal acceleration. In these cases, an object captured in two different fields appears to have [illegible] serrated vertical edges when viewed as a frame, with the depth of the serration varying with time due to acceleration. Frame-based motion estimation is quite ineffective in this case and leaves a good deal of difficult-to-compress high-frequency serrations left over. Field-based prediction can perform quite well. On the other hand, for more benign motion cases, frame-based prediction can more effectively exploit vertical redundancy. Allowing a choice between the modes to occur on a macroblock basis gives the best of both worlds at the cost of a slight increase in overhead (the net effect being in favor of allowing the choice). This type of prediction is most important on some of the most difficult-to-compress material, such as a basketball game in which a fast-moving foreground is combined with a highly detailed slow-moving background. The ability of adaptive field/frame motion compensation to improve some of this worst-case material makes it even more valuable than the average improvement alone would imply.

A special version of field-based motion estimation, *dual prime*, is also available. Dual prime can only be used in P-pictures in the absence of any B-pictures between the reference picture and the current picture. Each field in the current macroblock is predicted using the average of field macroblocks from each field in the preceding picture. Rather than sending four separate vectors in order to identify the offset between each of the current field macroblocks and their predictors in each of the previous fields, a single vector is sent and extrapolated to calculate the four vectors using a constant-velocity model. Let v_{ij} be the vector from field i in the current picture to field j in the reference picture, and let v be the transmitted vector. Then we have

$$v_{11} = v_{22} = v \tag{11.2}$$

$$v_{21} = \left(\frac{3}{2}\right) v \tag{11.3}$$

$$v_{12} = \left(\frac{1}{2}\right) v \tag{11.4}$$

These formulas can be understood from the constant-velocity model, since if a region being tracked was translating with constant velocity, the displacement between fields 1 and 1 in the current and reference picture and between fields 2 and 2 would be identical; the displacement between field 2 in the current picture and field 1 in the reference picture would be $\frac{3}{2}$ as large (since the temporal distance is $\frac{3}{2}$ as large as between fields 1 and 1 or 2 and 2) and between field 1 in the current picture and field 2 in the reference picture would be $\frac{1}{2}$ as large (since the temporal difference is $\frac{1}{2}$ as large). An additional ± 1 pixel offset is available to make minor corrections in the model. Dual prime achieves some of the gains of B-pictures through the averaging effect between fields, which can reduce noise and synthesize a finer degree of motion compensation offset resolution.

11.3.3 Frequency Domain Decomposition

After motion compensation has been performed, the residual information (or original picture information in the case of a nonpredicted block) is transformed using the DCT. To prepare the macroblock for transformation, six 8×8 blocks are extracted from the macroblock, four from the luma picture and one each from each of the chroma pictures. The luma blocks for MPEG-1 and the frame DCT mode of MPEG-2 are shown in Figure 11.11; for the field DCT mode of MPEG-2, in Figure 11.12.

The field DCT mode is useful for much the same reason that field motion compensation is useful: material with a high degree of horizontal motion, especially

