Videoconferencing
A review of picture compression and coding techniques

N W Garnham BEng(Hons) AMIEE

Department of Electrical and Electronic Engineering
University of Nottingham
United Kingdom

Indexing terms: Video Codecs, Videophony, Image Compression, Transform Coding, Low Bit-Rate Coding

---

Abstract: *This paper introduces the current state of technology for videoconferencing systems, with particular reference to video codecs based on the CCITT H261 algorithm. A description of the DPCM/DCT coding process is given, along with introductory coverage of motion compensated prediction and an explanation of video standards.*

---

1. Introduction

Television is a 20th century phenomena now commonplace in homes throughout the world. The vast demand for television has resulted in large scale production and the use of advanced techniques to bring pictures to our screens. Indeed, the rapid advance in technology can be demonstrated by comparing the 'insides' of a modern television set with one ten years its senior. Older models are a maze of chips, discretes and wires that would confuse the brightest technician - contemporary sets have just a few VLSI chips and interconnecting circuit boards to simplify construction.

Although modern television sets are relatively inexpensive, the concept of television using terrestrial communications systems is, from an engineering viewpoint, highly expensive. This cost is not so much hardware related, but is down to the bandwidth required to transmit a channel of sound and pictures for colour television. The four terrestrial channels in the United Kingdom require bandwidths of 8MHz to carry sound, chrominance (colour) and luminance (brightness) signals and with space in the radio frequency spectrum being a rare commodity, television can be said to come to us at a very high price.

Relating the bandwidth constraints of colour television to the terrestrial scenario allows good understanding of why it is only recently that work has taken place in earnest to develop affordable systems to send video data along telephone lines for the purposes of videoconferencing. In its uncompressed state, broadcast-quality digital television needs bit rates of over 100 Megabits per second - well over that available for even the newest Integrated Services Digital Network (ISDN) channel, working at 64 kbits/s. Hence, modern videoconferencing hardware is designed to analyse picture information and prevent repeated transmission of the same picture information.

Consider a familiar situation where a newsreader appears on a television screen. Whilst the news is being read, most of the picture will not change other than, say, the lips, eyelids and occasional body movements. This fact can be used to good effect, such that only information about *differences* that have occurred will need to be sent to the receiver. This process is called *interframe coding* and is ideal for the head-and-shoulders data often associated with the conversant parties in videophony. It is also possible to extract information about differences between adjacent pixels on the *same* frame at a given instant. This process of *intrafield* coding makes large areas of consistent colour and shade (the plain background in the newsreader example) easily represented by extracting information about the boundaries where significant changes in luminance and chrominance occur. These two methods of data compression have been used to good effect in developing hardware to send videoconferencing pictures and sound over single telephone channels at 64 kbits/s.

At an early stage, the telecommunications industry in Europe identified the need for close collaboration to ensure the adoption of a system which could be applied in all countries and make videophony available to a world market. Even though a European standard specification did emerge in the 1980's for a 2Mbits/s, 625 line, 25 frames per second PAL system, demand on the other side of the Atlantic required plans using the 525 line, 30 frames per second NTSC system. Eventually, conversion between these two video standards became the focal point of international co-operation and under the auspices of the International Telegraph and Telephone Consultative Committee (*CCITT*), a videophone algorithm was recommended, meeting the needs of new ISDN systems and working for all bit rates between 64kbits/s and 2Mbits/s.

The resulting CCITT *H.261* algorithm, as it is known, forms the basis of this paper and although intended for videoconferencing systems under the new ISDN network being installed throughout the world, many of the concepts employed are equally applicable to other areas of broadcast television. In particular, the technology behind high-definition television makes use of many of the coding principles involved, since an increased amount of picture data is to be transmitted within the constraints of existing terrestrial bandwidths.

