MPEG Video

Mike Knee

Snell & Wilcox

NOTE: TEXT IN BLUE IS FOR INFORMATION & WILL NOT PRINT

1. Introduction

<<< Click & Copy & Paste for more HEADINGS

This paper gives a brief description of the MPEG video coding standards. It concentrates on MPEG-2 but has a little information on MPEG-1 and MPEG-4 as well.

- Section 2 reviews the compression principles used in MPEG-2 Video and describes how they are used.
- Section 3 describes the structure of an MPEG-2 bitstream and those elements of the MPEG-2 Systems layer most relevant to video coding.
- Section 4 briefly discusses advanced aspects of the use of MPEG-2 Video in digital broadcasting: pre and post-processing, concatenation, switching, transcoding and monitoring.
- Section 5 briefly introduces MPEG-1, advanced profiles of MPEG-2, and MPEG-4 Video
- Section 6 gives a short list of references and Internet resources.

2. Principles of MPEG-2 Video

<<< Click & Copy & Paste for more HEADINGS

2.1 The MPEG-2 standard

The MPEG-2 standard consists of several parts, of which the most important to us is the video part [1]. The standard defines a compressed video bitstream and describes how it can be decoded. It is important to recognise that it does not describe how to take an input picture and compress it to make an MPEG-2 bitstream – it is not a coder specification. The designer of a coder has complete freedom to choose which aspects of the standard to use and how to use them. It is therefore best to think of the MPEG-2 standard as a toolkit for video compression, from which appropriate tools can be selected for different applications.

2.2 Redundancy and irrelevancy

Video compression is hard work. An analogue television signal has a bandwidth of about 5.5 MHz and already incorporates a little bit of compression by overlapping the luminance and colour information using the PAL, NTSC or SECAM standards. Many people assume that digital television allows you to fit more channels into the same space, but this is not true without compression. A standard uncompressed digital television signal [2] has a bit rate of at least 216 Mbit/s, which will occupy a bandwidth of around 50 MHz using a reasonably efficient modulation scheme. So a video compression scheme has to give us a ten-fold reduction in bit rate even to get us back to where we started. To obtain the oft-quoted figure of four or five channels for the price of one, we need to compress our digital television signal by a factor of at least 40.

Fortunately, we can achieve this difficult goal by exploiting two properties of television signals: redundancy and *irrelevancy*.

Redundancy comes from the fact that much of a television signal is predictable. Picture information usually varies slowly in space and time, because much picture material consists of smooth regions of colour, moving smoothly across the screen. The first aim of compression is to remove the redundant information, leaving only the true information content, also known as the *entropy* or the unpredictable part of the signal. In a compression decoder, the redundant information can be put back into the signal, making for *lossless compression*. Unfortunately, even the best lossless compression algorithms can only compress by a factor of 3 on average.

We have to exploit *irrelevancy* in order to obtain the further 15-fold compression we require. Irrelevant information is information which may not be mathematically redundant, but is irrelevant because it cannot be seen by the human eye under certain reasonable viewing conditions. For example, the eye is much less sensitive to noise at high spatial frequencies than at low spatial frequencies, and much less sensitive to loss of resolution immediately before and after a scene change. The MPEG-2 standard includes tools that allow irrelevant information to be removed easily. In practice, these tools only get us about halfway to our goal, so ultimately there is a trade-off between final bit-rate and visible impairments to the picture.

I shall now describe each of the MPEG-2 video tools and attempt to explain how they remove redundant and irrelevant information to produce a compressed bitstream.

2.3 Simple pre-processing

The MPEG-2 toolkit allows for some mild pre-processing of the raw digital video signal, which reduces the bit-rate we have to begin with by about 40%. The two main pre-processing steps are:

- Remove horizontal and vertical blanking, and make sure that non-video waveforms contained in those parts of the video signal (such as teletext) are transmitted separately
- Further subsampling of the chrominance signal by a factor of 2:1 vertically, on top of the 2:1 horizontal subsampling already done in the digital video standard.

Further pre-processing is possible, as described briefly in Section 4.1, but this is not part of the MPEG-2 standard

2.4 Motion compensated prediction

2.4.1 Prediction

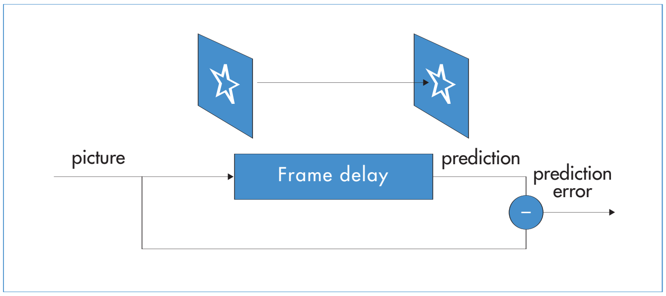
<<< Click & Copy & Paste for more Sub-Sub-Headings

One of the most important ways of reducing redundancy in television picture sequences is the use of motion compensated prediction. Prediction is any process in which past information is used to predict current information, leading to a *prediction error* or *residual* which is the difference between the current information and the prediction. The goal is to make the residual as small as possible, because that is what is transmitted. The decoder can form the same prediction and add it to the transmitted residual to obtain a decoded picture.

2.4.2 Motion compensation

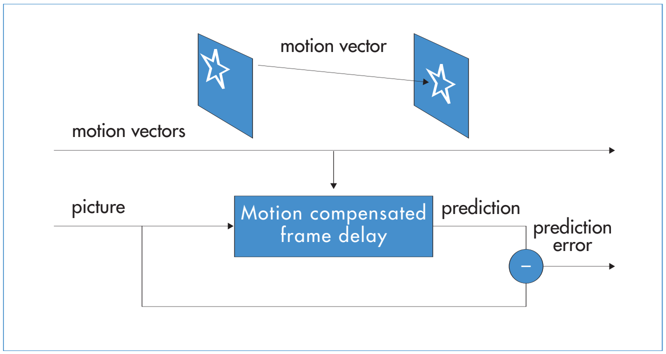
<<< Click & Copy & Paste for more Sub-Sub-Headings

MPEG-2 Video uses temporal prediction. In its simplest form, the prediction for a pixel consists of the same pixel in the previous frame, so the predictor consists of a frame delay as shown here.