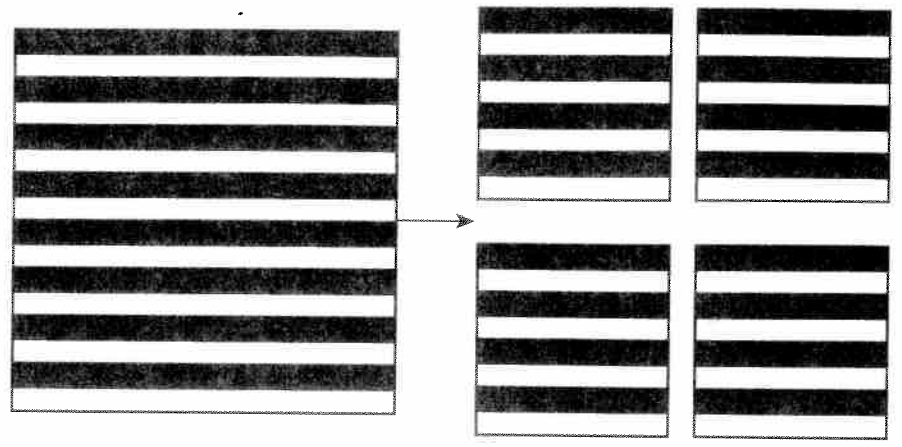

FIGURE 11.11 Mapping from 16 × 16 Block to 8 × 8 Blocks for Frame-Organized Data

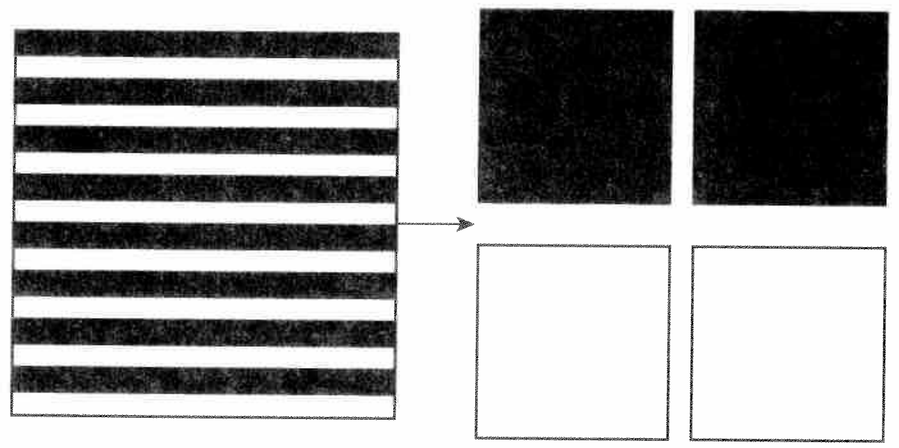

FIGURE 11.12 Mapping from 16 × 16 Block to 8 × 8 Blocks for Field-Organized Data

acceleration, captured using an interlaced video standard, tends to produce difficult-to-compress high frequencies when transformed using a frame-based DCT. The choice of frame versus field mode for motion compensation and DCT are not bound together, and there is a small benefit to be had by making the decision independently.

Discrete Cosine Transform

The MPEG standard does not specify exactly how to perform the 8 × 8 DCT and inverse discrete cosine transform (IDCT) required. This is too limiting given that many different software and hardware implementations of MPEG are expected. Instead, MPEG calls for the IDCT as implemented to be able to pass a test developed as part of the ITU-T standardization of H.261, a videoconferencing standard. In this test, IEEE P1180/D2, the proposed IDCT inverse transforms a particular set of blocks, and the results are compared to an idealized set of results. Providing the proposed IDCT has less than a specified number of discrepancies at each of several magnitudes, it is an acceptable IDCT.

Allowing similar but not identical IDCTs in different implementations will cause mismatch in the encoders and decoders of different manufacturers. Since the encoder includes an IDCT for the purpose of reproducing the pictures that the decoder generates, the fact that an encoder and decoder have slightly different DCTs will introduce a gradually increasing difference between these pictures, resulting in inappropriate coding decisions on the encoder's part. This phenomenon is managed by limiting to 132 the number of times a specific macroblock location may be coded as predicted in P-pictures before that macroblock location must be coded without prediction. Moreover, MPEG-2 added some specific quantization rules that were found to further mitigate the problem of mismatch.

11.3.4 Quantization

After transformation, the DCT coefficients typically have 12 bits or more of precision. On the face of it, this is not a very auspicious compression, since the data started with 8 bits per pixel. Quantization is the key step that exploits the efforts provided by motion compensation and the DCT to reduce the bits required since the information has now been organized so that many coefficients are essentially irrelevant to the final reproduction quality and only a few need be treated with care.

