SNELL & WILCOX
Engineering with Vision

MPEG VIDEO

The MPEG video compression principles are not easy to grasp, expecially for many in CCTV with an average understanding of broadcast TV and digital video. This well written and illustrated article by Mike Knee from Snell & Wilcox explains such concepts very nicely.

INTRODUCTION

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.

We review the compression principles used in MPEG-2 Video and describe how they are used.

PRINCIPLES OF MPEG-2 VIDEO

The MPEG-2 standard

The MPEG-2 standard consists of several parts, of which the most important to us is the video part. 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.

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 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 exam-

ple, 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.

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, but this is not part of the MPEG-2 standard.

Motion compensated prediction

Prediction

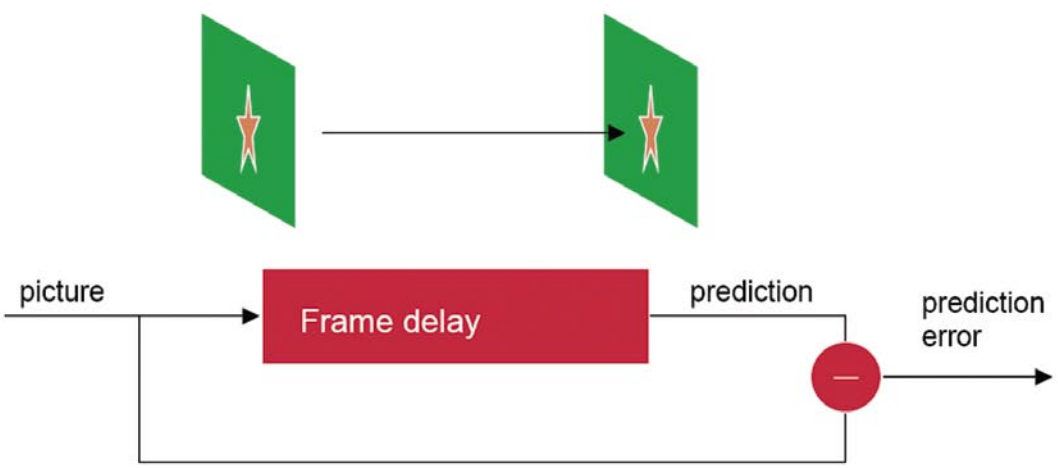
One of the most important ways of reducing redundancy in television picture sequences is the use of **motion compensated prediction**. Prediction is any process in which past information is used to predict current information, leading to a **prediction error** or **residual** which is the difference 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.