For still pictures, a perfect prediction is generated and the prediction error is zero. But when the picture has significant detail and motion, this simple temporal prediction is quite poor, often worse than having no predictor at all.

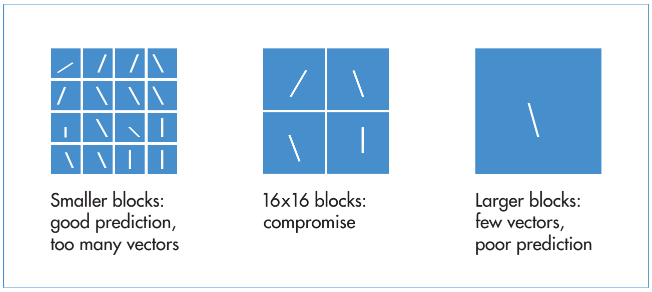
The answer is to use *motion compensation*. If we can measure the motion or displacement between the two frames, we can displace our prediction so that it corrects for the motion. Notice that we now have to transmit the motion vectors as well as the prediction error, so that the decoder can form the same prediction.



2.4.3 Motion compensation block size

<<< Click & Copy & Paste for more Sub-Sub-Headings

There is now a trade-off between the spatial resolution of the motion vector field and the cost of transmitting the motion vectors. At one extreme, having a motion vector for every pixel should produce an extremely good prediction, but at the cost of having to transmit a huge number of motion vectors. At the other extreme, a single, global, motion vector for the whole picture costs almost nothing to transmit, but will not give a very good prediction. The MPEG committee settled on transmitting one (or two, depending on interlace characteristics) motion vector for each 16x16 block or *macroblock*.



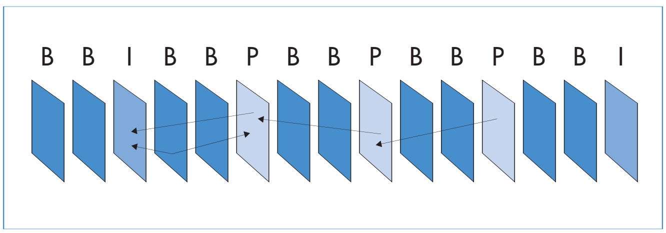
2.4.4 Methods of motion estimation

The motion estimator is one of the most complicated and computationally intensive parts of any MPEG-2 video coder. Most coders use a method based on block matching, in which the best match to a current macroblock is sought within some search range. So it is often said that "MPEG-2 uses block matching". But we should remember that the coding method is not defined in the standard. Alternatives are possible, of which the most successful seen so far is to take a high-quality, pixel based set of field-to-field motion vectors obtained by phase correlation [3] and to obtain the MPEG-2 macroblock-based vectors for prediction by tracing those pixel-based vectors from field to field [4]. This approach significantly reduces the overhead for transmitting motion vectors and increases the visual quality of the residual. Both those advantages are important, particularly when coding at low bit rates.

2.4.5 Bidirectional prediction and the Group of Pictures

In MPEG, pictures that are predicted from past information are known as *P-pictures*. You can obtain an even better prediction if you are able to make use of pictures in the past and in the future. Such pictures are called *bidirectionally predicted pictures*, or *B-pictures*. They make use of two pictures for the prediction, one coming before the current picture and one after. For each macroblock, a choice can be made of which picture will give the better prediction, or whether it is better still to take an average of two predictions. Of course, such a system has to be causal; the "future" picture used in bidirectional prediction actually has to be transmitted first and does not itself use bidirectional prediction. What typically happens in MPEG-2 is that every third picture is transmitted as a P-picture and intermediate pictures are sent as B-pictures. From time to time, prediction has to be disabled so that the decoder has something to start with when first switched on or following an error. Such pictures without prediction are called *I-pictures (intra pictures)*.

The resulting pattern of I, P and B pictures is known as the *Group of Pictures structure* or *GOP structure*. A typical example of a GOP is shown here, with arrows pointing to the pictures from which predictions are taken.



<<< Click & Copy & Paste for more Sub-Sub-Headings

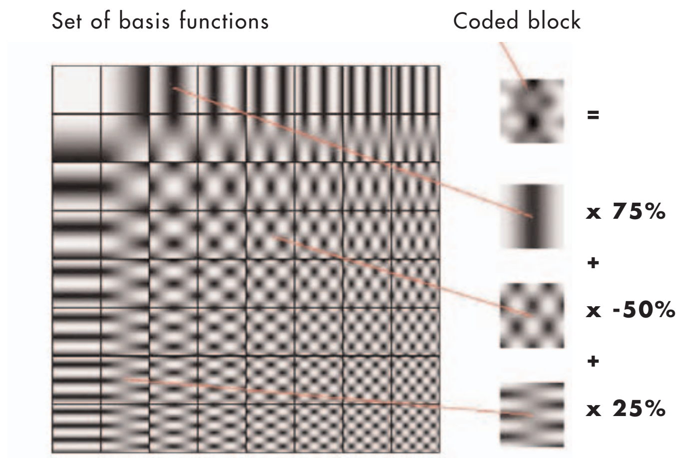
2.4.6 Prediction modes

We have already seen for B-pictures that the coder has a choice for each macroblock as to whether to use forward, backward or bidirectional prediction. There are other choices to be made for each macroblock in all predicted pictures, for example, whether a 16x16 macroblock as taken as a whole or whether it is split into two 16x8 halves with lines coming from two interlaced fields. Further choices can be made: whether to turn off the motion compensation, setting the motion vectors to zero, and finally whether to turn off the prediction altogether and send an intra-coded macroblock.

2.5 The Discrete Cosine Transform

The second key technique of MPEG Video is the use of the Discrete Cosine Transform or DCT. Although this is cascaded with the prediction error generation in an MPEG coder, it really comes into its own when no prediction is involved and an intra-coded picture or macroblock is being transmitted. In MPEG, the DCT is applied separately to each 8x8 block of the picture. There are several ways to envisage the DCT. It can be thought of as a matrix multiplication, as an axis rotation in 64- dimensional space, as the FFT of a block placed next to its reflection, or as a decomposition into basis functions. There are plenty of sources (e.g. [5]) for more mathematical descriptions of the DCT; here I shall give a description in terms of decomposing the block into basis functions, which are blocks containing "pure" signals of specific horizontal and vertical spatial frequencies. The function of the DCT is to describe how much of each basis function is present in the block; the resulting multipliers for each basis function are known as DCT coefficients, from which the original block can be recovered via the inverse DCT.