Quantizer Step Size

MPEG applies a uniform step size quantizer to each coefficient. However, the step size of the quantizer may vary from coefficient to coefficient and macroblock to macroblock. In fact, the quantizer step size is determined at the decoder by the following equation:

$$ss = qf[m, n] \times qs \tag{11.5}$$

*	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

Kvant på
verdi i [illegible]

FIGURE 11.13 Intraframe Quantizer Weighting Matrix $qf[m,n]$

16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16

Kvant av
residue etter prediksjon

FIGURE 11.14 Interframe Quantizer Weighting Matrix

The factor $qf[m, n]$ is dependent on the location of the coefficient within a block. The factor qs is the base quantizer step size. This allows emphasis on more perceptually important lower frequencies. MPEG provides two default weighting matrices $qf[m, n]$ for use on predicted blocks and nonpredicted blocks, shown in Figures 11.13 and 11.14. Note that for nonpredicted blocks, the DCT DC coefficient, the upper left-hand coefficient that is proportional to the average value of the space domain block, is not quantized using a weighting matrix value; hence that location in the weighting matrix has an * in Figure 11.13. This unusual treatment reflects the fact that the eye is highly sensitive to errors in the DC level of nonpredicted blocks and tends to quickly recognize these as blocking or tiling distortion in the reconstructed picture.

The default matrix for nonpredicted blocks shows a strong bias towards lower frequencies, consistent with other DCT-based algorithms applied to nonpredicted material, such as JPEG (Chapter 9). However, the default matrix for predicted blocks is flat. In fact, it has been found empirically that since motion

compensation itself tends to remove much of the low frequencies in any case, there is little benefit at the quantization stage to further emphasizing them.

MPEG allows encoders to override the default matrices (in the video sequence header, a header inserted at least at the beginning of the sequence and then periodically at the beginning of pictures at the discretion of the encoder); many contemporary encoders do this.

While the manner in which the decoder will map quantized values to reconstructions is precisely prescribed, the choice as to how to map the incoming values at the encoder to quantized values is not. A typical strategy for optimizing the encoder quantization is to increase the absolute threshold that bounds the range that will be mapped to the quantized value 0. This is because the coding strategy described in the next section is very efficient at compressing long runs of 0 quantized values, and the slight increase in distortion suffered by the more inaccurate values is more than offset by the savings in bits, a savings that can be applied to allow finer quantization in more critical areas. Further, for interframe material in particular, low-level random noise is also effectively reduced by a wider range for the 0 quantized value.

11.3.5 Variable-Length Coding

Both the quantized coefficients and several different types of side information (macroblock prediction type, motion vectors, etc.) exhibit statistical concentration so that the overall average bit rate can be lowered by using variable-length coding. Note that the use of variable-length coding will necessitate buffering when MPEG is to operate over a fixed-rate channel.

Modified Huffman Codes

The type of variable-length code used throughout MPEG is a modified Huffman code. Huffman coding, discussed in Chapter 2, provides optimal variable-length coding for the chosen alphabet. However, for large alphabets, storing, encoding, and decoding the Huffman code can be expensive. Hence, the table size is generally restricted to a subset consisting of the most probable symbols. An additional codeword is added to the code, termed "ESCAPE." When an input symbol is observed that does not belong to the chosen high-probability symbol set, it does not have its own Huffman codeword; instead, the ESCAPE codeword is issued followed by the explicit (i.e., standard indexed value) of the input symbol. The use of an ESCAPE strategy does increase the average bit rate of the code, but if the total probability of all symbols that will use the ESCAPE is sufficiently small, the impact is negligible. A second modification to the MPEG

TABLE 11.3 Example of a Simple Modified Huffman Code

Symbol	Code
0	01
1	10
2	110
3	111 011
4	111 100
5	111 101
6	111 110
7	111 111

[illegible]

Huffman codes for video is that the all-zero codeword is not used, in order to avoid emulation of a start code via a legitimate sequence of other codes. Since the start code begins with 23 zeros and is used as a resynchronization point when decoding an errored bitstream, the occurrence of 23 zeros as a sequence of other codewords could cause a spurious resynchronization with disastrous effect. Table 11.3 shows an example of a simple modified Huffman code using these rules for an alphabet consisting of {0, 1, 2, 3, 4, 5, 6, 7}. The symbols 3–7 are escaped; the codeword of all zeros is not allowed.

Two-Dimension to One-Dimension Conversion

The quantized DCT coefficients form the bulk of the material that needs to be variable-length encoded. Empirically, though, after quantization, the 8 × 8 blocks of DCT coefficients tend to exhibit a substantial number of zeros, particularly in the higher frequencies, as in Figure 11.15.