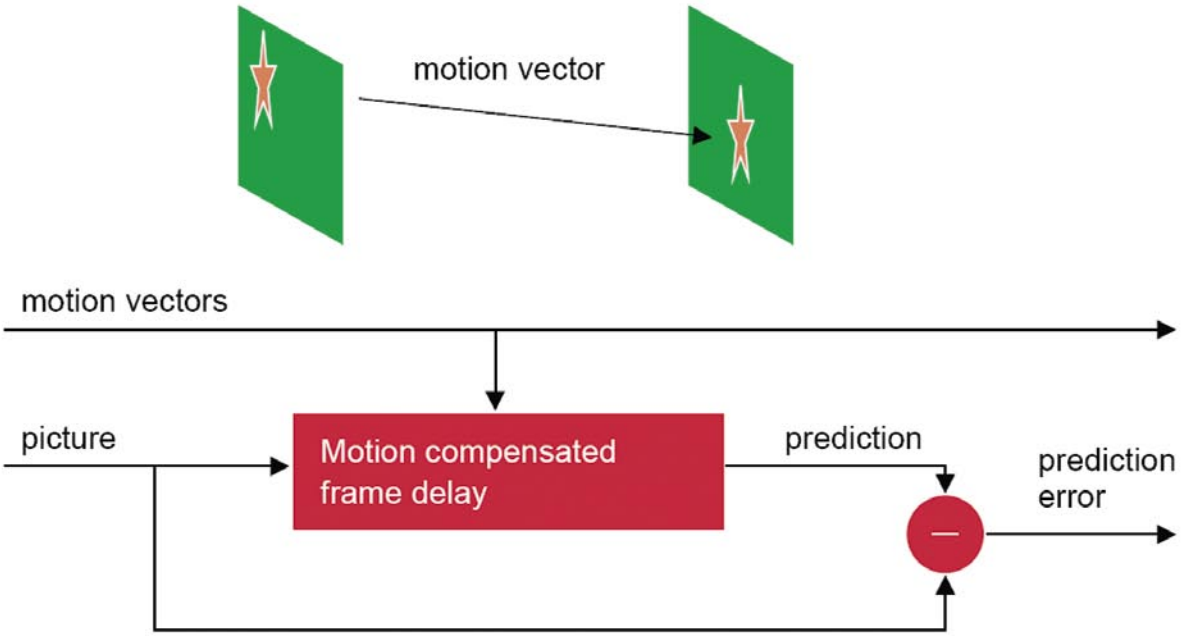
Motion compensation

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 below.

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.

Motion compensation block size

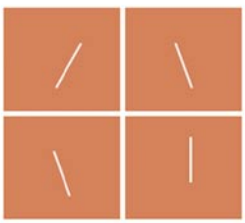
There is now a trade-off between the spatial resolution of the motion vector field and the cost of transmitting the motion vectors. At one extreme, having a motion vector for every pixel should produce an extremely good prediction, but at the cost of having to transmit a huge number of motion vectors. 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**.

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** and to obtain the MPEG-2 macroblock-based vectors for prediction by **tracing** those pixel-based vectors from field to field. This approach significantly reduces the overhead for transmitting motion vectors and increases the visual quality of the residual. Both those advantages are

Smaller blocks:
good prediction,
too many vectors



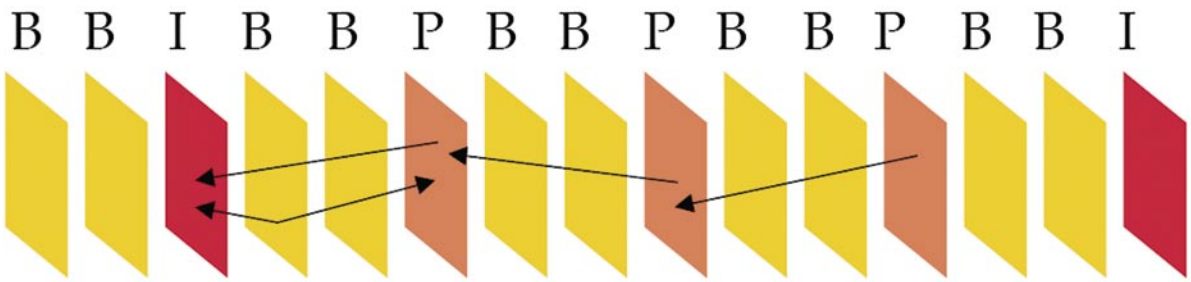
16x16 blocks:
compromise

Larger blocks:
few vectors,
poor prediction

important, particularly when coding at low bit rates.

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.



Group Of Pictures

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.