Here is an example of the DCT applied to a block of luminance information which could be from a real picture – a defocused stone Celtic cross, perhaps, or a small figure in a crowd:



In this example, the block to be coded is found to contain three of the 64 basis functions, in the proportions shown.

The DCT description of a picture can be used to exploit both aspects of compression introduced in Section 2.2. First, redundancy is reduced because the vast majority of non-zero DCT coefficients of a picture correspond to basis functions towards the top left of the set of basis functions. We can exploit this highly non-uniform distribution of the DCT coefficients by using variable-length and runlength coding as described below.

The DCT description also provides us with an ideal mechanism to exploit irrelevancy. Having now decomposed the picture information into spatial frequencies, we are now in a position to code the higher frequencies with less precision than the lower frequencies. This is the function of the quantizer weighting matrices described in the next section.

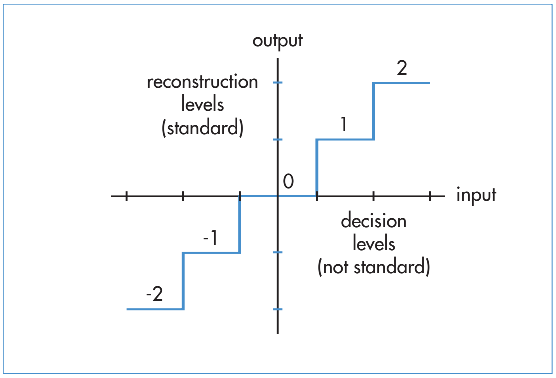
2.6 Quantization

The processes of prediction error generation and DCT transformation have the paradoxical effect of *increasing* the number of bits needed to describe each pixel. If the input signal has 8 bits per pixel, the output of the DCT applied to prediction errors will have 12 bits per pixel! It is not until we begin to reduce the precision of the DCT coefficients that we can perform any compression. This action is known as *quantization*, and has two components in MPEG.

First, we apply a *weighting matrix* to the 64 DCT coefficients. We divide the coefficients by a grid of numbers which follow the amount of noise that the human eye can tolerate at each spatial frequency. The default grid used in the MPEG standard is shown here:

8	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

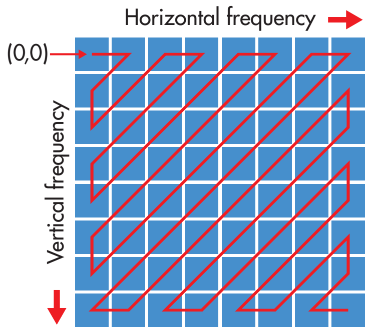
Second, we perform a global quantization of the weighted coefficients by mapping the input levels into a reduced set of output levels. This can be done by straight division and truncation of the result, but the MPEG standard does not specify exactly how the mapping is performed – only the output levels are specified. Here is an example of a quantizer mapping:



The overall spacing of the quantizer reconstruction levels affects both the final picture quality and the bit rate, and can be varied on a macroblock basis both in response to measures of how critical the macroblock is and, more importantly, in response to the bit-rate control mechanism described below.

2.7 Variable-length coding

The variable-length coding performed in MPEG-2 Video is a combination of variable-length coding of the DCT coefficients, which have highly peaked probability distributions, and run-length coding of zero-valued DCT coefficients. In order to benefit from run-length coding, the DCT coefficients of a block need to be re-ordered in such a way that long runs of zeros are more likely. This is done by scanning the coefficients in a zigzag pattern as shown here:



In fact, the variable-length coding of non-zero coefficients and the run-length coding of zero coefficients are combined into one Huffman code [6] for which each symbol represents a certain number of zeros followed by a non-zero value.

Huffman coding is also applied to macroblock prediction modes and to motion vectors (which are transmitted as differences between motion vectors on successive macroblocks). Run-length coding is effectively applied to these parameters through a mechanism of "skipped macroblocks", and to quantizer scaling information by only transmitting changes in the quantizer scale from one macroblock to the next.

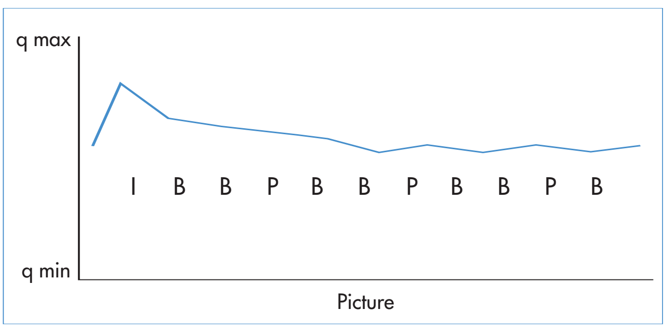
2.8 Rate control

The primary use of MPEG coding in broadcasting is to generate a constant bit rate to fit the available channel or, in the case of statistical multiplexing, to share a constant bit rate between several video signals. In both cases, the transmission bit rate required is rarely equal to the instantaneous bit rate at the output of the variable-length coder. A buffer store is therefore introduced at the encoder output, with a complementary buffer store at the decoder input to ensure that for every picture that enters the encoder, a picture leaves the decoder. It is then important to ensure that the average bit-rate of the buffer input is the same as that of the channel and that neither buffer overflows or underflows.