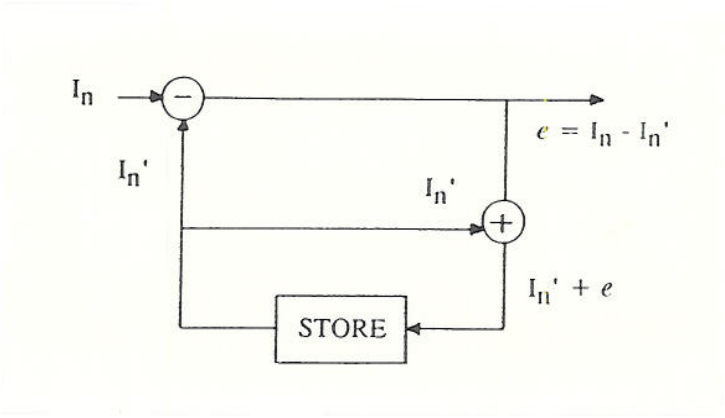
2. Principles of Video Codec Design

2.1 Differential Pulse Code Modulation (DPCM)

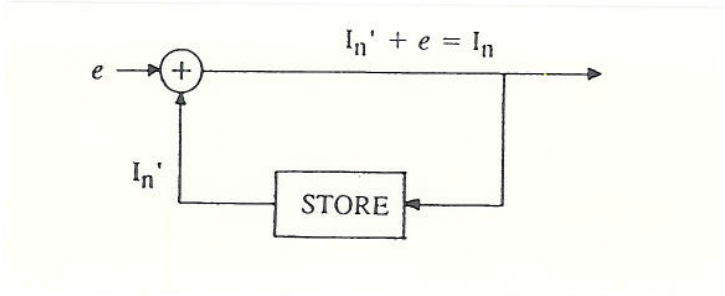
At the heart of the videoconferencing algorithm recommended by CCITT Recommendation H.261, is the use of Differential Pulse Code Modulation, or DPCM. Whereas conventional Pulse Code Modulation uses a fixed-length set of binary codes to represent temporally-current data, DPCM extracts difference data between adjacent temporal data sets. The resulting information can then be transmitted using varying-length codesets, where the least number of bits are used to represent the most frequently-occurring values and the most number of bits the least-frequently occurring values. Practically this would mean that if most of the scene being transmitted was constantly the same colour and luminance, the relevant chrominance and luminance data would be frequently used.

The concept of differencing implies speedy processing of picture data to minimise delays and is facilitated by the availability of fast bipolar integrated circuits. As well as considering temporally adjacent picture frames and extracting difference data (*interframe coding*), the technique recommended by H.261 incorporates *intrafield* processing and by employing conversion of spatial data into the transform domain, is often called a *hybrid* model - optimised to use minimal bit rates for the reproduction of satisfactory picture quality.

To explain the principles behind the DPCM process, consider the simple loop of figure 1.



(a)



(b)

Figure 1: A simple DPCM loop showing (a) encoding for data transmission and (b) decoding for reception.

As can be seen, the use of a store to hold current frame data on the DPCM loop allows updating of that store by only sending a difference signal e to the decoder. Initially, the value of a given pixel parameter held in store is subtracted from its 'new' value and hence the current frame values are being subtracted from the next frame values. The result is a difference signal e, which can be transmitted to the decoder. At the decoder, another frame store holds the current values and is updated by simply adding the difference values to produce a representation of the next picture frame.

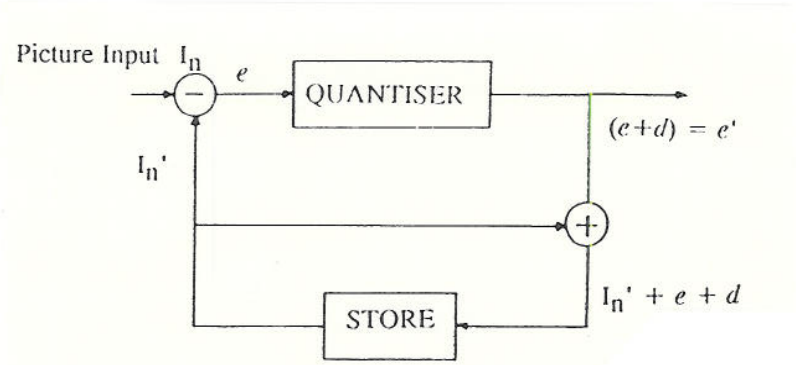
Algebraically, if I_n is considered the next frame value, I_n' the current frame value and e the difference between them, then:

$$e = I_n - I_n'$$

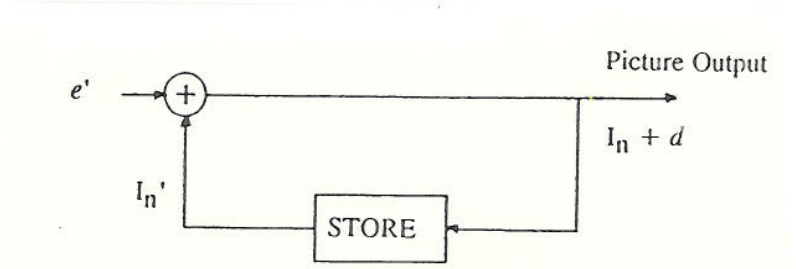
and $$I_n = I_n' + e$$

hence the next picture frame can be produced at the decoder.

One key feature of the DPCM/DCT model is the incorporation of quantisation to reduce the large number of values for chrominance and luminance that are possible to a quantised set, requiring less bits for data handling and storage. In doing so, distortion is added to the picture and whilst this is hardly noticeable up to a point, coarse quantisation can produce a very poor reproduction of the original picture. Adding a quantiser to the simple DPCM loop encoder is shown in figure 2.



(a)



(b)

Figure 2: A simple DPCM quantised loop showing (a) encoder and (b) decoder

The distortion is carried through to the output of the decoder and can be described algebraically as:

$$e' = I_n - I_n' + d = e + d$$

thus $I_n + d = I_n' + e'$ at the decoder

where d is the distortion produced by quantisation
and e' is the distorted difference signal produced.