Prediction modes

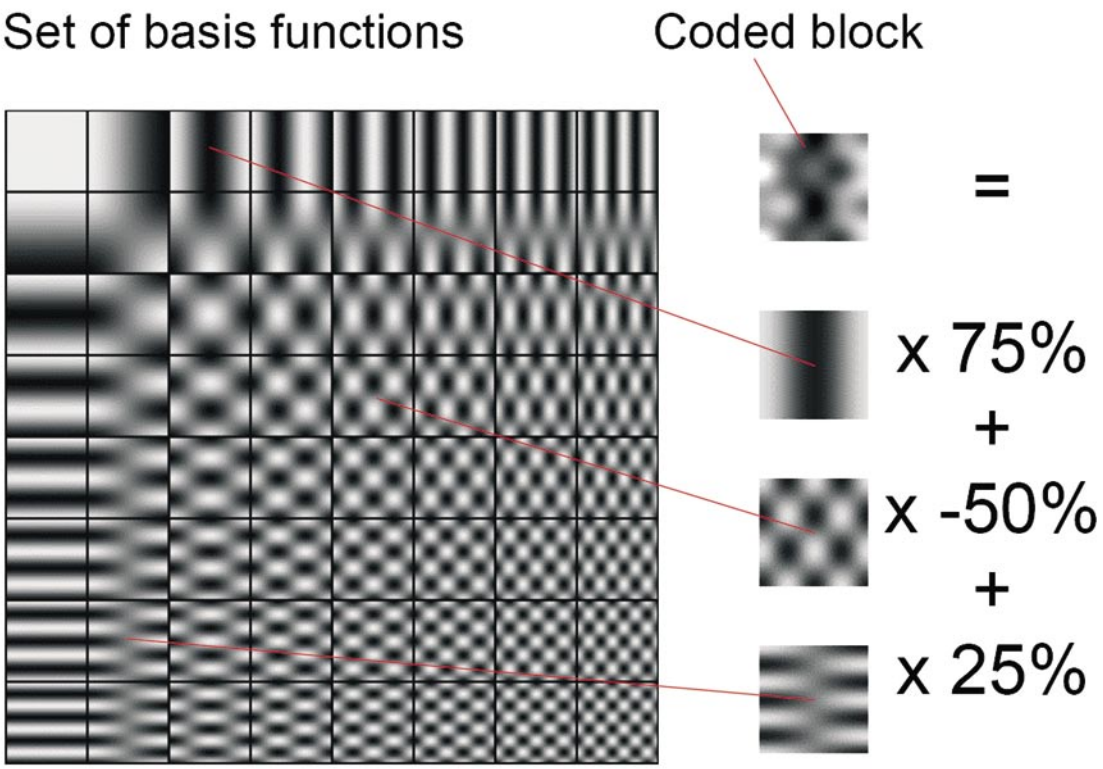
We have already seen for B-pictures that the coder has a choice for each macroblock as to whether to use forward, backward or bidirectional prediction. There are other choices to be made for each macroblock in all predicted pictures, for example, whether a 16x16 macroblock as taken as a whole or whether 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.

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 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 (see illustration above).

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 previous section. 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.

Quantization

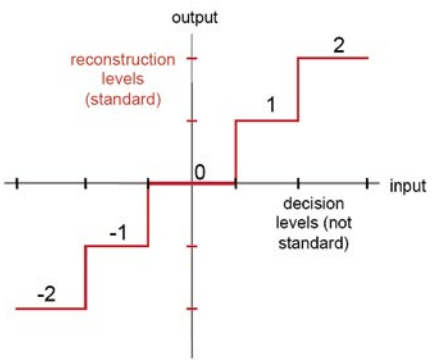
The processes of prediction error generation and DCT transformation have the paradoxical effect of increasing the number of bits needed to describe each pixel. If the input signal has 8 bits per pixel, the output of the DCT applied to prediction errors will have 12 bits per pixel! It is not until we 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 in the following table:

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

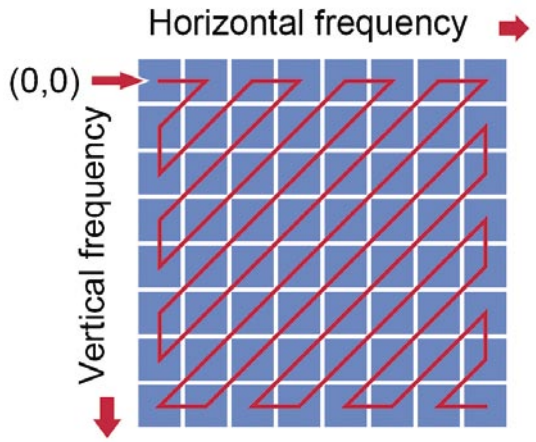
Second, we perform a global quantization of the weighted coefficients by mapping the input levels into a reduced set of output levels. This can be done by straight division and truncation of the result, but the MPEG standard does not specify exactly how the mapping is performed – only the output levels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

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 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. [•]

For more information on Snell & Wilcox please visit their web site www.snellwilcox.com