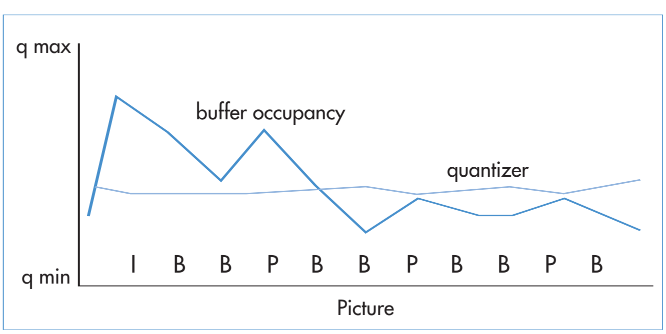
How can the average bit rate at the encoder buffer input be controlled? The most common method is to vary the overall quantizer scale. A coarser scale will generate a lower average bit rate, at the expense of picture quality. A finer scale will produce better pictures but at a higher average bit rate. Unfortunately, it will be impossible to fix the quantizer at a certain level which has previously been calculated to generate the correct average bit-rate, because the picture activity varies unpredictably. We should note here that quantizer scale is not the only adjustable parameter affecting bit rate. Image pre-filtering and quantizer weighting matrices, for example, can also be used.

The design of a rate control algorithm is an interesting application of control theory. It differs from most control problems, where the goal is to keep a parameter as constant as possible by turning a knob – and subject to certain limits, the position of the knob is irrelevant. In rate control, the goal is to keep a parameter (the buffer occupancy) within fairly broad limits by turning a knob (typically the overall quantizer scale) as slowly as possible.

The simplest rate control algorithm for real-time encoders is to make the quantizer scale proportional to the buffer occupancy. As the buffer fills, the quantizer gets coarser, which tends to reduce the average bit rate, helping the buffer to empty. For a typical GOP structure, the following graph shows what would happen to the quantizer scale if such an algorithm were used.



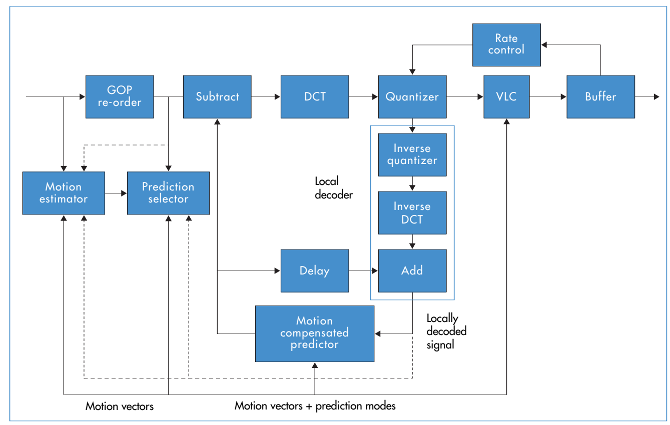
While very stable, such a simplistic algorithm fails to deliver a constant quantizer, even if the picture activity is uniform, because the I-frame generates many more bits than the other frames. The aim of a good rate control algorithm is to plan for the expected differences in bit rates generated by I, P and B frames. Many algorithms apply some kind of model, based on measurements of bit rate against quantizer for a training set of sequences. An example of the buffer occupancy and quantizer scale resulting from such an algorithm is given here:



Rate control algorithms vary greatly according to the application. For example, for variable-bit rate operation, the buffer constraints may be even less severe, but the requirement for constant quality remains just as important. In non-real-time encoding, for example DVD authoring, the immediate buffer and bit-rate constraints are very mild but there is a requirement to fit the total bitstream within a certain space, preferably with not too much to spare. Such applications have the added possibility of *multiple-pass encoding,* whereby the entire sequence can be analysed or trial-encoded once or more before making the final rate control decisions for every picture.

2.9 Complete coder

The following diagram shows all the tools described above put together to form an MPEG coder.



2.9.1 Main encoding path

<<< Click & Copy & Paste for more Sub-Sub-Headings

The heart of the MPEG encoder is actually the subtractor. Here, the motion compensated prediction for a picture is subtracted from the input picture, which has been presented in the correct order for encoding according to the desired GOP structure. The subtractor is disabled (or the prediction set to zero) for I-frames. The output of the subtractor, which is the prediction error (or the picture in the case of I-frames), is passed to the DCT, whose output coefficients are quantized. The output of the quantizer is a level which is zigzag scanned, run-length and variable-length coded in the block marked VLC. The variable rate bitstream is buffered and the occupancy of the buffer is monitored by the rate control block which controls the quantizer.

2.9.2 The need for a local decoder

<<< Click & Copy & Paste for more Sub-Sub-Headings

The motion compensated predictor requires a signal on which to base its prediction. This signal will be delayed internally in the predictor. It is important to see that the predictor is in a loop which must be replicated in the remote decoder. The predictor must therefore base its prediction on a signal which is also available in the remote decoder. That is why we have to generate a *locally decoded* signal in the encoder. This is done in the *local decoder,* which consists of blocks to undo the encoding stages of quantizer and DCT, followed by an adder which takes the decoded prediction error and adds it back in to a suitably delayed version of the prediction to produce a locally decoded signal.

2.9.3 Motion estimation and prediction selection

<<< Click & Copy & Paste for more Sub-Sub-Headings

Two more blocks need to be discussed. The first is the *motion estimator,* which estimates the displacement between the current picture and the reference picture(s) that will be used in the prediction. In this diagram, it is shown taking its main input from the display-order signal. This is because, in the implementation of MPEG-2 most familiar to the author [4], the basic motion estimation is carried out between fields in the input picture. It is interesting that the motion estimator also has an optional input from the local decoder. This is because it has been found that for the final stage in motion estimation, when vectors are calculated to sub-pixel accuracy, it is better to use decoded versions of the reference frames, which contain the impairments and artefacts that will actually affect the prediction, than to base the entire estimation process on uncompressed input frames.

Finally, there is a *prediction selection* block, which decides (for example) whether forward, backward or bidirectional prediction is to be used for each macroblock in turn, whether the prediction is frame-based or field based and on which fields, whether it is better to set the motion vector to zero and ultimately whether it is worth having a prediction at all. Again, such decisions can be made using a combination of uncompressed input information and locally decoded information.

The results of the motion estimation and prediction selection processes need to be transmitted to the decoder as part of the MPEG bitstream. Like the quantized DCT coefficients, motion vectors and prediction modes are also variable-length and run-length coded.