As will be described later, the H.261 algorithm suggests a method of automatically changing the quantisation step to maintain a regular flow of data along the 64kbits/s channel

2.2 The CCITT H.261 Algorithm

During the early days of videoconferencing system design, the complexity of picture coding algorithms made it clear that building real-time hardware to prove design concepts would be prohibited by the vast amount of work involved. Hence, during the process of international collaboration, extensive use was made of flexible software-based simulation systems - ideal since once the data had been downloaded into bit form, it could easily be processed by a Mainframe computer. A video sequence of about thirty seconds real-time data can sensibly be stored and the results time-scaled, reflecting the increase in performance that dedicated hardware would produce.

The first stage in the process is to digitise a sequence by assigning for each pixel a value for both luminance and chrominance, storing the results in random-access memory. The video luminance signal is sampled at 6.75 MHz and digitised to a resolution of eight bits per sample; a similar process applies for chrominance, but at 3.375 MHz (half the luminance rate). With data then available, the process of data compression and coding can commence. A simple block diagram of the coding method is shown in figure 3.

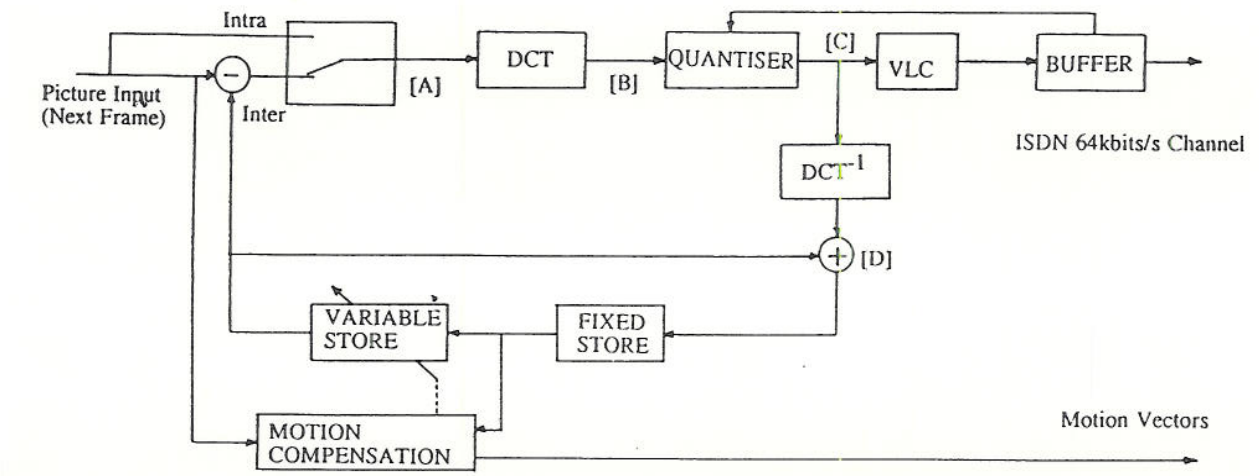


Figure 3: A simple block diagram of the H.261 hybrid DPCM/DCT encoder

2.3 Selecting Spatial or Temporal Compression

Primarily, a need exists to determine whether the video data compression is to take place between adjacent frames (temporally), or between adjacent spatial elements in a given frame i.e. interframe or intrafield respectively. The H.261 algorithm uses a process to distinguish between the two and where there is little change between each frame, the temporal interframe mode is employed, comparing the current frame data in the store with the next frame and extracting only difference information. However, there will be occasions when the scene changes very significantly - either due to a sudden movement of a large object across the scene, panning, or a *wipe* to a new scene altogether. In this case, intrafield coding must be used and the codec will operate to encode difference information between pixels on the new frame. The switch to intrafield inevitably results in a large surge of new data for encoding - providing the stored values against which (when reverted back to interframe mode) subsequent temporal differences can be compared.

Normally the switching function between the two coding modes is based on a threshold, such that if the interframe difference value exceeds a given level, the codec will switch to intrafield mode to extract all the pixel luminance and chrominance values for the new frame, subsequently encoded. For the purposes of consistency, this paper will assume the use of interframe coding.

2.4 Transform Coding

With the frame-temporal differences available at point [A] (figure 3), work can commence on data compression. After differencing, the picture is divided into 8x8 pixel blocks and a discrete cosine transform (DCT) applied to each block.

Transform coding does not directly cause data compression - there is still an 8x8 array of coefficients in the transform domain. However, data compression is much easier if a good transform method is used. Following transformation to the DCT domain, the coefficients in the 8x8 block can be considered as the magnitudes of the various spatial frequencies present in the pel domain. Consider a block in which all the pixel values are identical, as is the case in plain areas of a picture, then after transformation the only non-zero coefficient will be the D.C. coefficient in the top left-hand corner. Hence this block could be said to have been compressed by a ratio of 64:1. Unless the frame is completely blank, there will be many blocks having more non-zero transform coefficients, but in almost all there will be a large number of values sufficiently close to zero to be either not transmitted, or represented by values of less than eight bits.