MPEG exploits this phenomenon by first converting the 8 × 8 array into a one-dimensional array and then applying coding that can exploit long runs of zeros. Both MPEG-1 and MPEG-2 use a zigzag scan for cases, such as in Figure 11.15, where there are a large number of zeros in the lower right-hand quadrant. MPEG-2 provides an alternative, vertical scan, which is often more effective, especially when a field-based DCT has been applied. The scan path of zigzag and vertical scans is shown in Figure 11.16.

Zigzag scanning applied to the example array results in the sequence:

$$-42\ 12\ -19\ 16\ 14\ 3\ -1\ -8\ 8\ 3\ 3\ -2\ -9\ 0\ 0\ 0\ 2\ 3\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0$$
$$0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ -1\ 0\ \ldots\ 0$$

	Increasing Horizontal Frequency →							
	−42	12	3	−1	0	0	0	1
Increasing	−19	14	−8	0	2	0	0	0
Vertical	16	8	−9	3	0	0	0	0
Frequency	3	−2	0	0	1	0	0	0
↓	3	0	0	1	−1	0	0	0
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0

FIGURE 11.15 Example of an 8 × 8 Array of Quantized DCT Coefficients

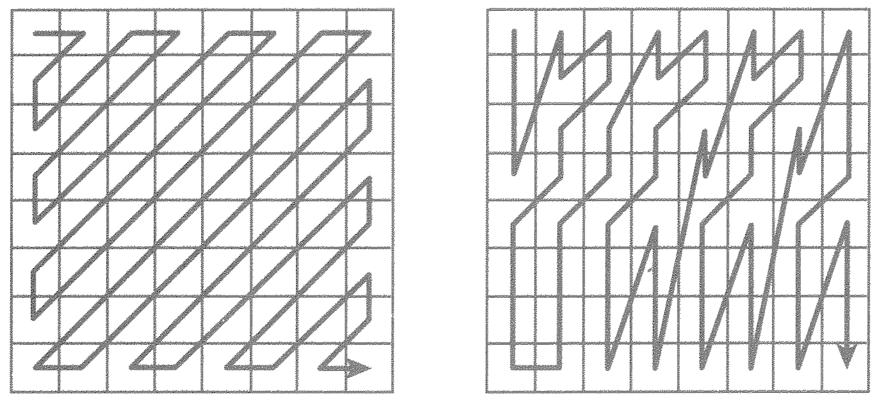

FIGURE 11.16 Coefficient Scan Pattern for Zigzag and Vertical Scans

The decoder will perform the inverse scanning as part of its reconstruction process.

Runlength Amplitude Coding

Once a 64-symbol vector has been created by scanning the 8 × 8 array for each block, runlength amplitude coding is used. First, the DC quantized coefficient (containing the block average) is treated specially, receiving its own dedicated Huffman code, because the DC quantized coefficient tends to have a unique statistical characteristic relative to other quantized coefficients. Moreover, since there is some redundancy between adjacent DC quantized coefficients in non-predicted blocks, only the difference between these is Huffman coded. The remaining quantized coefficients are parsed into a sequence of runs, where a run is defined as zero or more 0s followed by a single nonzero value. This becomes a new symbol for Huffman coding, consisting of a runlength-amplitude pair. The parsing into runlength-amplitude pairs for the example of the last section is

(0, −42), (0, 12), (0, −19), (0, 16), (0, 14), (0, 3), (0, −1), (0, −8), (0, 8),
(0, 3), (0, 3), (0, −2), (0, −9), (3, 2), (0, 3), (10, 1), (2, 1), (0, 1), (6, −1),
(EOB)

Note that a special codeword is reserved to indicate nothing but 0 quantized coefficients from that point until the end of the block (EOB).

MPEG-1 and MPEG-2 share a modified Huffman code for runlength-amplitude pairs. MPEG-2 also provides an additional modified Huffman code that has been specifically optimized and found to provide a slight improvement on coding nonpredicted (or intraframe blocks), called the *alternative intra Huffman code*.