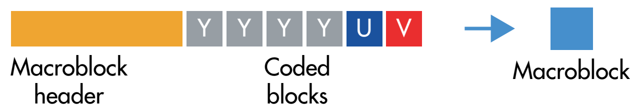
3. MPEG-2 Bitstreams

3.1 Video Elementary Stream

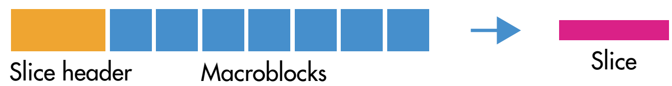
The output of an MPEG-2 video encoder is known as an *Elementary Stream* (ES). If such a stream is held on a server, it contains all the information necessary to produce a decoded video signal.

We shall now describe how the video elementary stream is built up.

The lowest-level entity in the stream is a coded *block* of DCT coefficients. Each block is terminated by an end-of-block code and the four luminance and two chrominance blocks in the macroblock are simply concatenated. The coded blocks are preceded by a *macroblock header* which contains all the control information belonging to the macroblock: spatial address, motion vectors, prediction modes, field/frame DCT mode, quantizer step size. The result is a coded *macroblock.*



Macroblocks are then grouped together into a slice:



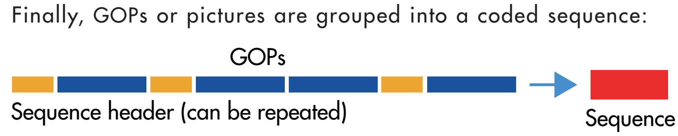
A slice is a set of consecutive macroblocks preceded by a *slice header.* The important thing about a slice is that it is the smallest entity in an MPEG elementary stream on which synchronization can be attained. The slice header contains a unique *start code* which cannot be duplicated elsewhere in the bitstream. There is usually one slice for each row of macroblocks, but in error-prone channels or in other applications where quick resynchronization is very important, it is possible to have several slices per row, even one slice per macroblock.

The *picture header* again has a unique start code and contains picture-specific control information, for example whether the picture is a field or a frame, its picture coding type (I, P or B), and the quantizer weighting matrices.

There is then the option of grouping pictures together and preceding them with a *GOP header* to make a *group of pictures:*

The GOP header contains a time code and some flags relating to editing.

Finally, GOPs or pictures are grouped into a coded sequence:



The *sequence header* contains basic information needed before decoding can begin, such as the size of the picture and the frame rate. Because it is so important, the sequence header is usually repeated periodically (say twice a second). Sequences can have an end but in most real-time applications the sequence is of unlimited length.

3.2 MPEG Systems Layer

An MPEG video elementary stream is rarely of much use on its own, except possibly for playing silent video clips over the Internet or from a server. The elementary stream usually forms part of the *MPEG Systems Layer,* which is a separate part of the MPEG standard [7] and which serves four purposes:

- to provide a means of combining or multiplexing video, audio and ancillary data bitstreams
- to transmit programme-specific information, enabling navigation and access control
- to provide a framework for error protection
- to convey timing information to the decoder

3.2.1 Transport Stream

MPEG video, audio and data streams are usually combined together to form a *Transport Stream.* A Transport Stream consists of short, fixed-length (188 byte) *packets* concatenated together. Each packet contains a *header* followed by a *payload* which is a chunk of a particular elementary stream. The header consists of a (non-unique) one-byte *start code* which can be used for flywheel-based synchronization, followed by a *packet identifier* (PID) which is a 13-bit code indicating which elementary stream the payload belongs to.

The task of a transport stream multiplexer is then to form the elementary streams into packets, to buffer them and to select them for transmission according to the rates at which the elementary streams are generated.

One additional complication is that, for reasons partly connected with timing information described below, video elementary streams undergo an intermediate stage of packetization into packetized elementary streams (PESs) at which stage some timing data is added.

3.2.2 Programme-specific information

A Transport Stream may contain several video, audio and data channels. The decoder obviously needs to know which video and audio signals go together to make each particular programme. The MPEG Systems Layer provides a two-layer system of navigation through the Transport Stream. First, packets with a PID of zero contain what is known as the *Programme Association Table* (PAT). This is a list of programmes together with the PIDs of further tables, one for each programme. Each of these further tables, known as *Programme Map Tables* (PMTs) contains a list of all the elementary streams making up the programme and the PIDs identifying them.

The MPEG specification for *programme-specific information* (PSI) also includes a *Network Information Table* (NIT) which contains information about the physical networks different programmes are transmitted on (frequencies, for example) and a *Conditional Access Table* (CAT) which provides a framework for scrambling, encryption and access control, though the actual implementation of conditional access is not part of the MPEG specification.

Particular implementations of MPEG have made numerous additions to the basic programme-specific information to provide more sophisticated navigation tools, enable interactivity and give additional data and software to the user. For example, the European DVB standard contains *Service Information* (SI) containing such things as a *Bouquet Association Table* (BAT), a *Time and Date Table* (TDT) and a *Running Status Table* (RST) as well as more detailed specifications of how conditional access data is transmitted. The American ATSC standard has equivalent but different extensions to the MPEG tables.

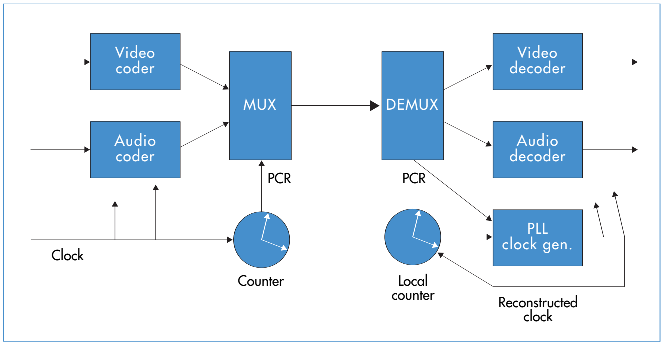
3.2.3 Error protection

The MPEG standard itself does not specify how error protection should be applied to the bitstream, because this is specific to the transmission or recording medium being used. In broadcasting, a common approach is to add error protection to each 188-byte Transport Stream packet, for example a Reed-Solomon code. Error protection is important because the variable-length coding, recursive prediction and limited resynchronization capabilities that are features of MPEG mean that single uncorrected errors in the video elementary stream can make a large mess!