The transfer function of the DCT is given by:

$$F(u,v) = \frac{1}{4}\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\ \cos[\pi(2x+1)\frac{u}{16}]\ \cos[\pi(2y+1)\frac{v}{16}]$$

with $u,v,x,y = 0,1,2,...,7$

where x,y are spatial co-ordinates in the pel domain
and u,v are co-ordinates in the transform domain.

Whilst the DCT is considered the most efficient transform method available, less sophisticated methods such as the Hadamard transform can be used with limited success in more basic applications. Normally, the magnitude of the variances in the transform domain can be related to the frequency of the pixel values in the original 8x8 pixel block, as shown in figure 4.

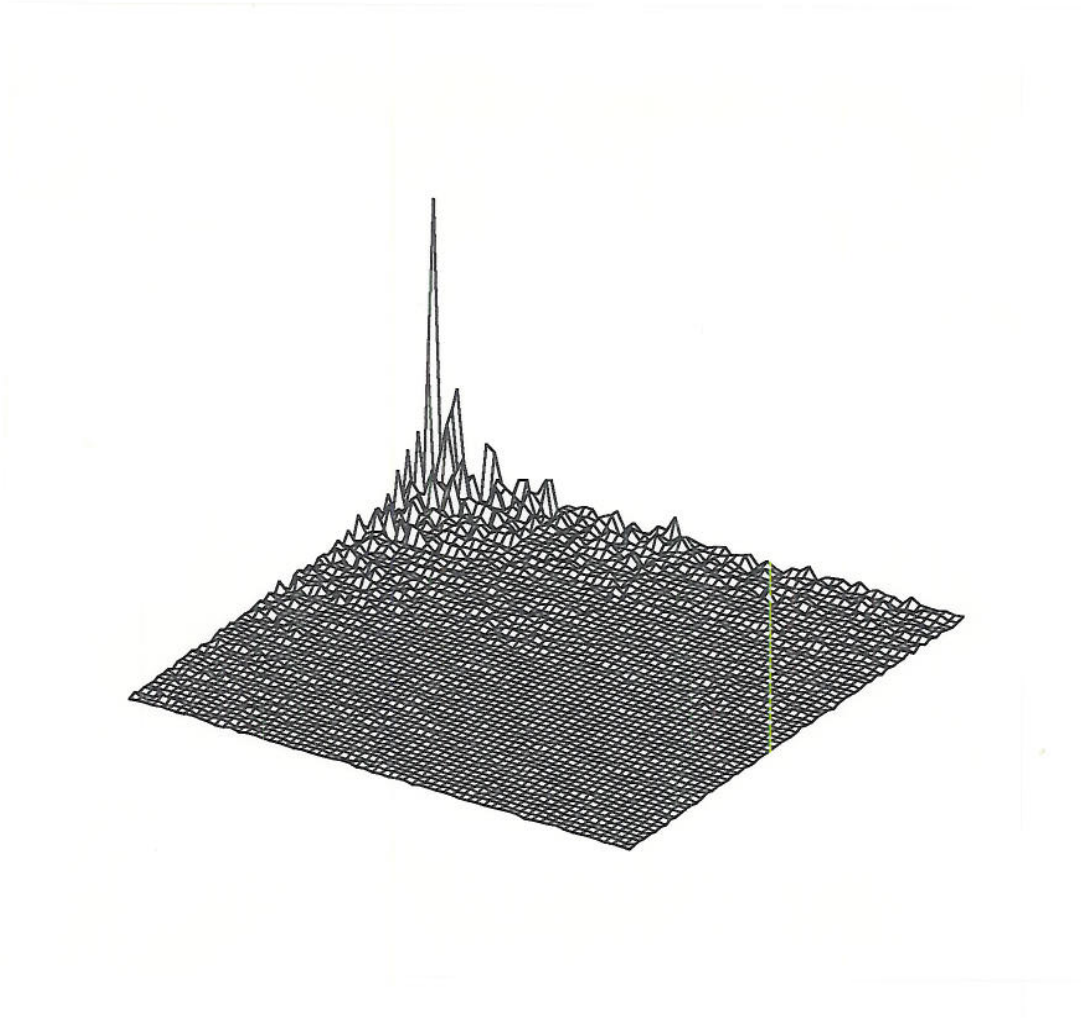
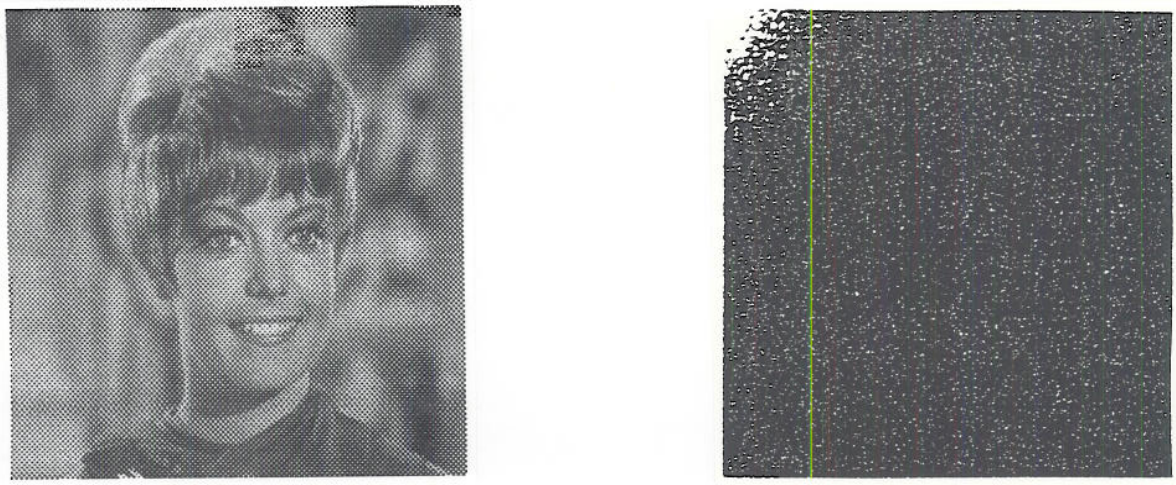