Side Information Coding

There are a variety of different types of side information, designed in general to reduce the number of blocks that actually have to undergo runlength-amplitude coding and to reduce the amount of information in those that do. The most important of these are summarized below.

MACROBLOCK ADDRESS. MPEG includes a mechanism for skipping macroblocks if the decoder should reconstruct the contents of the macroblock simply using the collocated macroblock in another picture. This is enabled by starting each macroblock with the increment with which the macroblock address should be increased; if the increment is greater than one, the macroblock is skipped.

MACROBLOCK TYPE. The macroblock type indicates whether a macroblock is not predicted, forward predicted or backward predicted, or both; whether the quantizer step size should change; whether (in MPEG-2) the macroblock is field or frame motion compensated and field or frame transformed; and whether all the blocks within the macroblock are coded.

CODED PATTERN. Just as with skipped macroblocks, not all of the blocks within a macroblock may have significant enough information after motion compensation, DCT, and quantization to be worth coding. The coded pattern code allows the encoder to select and signal one of several patterns indicating which blocks are coded and which are not. Uncoded blocks are reconstructed using the predictor only.

MOTION VECTORS. In the case of a motion-compensated macroblock, motion vectors are sent. Motion vectors are themselves coded using prediction: a set of rules allows construction of a predictor for the current motion vector from the last transmitted motion, and only the difference is actually transmitted. The motion vector prediction loop is restarted at the beginning of a slice and a few other conditions.

11.3.6 Syntactical Layering in MPEG

The MPEG syntax includes several distinct layers, many of which we have already discussed:

- Sequence: The sequence layer defines picture size, picture rate, expected buffer sizes, and a number of other near static parameters. It also provides an opportunity to define nondefault quantizer matrices for MPEG-1 (this can happen every picture in MPEG-2).
- Group of Pictures: A contiguous set of pictures in transmit order that aid random access. The first picture in transmit order in the GOP must be an I-picture. The GOP header can indicate whether it is open or closed, whether it has been separated from the original preceding GOP (`broken_link`), and SMPTE time code for the start of GOP. The GOP structure is mandatory in MPEG-1 and optional in MPEG-2. [illegible]
- Picture: The picture header defines picture type (I, B, or P) as well as motion vector range in that picture.
- Slice: The unit of resynchronization. A collection of macroblocks preceded by a unique resynchronization pattern (a start code).
- Macroblock: The unit of motion compensation. A 16 × 16 region in the luma combined with the corresponding 8 × 8 regions in both chroma components. The macroblock header gives the prediction mode for the block, macroblock address, coded block pattern, and, if desired, can cause an overall change in quantizer step size through the base quantizer step size variable.
- Block: The unit of transformation. An 8 × 8 collection of pixels. ([illegible])

SMPTE Society of Motion Picture and Television Engineers

11.3.7 Rate Control

One of the most important encoder design issues is how to include an effective rate control system into a compressor. As mentioned earlier, once variable-length coding has been applied, some buffering is required to absorb instantaneous fluctuations in bit rate if the stream is to be delivered over a fixed-rate channel. Such buffering must be available at both encoder and decoder. A tenet of MPEG is that once the decoder buffer size has been specified, it must be possible to decode a legal bitstream in that buffer without overflowing or underflowing (except for the low-delay mode of MPEG-2, which does allow underflows).

Fixed-Rate Channels

For fixed-rate channels, with a little effort, the requirement on the decoder buffer can be made equivalent to a requirement on the encoder buffer as follows:

- If the encoder and decoder buffer are of the same size,
- and the encoder compresses units at the pace at which they arrive (i.e., each unit of information is compressed in a period that is the inverse of the number of units per second),
- and the decoder decompresses units at the pace at which they are to be displayed,
- and the decoding starts when the decoder buffer's fullness equals the buffer size minus the encoder's buffer fullness,
- and the encoder ensures that the encoder buffer never overflows (by throttling information when necessary) and never underflows (by using stuffing to send filler bits when necessary),
- then the decoder buffer will never underflow or overflow.