3.2.4 Timing information

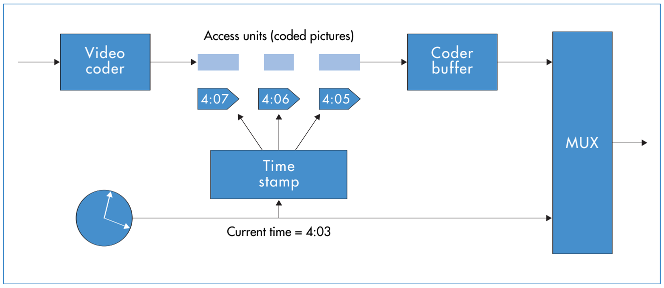
Perhaps one of the most important features of the MPEG Systems Layer is that it provides a mechanism to enable the decoder to remain in step with the encoder when both are working in real time. The decoder has to be able to recreate the master clock that was used at the encoder, so that for every picture that enters the decoder, one leaves the decoder. It then has to know exactly when, according to this clock, it should remove data from the buffer for decoding and presentation to the display device.

The following diagram illustrates how the first of these two problems is solved:



The video-locked master clock at the encoder side drives a counter whose value is periodically multiplexed into the transport stream as the *Programme Clock Reference* (PCR) and transmitted without passing through any of the large elementary stream encoder buffers. It is demultiplexed before passing through any large decoder buffers, so that it has been subjected to a minimum and fixed overall delay. The demultiplexed value is compared with a locally generated count and the difference used to control a local phase-locked-loop clock generator. In this way the decoder clock is kept in sync with the encoder clock. Note that there is no need for the bit rate or channel clock to have any relation to the encoder or decoder clocks, or even to be constant. Different programmes may have independent programme clocks, or a common clock can be shared by several programmes.

The second problem is solved by the use of time stamps, as shown here:

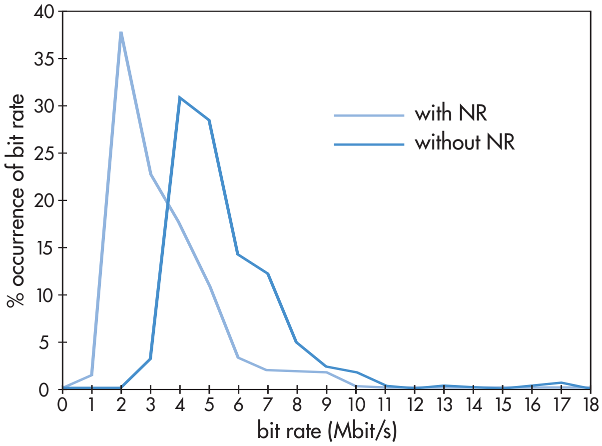


From its programme clock, the encoder knows what time it received each picture. It therefore knows what time the decoder should present each picture to the display, assuming that there is no delay in the channel. It calculates this time for each picture and transmits the information as a timestamp in the picture's PES header. The timestamp stays with its picture, passing through the encoder buffer. At the decoder it is compared with the local programme clock and used to control the decoding and presentation of each picture. Fixed but unknown delays in the channel are not a problem because they affect the PCR and timestamp information to the same extent as the encoded data itself. <<<Click & Copy & Paste for more Sub-Sub-Headings

4. MPEG-2 video in digital broadcasting

4.1 Pre-processing

Any compression system aims to remove redundancy from the bitstream and encode what remains. Random noise on the source picture is by definition not redundant and will not be removed. The encoder will have to devote its bit rate resources to encoding the noise at the expense of the unpredicted parts of the picture. Noise reduction at the input to an encoder is therefore highly desirable. Other pre-processing may be worth while. At low bit rates, the balance between MPEG artefacts and overall lack of sharpness may shift to the extent that it is worth low-pass filtering and subsampling the input picture.
It is also a good idea when starting with a composite signal such as PAL to apply a high-quality decoder which minimizes cross-colour and cross-luminance, both of which are undesirable and eat up bit rate.
This graph shows how much effect a good noise reducer can



have on the efficiency of an MPEG coder. Two encoders are working with fixed coding quality and variable bit-rate on a typically noisy source. A histogram of the occurrence of different bit rates is shown for each encoder. Encoder 2 has no pre-processor while encoder 1 has pre-processing. This simple example shows that preprocessing can easily be worth 2 Mbit/s per channel on fairly typical material.

Further improvements to MPEG-2 encoding quality can be achieved by analysing the video signal and using the results of the analysis to control the encoder. For example, a 60Hz video signal derived from 24Hz film using 3:2 pulldown can be analysed so that the 3:2 phase information is passed to the encoder, which can then make use of the MPEG-2 syntax to avoid re-transmitting the repeated field.

4.2 Concatenation

There are many places in a broadcast chain where it is necessary to decode an MPEG bitstream, perform some video processing, and re-encode the result to MPEG. When the processing is substantial, involving spatial distortions and changes of scale, full-blown re-encoding is inevitable. However, the re-encoding is often working on a signal that may be very close to the decoded signal. In that case, there is a significant concatenation loss because successive encoders may make different encoding decisions. In the five or six generations typical in a broadcast chain, this loss can be substantial – as much as 6 dB in peak signal-to-noise ratio (PSNR) [8]. The concatenation loss can be reduced or even eliminated by re-using the coding decisions that were used in the previous encoder. This information is available in the bitstream and can be extracted from a suitably equipped decoder. International standards now exist for this so-called re-coding data [9] and for particular implementations such as history data [10] and MOLE™ [11]. The MOLE system reduces the concatenation loss to zero by preserving all the picture-level and macroblock-level information, and it offers great flexibility by carrying all the information in the decoded picture itself, in such a way that a suitable re-encoder can find out exactly where the re-coding data is valid and where the picture has been altered.

4.3 Switching

Switching MPEG bitstreams, especially in real time, can be difficult because of the variable numbers of bits per picture and because of the need for access to previously decoded frames when forming motion compensated predictions. There are two main approaches to the problem.

Bitstream splicing [12] relies on choosing suitable points in the bitstream where switching can be carried out. This is relatively simple to implement but is limited to a straight cut, imposes severe restrictions on the choice of exact switching point and often imposes limitations on the original encoding process, forcing it to achieve a certain buffer occupancy at a certain time, for example.