Figure 4: Transform domain variance distribution

Since we normally say that humans cannot readily detect sharp image transitions, the lower variances (representing the high-frequency elements in the 8x8 block) can be discarded allowing subsequent data compression. Figure 5 shows the effect of applying the discrete cosine transform to the whole of a given scene - notice how the higher coefficients are grouped into the top left-hand corner of the transform domain.



(a) (b)

Figure 5: DCT of (a) pel domain image using (b) clipped magnitude.

2.5 Quantisation and Data Compression

With data in each 8x8 block now reduced to a set of coefficients in the DCT domain (point [B], figure 3), the compression which follows makes use of the transform coefficient statistics already known - i.e. that each 8x8 block can be represented by a few transform coefficients grouped in the top left-hand corner of the 8x8 coefficients in the transform domain, representing D.C and a few low frequency coefficients. Quantisation is then performed on the transform coefficients, setting all small values to zero and quantising all larger values to a set of preferred magnitudes ready for coding and transmission. The quantised transform coefficients (point [C], figure 3) are then scanned in a zigzag manner, shown in figure 6.

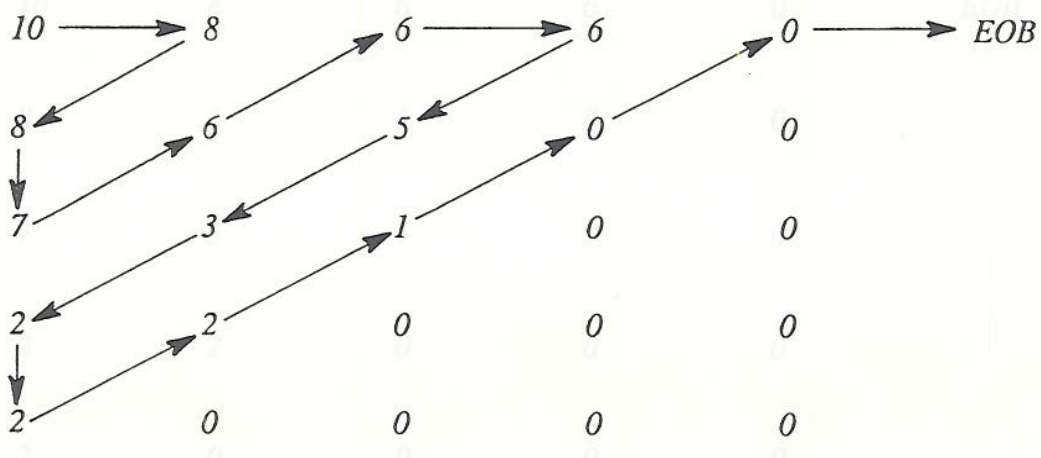


Figure 6: Codec scanning of coefficients, starting at D.C. and ending with EOB.

The quantised values are scanned in an order which indicates descent from the highest transform coefficient to the lowest. However, the effect of quantisation produces a large quantity of zero coefficients. Their presence makes data compression possible, since no need exists to transmit long streams of zero values. Instead, once three or four zero coefficients have been encountered by the scanning process it stops and sends an End of Block (EOB) code, marking the end of all non-zero coefficients for the current 8x8 transform block. In the example of figure 6, sixteen codes would be needed (including the EOB character), representing a compression of about 25% from the original set of transform coefficients.

2.6 Methods of Data Representation by Low Bit-Rate Coding

Now that data compression has been achieved by first taking interframe difference data and then quantising the transform coefficients of each 8x8 block, a set of codes requires definition to correspond to the quantisation values (which will be predetermined) taking account of the frequency of occurrence for each value. Practically, this is achieved by the use of variable length coding (VLC), where codes of the least bit length are used to represent the most commonly occurring coefficients, the bit length increasing as the frequency of coefficients decreases.