More specifically, the decoder buffer will persist in "mirror state" relative to the encoder buffer: its fullness will always be equal to the buffer size minus the encoder buffer fullness at a time earlier by the channel rate divided by the buffer size. Hence, a strategy for fixed-rate channels is to ensure that an encoder buffer never underflows or overflows.

Variable-Rate Channels

MPEG supports variable-rate channels. The overriding constraint is that the decoder's buffer not overflow or underflow during decoding. Variable-rate channels come in two major classes: slave and master. A slave variable-rate channel has the rate selected by the MPEG system. A master variable-rate channel tells the MPEG system what the current rate is. Variable-rate channels are often also constrained by a maximum and minimum allowable instantaneous bit rate. In the very special case of a slave variable-rate channel with no limit on maximum instantaneous bit rate and a zero minimum instantaneous bit rate, the MPEG system can effectively ignore the presence of a buffer at the decoder, and encoding can proceed with the aim of minimizing the long-term average bit rate without regard to instantaneous fluctuations (with the exception that no picture could exceed in total the entire buffer size since a picture must be decoded instantaneously in the STD model). All other variable-rate channels require the encoding to take into account the state of the decoder's buffer. In many cases, a generalized version of the theorem stated in the previous section can be applied; in some, though, it may be necessary to directly model the decoder's buffer.

Storage systems can often be used as slave variable-rate channels with specific minimum and maximum instantaneous bit rates. For instance, an MPEG-2 laser-disk-based system with a minimum transfer rate of 2 Mbps and a maximum of 8 Mbps could provide good handling of much difficult material, but could also be expected to achieve generous total stored time for most entertainment source material, which frequently varies in overall source difficulty (e.g., for basketball, the commentators versus the game).

A special case of a master variable-rate channel is statistical multiplexing. This technique considers a group of channels that are encoded together. The total bit rate available to all channels is dynamically allocated to each of the channels in order to maximize the minimum quality across the set. An external algorithm must perform the rate allocation, and although each encoder will attempt to obtain the bit rate it deems necessary, there is no guarantee it will receive it. Statistical multiplexing can be quite effective in improving average quality, even on a small number of video channels; however, it is difficult to mix and match precompressed channels from multiple statistically multiplexed streams because there is no guarantee that a new combination of channels will meet any fixed bit rate target (other than the sum of the maximum possible bit rate of all streams).

Rate Control Strategies

In most cases, for fixed- or variable-rate channels, it is necessary to (implicitly or explicitly) track the decoder buffer state and take appropriate action to avoid overflows or underflows. For fixed-rate channels, using the relationship between encoder and decoder buffer states, it is sufficient to ensure that an equivalent-size encoder buffer never underflows or overflows. Avoiding underflow is achieved by using stuffing, a way to continue to generate bits for the channel without sending actual encoded image information. MPEG provides for stuffing in several places—for instance, before video start codes and as part of the macroblock address code. Preventing overflows is more difficult. Overflows occur because the material is exhibiting a sustained difficulty relative to the bit rate that has been allocated. The only option available to the encoder is to reduce the fidelity with which the material is compressed in order to avoid overflowing the buffer. The first line of defense is generally the quantizer step size. As described previously, a key in determining quantizer step size is the base quantizer step size variable, which is selected at the encoder's discretion and may be varied on a macroblock-by-macroblock basis. Usually, the base quantizer step size depends, at least partly, on the deviation between expected and observed buffer fullness. Other options for preventing buffer overflow are increasing the number of skipped macroblocks (i.e., increasing the distortion threshold below which we will simply not bother encoding a macroblock) and modulating an external filter to reduce information (particularly high frequencies) entering an encoder. Because of the very long feedback loop involved in the MPEG compression, this latter technique requires great care to avoid instability.

An additional task of rate control is to assign bits to each of the picture types (I, B, and P). As noted earlier, this needs to be highly dynamic to reflect the change in temporal redundancy of the material. The goal is to provide enough bits to each picture type to keep the perceived reconstruction quality constant.

Finally, contemporary encoder rate control often combines some method of scene analysis to additionally modulate quantizer step size. Since the human visual system is less sensitive to distortion in the presence of coarse random texture and more sensitive in flat regions, an encoder will often vary the quantizer step size to spend more bits in areas that are more important to the human visual system.