The other approach is to decode the bitstreams and perform the switching function in the decoded video domain. This offers complete flexibility but can only be done without loss if re-coding data is used. The MOLE system allows a completely standard digital video mixer or DVE to be used while eliminating concatenation losses wherever this is possible [13].

4.4 Transcoding

Transcoding covers any process where one bitstream is converted to another. The conversion may beto another but related coding standard, such as DV, or it may be to another MPEG bitstream at a reduced rate. In both cases, it is l important to make proper use of re-coding data wherever possible. Sometimes the quality of transcoding can be significantly improved if further use is made of information about the previous encoding process. For example, the use of *a maximum a posteriori probability* (MAP) quantizer [14] can lead to a transcoded picture quality that is as good as "standalone" coding at the transcoder's output bit rate.

4.5 Post-processing

The MPEG standard assumes that there may be some post-processing in the form of chrominance sample rate upconversion. Other than that, the standard does not really specify exactly what should be done by way of post-processing for display. There are several ways in which decoders may differentiate themselves from competing devices by applying post-processing, for example:

- concealment of uncorrected errors. The MPEG standard provides for the transmission of concealment motion vectors which can be used in "repairing" or replacing corrupted blocks, but as usual does not specify how such information should be calculated or used
- reduction in visibility of blocking artefacts by adaptive filtering in response to some measure of the "blockiness" of the decoded picture
- non-standard enhancements to MPEG-2 in which user information or metadata is used to convey instructions to the display processor on how to apply post-filtering

4.6 Monitoring

One problem which has beset MPEG engineers since before the standard was published is that of monitoring or measuring coded picture quality. Such monitoring must take place on many levels, including bitstream integrity and consistency of systems-layer tables. At the level of the video elementary stream, there are two main approaches to measurement or monitoring, each of which has its place.

First is *double-ended* measurement, in which a decoded picture is compared with a source. The metrics used vary from the simple peak signal to noise ratio (PSNR) to a number of methods that try to take the properties of the human visual system into account. A very detailed comparison of the main contenders was carried out by the ITU Visual Quality Experts'

Group (VQEG) [15]. Doubleended measurement is useful as a lab tool but not for most monitoring purposes where there is no access to the uncompressed source.

The other approach is single-ended measurement, which attempts to estimate the level of degradation in a decoded picture without reference to the source, by looking at the decoded picture itself or at parameters in the bitstream. Two methods based on this approach are blockiness measurement on the decoded picture [16], and PAR™, which gives an estimate of PSNR based on quantizer information and bit counts taken from the bitstream [17].

5. Other MPEG Flavours

5.1 MPEG-1

MPEG-1 [18] is the precursor to MPEG-2. It was designed to compress progressively scanned, halfresolution images to 1.5 Mbit/s so that a feature film could be compressed onto a standard CD. It is a subset of MPEG-2, lacking the later standard's ability to handle interlaced sequences.

5.2 MPEG-2 Levels and Profiles

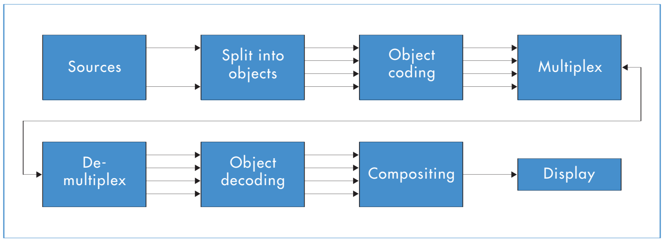
MPEG-2 is designed for a wide range of applications. It is clearly unreasonable to demand that a decoder can only be deemed compliant with the standard if it can apply the entire MPEG-2 toolkit on the maximum possible picture and sample rate likely to be encountered. The standard gets over this difficulty by defining a two-dimensional matrix of compliance points at which decoders can be specified.

One dimension of the matrix is that of *levels,* which define different sets of picture size and clock rate limits. Low level handles MPEG-1 resolution, Main level is for standard definition TV, and the High-1440 and High levels are for HDTV resolutions.

The other dimension is *profiles,* which are different degrees of complexity. Everything described so far in this paper relates to Main profile, which is the most common. Simple profile allows for a system without B-frames, while a number of "scalable" profiles are more complex, involving two or more bitstreams; a base layer with a basic level of picture quality, and an enhancement layer providing a "top-up" giving a greater degree of spatial, temporal or greyscale resolution. The base layer can be given a greater degree of error protection, while enhancement layer packets can be flagged as less critical in packet-based transmission systems such as ATM.

5.3 MPEG-4

There is no space here for more than a few sentences about the newer MPEG-4 standard [19]. It differs from MPEG-2 by handling visual objects rather than just pictures. This diagram is a very high-level representation of an MPEG-4 chain:



Video information is coded separately as synthetic or natural objects, and the resulting bitstreams are multiplexed together. The decoder contains a *compositor* which puts the decoded objects back together, under the control of instructions which are either decoded from the bitstream or over-ridden by input from the user, providing a measure of interactivity.

In coding natural video material, MPEG-4 provides some enhancements to the MPEG-2 toolkit, such as adaptive DC prediction, AC coefficient prediction, reversible VLC coding, global motion compensation, quarter-pixel motion estimation and shape-adaptive DCT coding as well as wavelet coding of textures and the use of sprites. However, the consensus seems to be that the chief interest of MPEG-4 is in offering increased functionality rather than a huge leap in coding efficiency.

5.4 JVT

The latest emerging standard in the field of broadcast digital video compression has been put together by the Joint Video Team (JVT) of MPEG and the telecoms branch of the International Telecommunications Union (ITU-T). On the face of it, the JVT codec is not a radical departure from the motion compensated hybrid model common to MPEG-1, MPEG-2 and MPEG-4. However, its careful design and specific targeting to "conventional" rectangular picture sequences (it does not use the synthetic and object-based tools of MPEG-4) have led to a substantial increase in coding quality at low bit rates. For example, some tests have indicated that it can offer a threefold reduction in bit rate for a given quality over MPEG-2.

The main features of the JVT codec which lead to such good coding performance include:

- multiple reference frames for prediction
- adaptive macroblock sizes and shapes, ranging from 4x4 to 16x16
- a 4 x 4 transform with integer coefficients (yes, simpler than the DCT!)

- context-based binary arithmetic coding (CABAC) using an adaptive probability model and providing a non-integer number of bits per symbol.