The use of variable length codes removes the redundancy that would occur if a fixed length word of, say, eight bits was employed. Generation of VLCs is related to probability and the method of producing the required sets is shown in the binary tree of figure 7. Starting at the root, a left-hand branch at each node adds a 0 to the code and a right-hand branch adds a 1. The assignment of these codes requires information on the probability of occurrence of each scanned quantised transform coefficient.

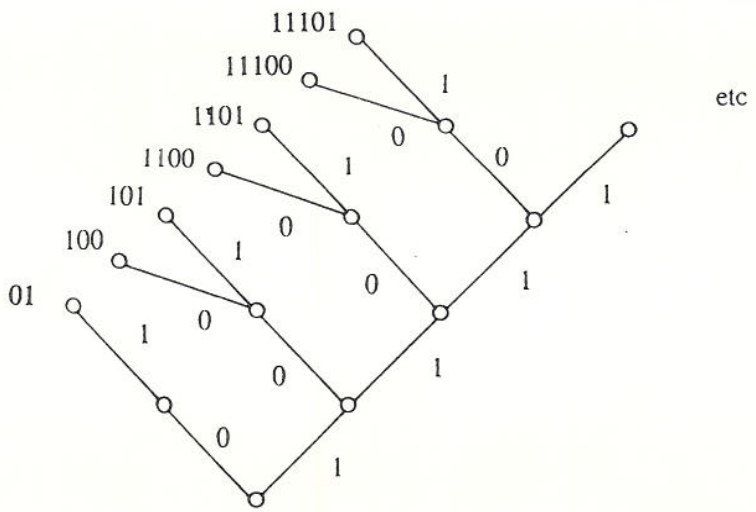


Figure 7: Binary tree representation of VLC generation.

Consider the following example. If a set of 100 coefficients is known, the probability of occurrence of each value can be derived.

e.g.: Value	Frequency	Code	Bits
12	40	01	2
11	15	100	3
13	14	101	3
10	12	1100	4
9	11	1101	4
8	8	1111	4

By multiplying the number of bits required to represent each value by the frequency, it can be seen that 291 bits would be needed to represent this sequence of 100 coefficients. Had four bits been employed for each value, 400 bits would have been needed and so the benefit of variable length coding is shown.

An algorithm for deriving a set of Variable Length Codes was proposed by Huffman (1952). The compact code can be constructed by first ordering the input probabilities according to their magnitudes, as illustrated in figure 8 for six input values. The two smallest probabilities are added on a step-by-step basis to form a new set of probabilities until only two probabilities are left.

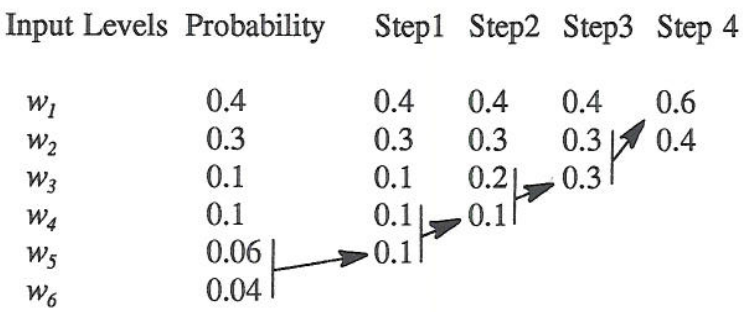


Figure 8: Construction of a Huffman Code

Code words are generated by starting at the last step and working backwards, assigning a 0 and a 1 at each step, decomposing probabilities. The 0 associated with 0.6 remains the first bit of its decomposed codewords and the 1 associated with 0.4 remains the first bit of 0.4 in each step back. When the input probabilities are reached, a compact code will have been generated to reflect the frequency of occurrence (figure 9).

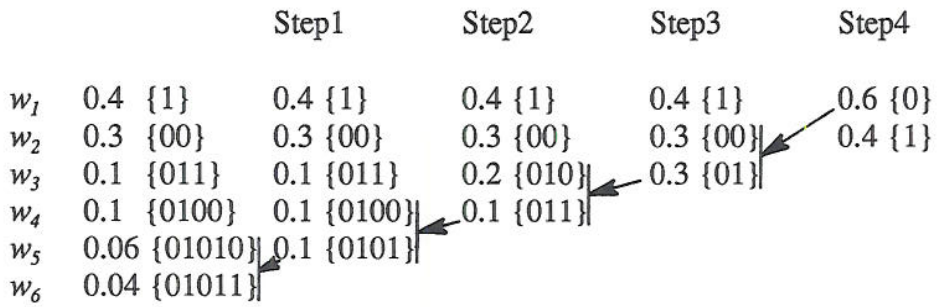


Figure 9: Construction of Huffman Codewords

2.7 Video Data Buffering

Prior to transmission along the ISDN channel, a buffer is employed (figure 3) to ensure that a constant flow of compressed data exists at 64kbits/s. As has already been shown, the resulting bit stream is very variable, with a high rate in areas of unpredictable motion or new scenes and a low rate in stationary scenes. The effect of the buffer is to smooth out fluctuations in bit rate to produce the constant stream required, so that whilst the buffer input could be anything between 0 and 384 kbits/s, its output is always 64kbits/s.