Good rate control is the parsimonious use of every bit available to the encoding system and can achieve better visual quality at lower bit rates than mediocre rate control. Bad rate control can actually create visible artifacts of its own; the most frequent is a visible pulsing associated with an inappropriate allocation of bits across different picture types.

11.3.8 Constrained Parameters, Levels, and Profiles

The MPEG video algorithm provides an enormous flexibility in terms of the image sizes, bit rates, and other key parameters that can be supported. It is unreasonable to expect simple applications to bear the cost of decoding at the substantially higher complexity of the worst-case use of the standard, particularly if the simpler applications will be built in highly cost-sensitive high-volume applications. Hence, MPEG has defined compliance points within the standard.

TABLE 11.4 Defined Levels and Profiles for MPEG-2

Level	Profile				
	Simple	Main	SNR	Spatial	High
High		X			X
High 1440		X		X	X
Main	X	X	X		X
Low		X	X		

For MPEG-1, the *constrained parameters* case provides the following set of specific restrictions that are reasonable for the kind of moderate-resolution, multimedia applications in the 1–3 Mbps range for which MPEG-1 was optimized.

- Horizontal size less than or equal to 720 pixels
- Vertical size less than or equal to 576 pixels
- Total number of macroblocks/picture less than or equal to 396
- Total number of macroblocks/second less than or equal to $396 \times 25 = 330 \times 30$
- Picture rate less than or equal to 30 pictures/second
- Bit rate less than or equal to 1.86 Mbps
- Decoder buffer less than or equal to 376,832 bits

Although the first two constraints seem to allow large image sizes (up to the full resolution of broadcast digital CCIR-601), the third and fourth constraints are more limiting in this regard. For instance, the Standard Intermediate Format (SIF) sizes of 352H × 240V (NTSC countries) and 352H × 288V (PAL countries) just satisfy these constraints; a CCIR-601 resolution would not. Much of the commodity silicon targeted at MPEG-1 applications is designed to the constrained parameters level.

SIF = CIF
↑
Common Int. For.

MPEG-2 is designed to address a much larger number of potential applications, and in order to be cost-effective in each, it has a much larger set of compliance points. These are indexed by *levels*, which set a rough limit on processing power based on image size, and *profiles*, which restrict the algorithmic features available. Table 11.4 shows the grid of compliance points for MPEG-2.

The compliance point addressed by a particular system is stated as "*x* profile at *y* level." The most popular compliance point is, not surprisingly, main profile at main level (or MP @ ML, for short). Following are brief discussions of each of the levels:

- Low—This sets constraints on image size of approximately half CCIR-601 vertically and horizontally, namely, 352H×288V. (SIF)
- Main—CCIR-601 size images, 720H×576V×25Hz or 720H×480V×29.97Hz.
- High 1440—HDTV up to 1440H×1152V.
- High—HDTV up to 1920H×1152V.

Following are brief descriptions of the profiles:

- Simple—No B-pictures. This allows a reduction in the memory required at decoders of as much as 8 Mbits, at some cost in bit rate efficiency.
- Main—The material in this chapter describes most of the aspects of the main profile.
- SNR—Adds a type of scalability based on transmitting two bitstreams. The first, when decoded, will provide a reasonable reproduction. The second provides additional information to refine the reconstructed DCT coefficients, so that when both are decoded together, an excellent reconstruction results. Such scalability allows simpler and more complex decoders to share the same bitstream.
- Spatial—Adds a different type of scalability, in which the first stream provides a low-resolution reproduction and the combination of both streams provides a full-resolution reproduction.
- High—Allows SNR, spatial, and an additional type of scalability based on the first stream providing a low picture rate reconstruction and both streams providing full picture rate, as well as a number of other high-complexity options.

11.4 MPEG Audio

The MPEG audio compression standard defines a family of algorithms appropriate for a wide range of audio material (e.g., speech, music, and the range of special effects that might be expected on a movie soundtrack) based on a combination of subband compression with techniques to explicitly exploit properties of human hearing.