The remarkable degree of consensus about the JVT specification has somewhat perversely led to its going under many different names, including JVT itself, H.26L, H.264 and ISO-14496 (MPEG-4) part 10, Advanced Video Coding (AVC). It is likely that the next version of the DVB standard will include JVT.

6. REFERENCES

1. ISO/IEC Standard 13818-2:2000. Information technology – generic coding of moving pictures and associated audio information: Video.
2. ITU-R Recommendation ITU-R BT.601-5, 1995. Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios.
3. Kuglin, C D and Hines, D C. The phase correlation image alignment method. Proc. Int. Conf. on Cybernetics and Society, 1995, pp 163-5
4. Thomas, G A and Dancer, S J. Improved motion estimation for MPEG coding within the RACE 'COUGAR' project. Proc. International Broadcasting Convention, 14-18 September 1995. IEE Conference Publication no. 413, pp 238-243.
5. Sandbank, C P (ed.). Digital television. Wiley, June 1990.
6. Huffman, D A. A method for the construction of minimum-redundancy codes. Proc. IRE, 1952, pp 1098-1101.
7. ISO/IEC Standard 13818-1:2000. Information technology – generic coding of moving pictures and associated audio information: Systems.
8. Knee M J and Wells N D. Seamless concatenation – a 21st century dream. Proc. International Television Symposium, Montreux, June 1997
9. SMPTE Standard 327M-2000. MPEG-2 video recoding data set.
10. SMPTE Standard 329M-2000. MPEG-2 video recoding data set – compressed stream format.
11. SMPTE Standard 319M-2000. Transporting MPEG-2 recoding information through 4:2:2 component digital interfaces.
12. SMPTE Standard 312M-1999. Splice points for MPEG-2 transport streams.
13. Brightwell, P J, Dancer S J and Knee, M J. Flexible switching and editing of MPEG-2 video bitstreams. Proc. International Broadcasting Convention, 12-16 September 1997. IEE Conference Publication no. 447, pp 547-552.
14. Tudor, P N and Werner, O H. Real-time transcoding of MPEG-2 video bit streams. Proc. International Broadcasting Convention, 12-16 September 1997. IEE Conference Publication no. 447, pp 296-301.
15. ITU Study Group VQEG (Visual Quality Experts' Group). See http://www.itu.int and ftp://ftp.its.bldrdoc.gov/dist/ituvidq
16. Lauterjung J. Picture quality measurement. Proc. International Broadcasting Convention, 11-15 September 1998. IEE, London, 1998, pp 413-7.
17. Knee, M. A single-ended picture quality measure for MPEG-2. Proc. International Broadcasting Convention, 7-12 September 2000. IEE, London, 2000, pp 95-100.
18. ISO/IEC Standard 11172-2:1993. Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s – Part 2: Video
19. ISO/IEC Standard 14496-2:2001. Information technology – Coding of audio-visual objects – Part 2: Visual

6.1 Internet resources

http://www.mpeg.org/	Unofficial MPEG site
http://mpeg.telecomitalialab.com/	Official MPEG site
http://www.dvb.org/	European DVB standards
http://www.atsc.org/	American ATSC standards
http://www.smpte.org	SMPTE standards
http://www.itu.int	International Telecommunications Union

snellwilcox.com

For further information

please contact one of our international sales offices:

Americas

New York
Snell & Wilcox Inc.
274 Madison Avenue
Suite 1704
New York City
NY 10016
Tel: +1 212 481 1830
Fax: +1 212 481 2642
info@snellamerica.com

Burbank
Snell & Wilcox Inc.
3519 Pacific Ave.
Burbank, CA 91505
Tel: +1 818 556 2616
Fax: +1 818 556 2626
info@snellamerica.com

Asia Pacific

Hong Kong
Snell & Wilcox (Hong Kong) Ltd.
Room 603, Tai Tung Building
No.8 Fleming Road
Wanchai
Hong Kong
Tel: +852 2356 1660
Fax: +852 2575 1690
swhk@snellwilcox.com.hk

China
Snell & Wilcox China
Unit 17
12th Floor of Tower B
Beijing COFCO Plaza
No.8 Jianguomennei Dajie
Dongcheng District
Beijing 100005
China
Tel: +86 10 6528 2761
Fax: +86 10 6528 7635
swchina@snellwilcox.com

India
Snell & Wilcox (India Liaison Office)
I.B. Centre
N-6 Greater Kailash-1
(Market)
New Delhi 110 048
India
Tel: +91 (0) 11 641 2609
Fax: +91 (0) 11 621 8777
swindia@snellwilcox.com

Japan
Snell & Wilcox Japan KK
3-1-3 Ebisu
Shibuya-ku
Tokyo
150-0013
Japan
Tel: +81 (0) 33 446 3996
Fax: +81 (0) 35 449 7392
swjapan@snellwilcox.com

South East Asia

Snell & Wilcox
South East Asia Sdn. Bhd.
Suite 9-09
Plaza 138
Jalan Ampang
50450 Kuala Lumpur
Malaysia
Tel: +60 (0) 3 2732 5557
Fax: +60 (0) 3 2732 8669
swmalaysia@snellwilcox.com

Europe, Middle East & Africa

UK
Snell & Wilcox Ltd.
Southleigh Park House
Eastleigh Road
Havant
Hampshire
PO9 2PE
United Kingdom
Tel: +44 (0)23 9248 9000
Fax: +44 (0)23 9245 1411
info@snellwilcox.com

France
Snell & Wilcox France
3 rue de Rome
93110 Rosny Sous Bois
France
Tel: +33 (0) 1 45 28 1000
Fax: +33 (0) 1 45 28 6452
swfrance@snellwilcox.com

Germany
Snell & Wilcox GmbH
Senefelderstrasse 3a
65205 Wiesbaden
Germany
Tel: +49 (0) 6122 98430
Fax: +49 (0) 6122 9843 55
swgermany@snellwilcox.de

Russia
Snell & Wilcox ZAO
Room 214
35 Arbat str.
Moscow 121835
Russia
Tel: +7 095 248 3443
Fax: +7 095 248 1104
arbat35@snellwilcox.ru

Code Corp17 03/06 v1