Although a buffer does have the capacity to store data on a first-in, first-out basis, this feature is limited by the technology currently available in that larger propagation delays in video transmission are inevitable as buffer capacity is increased. So as well as striving for improvements to the capacity and accessibility of buffers using fast bipolar technology, a method of controlling the flow of data out of the coder has been applied and is shown in figure 10.

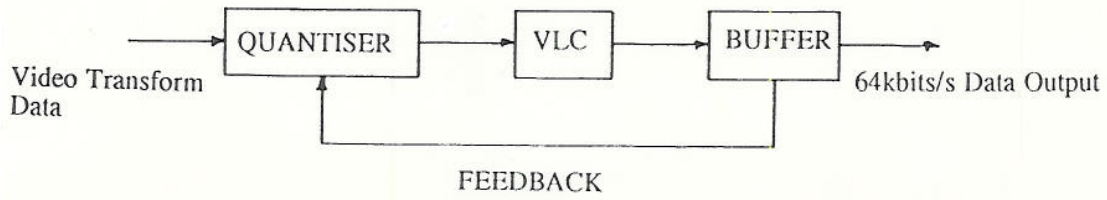


Figure 10: Using quantisation step control feedback to produce constant output bit rate but variable picture quality

The presence of a feedback path enables the buffer status to affect the coarseness of the quantisation intervals within the encoder. As the buffer fills with data (particularly during significant interframe motion), the effect of the feedback is to coarsen the quantisation step employed and thus reduce the data input rate to the buffer. The main drawback to this principle is that as the buffer reaches capacity and the quantisation interval is at maximum, picture quality can be very poor. Even though the loss of quality is only temporary (subsequent interframe data correcting poor resolution as the picture settle), its presence has made the use of this coding mechanism unsuitable for broadcast-quality television, where high transmission quality is required all the time.

In a videoconferencing situation where the video signal is frequently switching between the conversant parties, picture distortion is quite a problem because of the buffering factors mentioned. Whilst large buffers, when practical, will reduce the dependence on quantisation step control, more complex methods of communication using packet video signals over a broadband ISDN system appear to be the most feasible solution to this problem.

2.8 Storage

In the simple DPCM loop of figure 1, the incorporation of picture feedback and storage allows *current* picture data to be stored in the codec for subsequent comparison with the next temporal frame. Having passed video data into the transform domain and performing quantisation, the coefficients must be transformed back into the pel domain to facilitate *like for like* differencing in spatial terms.

An inverse DCT function is employed to re-form pel difference data from the transform coefficients at point [D] on figure 3, where the differences are added to the current frame value in the fixed store to update the retained frame to that of the new frame undergoing coding.

2.9 Motion Compensated Prediction

Although the compressed data produced by coding the interframe difference signal may well provide an efficient method of picture transmission, the performance of interframe coding can be significantly improved by the use of motion compensated prediction. If movement has occurred in a given block, then application of a simple linear translation can re-map objects in that block on the next frame. For example, an ear may move slightly and could be translated, providing no significant change in orientation has occurred.

The block-based method used in conjunction with the H.261 algorithm is simple and can be speedily executed by the hardware in the codec. A search window of 24x24 pixels is defined, equivalent to the area occupied by nine 8x8 pixel blocks. Then for each current 8x8 block within

the window, a scanning process is used to assess the likely source within that window of the present block from the previous frame, from which a motion vector can be derived.

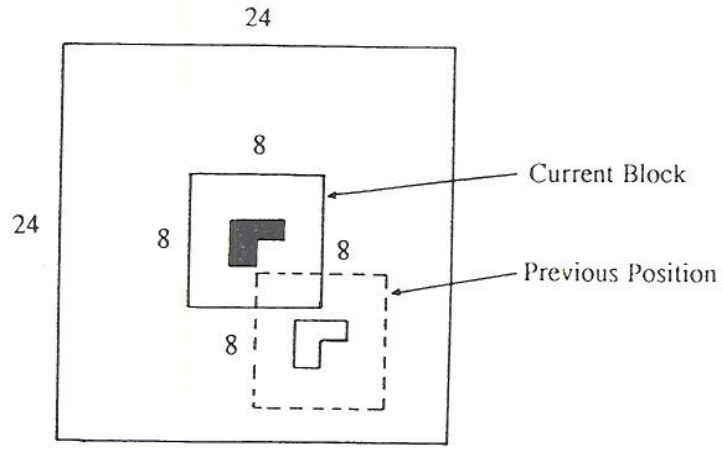


Figure 11: Scanning process for motion compensated prediction

Consider the example shown in figure 11. The current position of the cornice shape is in the central 8x8 pixel block within the search window. To generate a motion vector, the prediction algorithm must compare the pixel values in this current block with *all* possible 8x8 pixel block positions in the corresponding search window within the *previous* frame. This is carried out by taking a sum of squared error pixel values for each previous frame trial block and subtracting this from the sum of squared error picture values for the current 8x8 pixel block being employed. The position in the previous frame search window where the minimum error sum exists can be considered the most similar to the existing block and hence the motion vector can be calculated (figure 12).

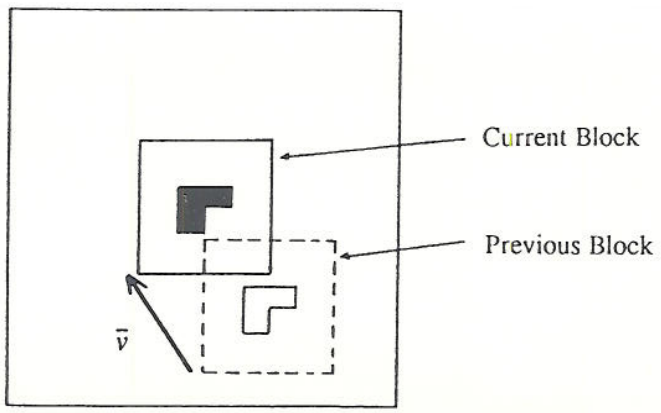


Figure 12: Generating a motion vector

Hence the summation algorithm employs the following function:

$$\sum e^2 = \sum_{x=0}^{7} \sum_{y=0}^{7} |P_c(x,y) - P_p(x,y)|^2$$

$\sum e^2$ **is the total e value for each 8 x 8 block**

P_c is the 'current' pixel value for the search template block
P_p is the 'previous' pixel value for the corresponding location

Whilst simple, the process of motion compensated prediction requires a significant computational overhead to perform the large number of calculations involved to compare each existing block with all the possible original locations in the 24x24 pixel search window. To demonstrate this, we could say:

1. There are 17x17 possible search positions (x,y) = 289 positions

2. Calculation uses 64 multiplications and 64 subtractions,
   hence 128 operations x 289 search positions = 36992 Operations

3. Each frame has 400 search windows (400 x 36992) = 14 million operations per frame

4. and 30 frames per second makes about 480 million operations per second !

With a motion vector v (x,y) determined, data can now be transmitted to the receiver to allow new picture reconstruction. Motion compensated prediction always provides a vector to show the origins of current blocks, rather than stating their destinations, hence:

$$\dot{I}_n = f(I_{n-1})$$

Once displacement vectors have been derived, these are transmitted to the receiver, together with the transform coefficient values for the difference picture (figure 13). Using the motion vectors, pixel data held in the fixed store is adjusted to conform to new block locations prior to picture decoding. A similar process is used for modifying picture data held in the frame store of the transmitting codec (figure 13).

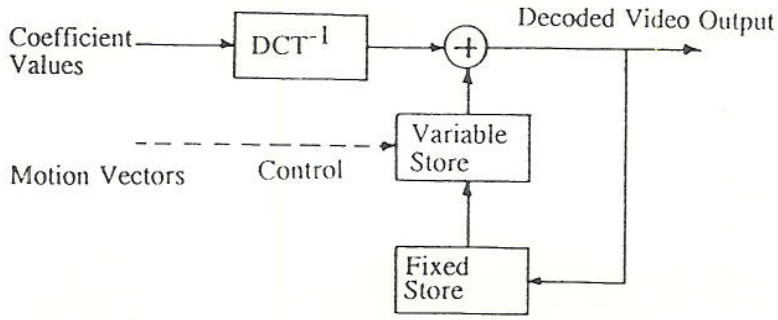


Figure 13: Simplified block diagram of DPCM/DCT Decoder

3. Video Standards

At the heart of the need for international collaboration in videoconferencing systems, were the conflicting requirements of the two video standards used. The 625 lines per picture at 25 pictures per second *(625/25 PAL)* system used in Europe needed to be interfaced with the 525 lines per picture at 30 pictures per seconds *(525/30 NTSC)* system used in North America and Japan. A number of options were considered to overcome this difficulty, such as designing codecs to receive both standards, but only transmit using their local TV standard.

In the end, the best option was to employ a *Common Intermediate Format* (CIF) solution, such that all codecs in both PAL and NTSC have pre and post-processing hardware to convert to the 288 non-interlaced lines per picture, at 30 frames per second format upon which CIF is based. The result of this is that all codecs use the same video communications protocol and have identical core designs, allowing manufacturers to produce hardware for sale around the world.

As CIF produces good vertical resolution, about 30% down on studio-quality video, it is ideal for videoconferencing. For some applications, such as face-to-face videophony, such resolution is not needed and the H.261 recommendation incorporates a secondary picture format, having 176 horizontal luminance samples per line and 144 lines per picture - known as *Quarter Common Intermediate Format* (QCIF).