VIDEO COMPRESSION METHODS

Of the many technical and scientific marvels which the second half of the twentieth century has witnessed, arguably none has had such a pervasive (or indeed enabling) effect as the explosive growth in the dissemination of information made possible by remarkable developments in the areas of communications and electronics. Of the five human senses, sight gives us the greatest ability to respond to a hugely varied and rapidly changing environment. The coupling of vision and information dissemination accounts for the phenomenal influence afforded by the ability to reproduce pictures at a distance. Successful evolutionary development has depended far more upon our ability to react to objects that are in motion rather than stationary; by extrapolation the transmission of *moving* pictures is of supreme importance to, and exercises a profound influence over, our present culture. Such capability has traditionally been provided by analog broadcast television, now being superseded not only by digital formats but also through the use of alternative transmission implementations—cable, the Internet, and others. The remarkable impact of presentation of moving, color pictures of actions taking place thousands of miles away is expensive, however, as a result of the capacity required for modern transmission systems. Roughly speaking, analog television always required a bandwidth some one thousand times greater than that necessary to transmit a reasonable quality speech signal (a few megahertz as compared with a few kilohertz), and the internationally agreed sampling rate of 13.5 MHz for digitized luminance information together with one half of this value for the two color signals, and a word length of eight bits leads to an overall rate of over 200 Mb/s per digital television signal (1), compared with 64 kb/s for a digitized speech channel. Although not all forms of video transmission require the same picture resolution as broadcast television (with high definition television requiring significantly more), the development of digital versions of currently existing analog services together with the continual introduction of new digital applications of all kinds means that the reduction of required channel capacity to the minimum needed for a given quality of service is a most desirable goal.

How, then, are we to go about defining what can be “left out” of the coding and transmission process in order to minimize the required data rate? (Note that it is only with the development of readily available high-speed digital technology that the complex processes necessary have become possible, basically through the comparative ease with which it is possible to store information in digital form.) There are two major factors that come into play here—the physical structure of the signal itself as picked-up by the video camera and the properties of the human eye, which determine what the viewer will see in the finally displayed image. It is instructive, first of all, to consider these matters with respect to still images. Think of a reasonably detailed image field (say a holiday photograph) and remember that typically, in digitized form, there will be some 700 data points horizontally and 500 vertically.

Somewhat surprisingly, we need not pay much attention to the color information within the image because it turns out that this can be processed in just the same way as the luminance (brightness) information—no new ideas are involved and the necessary data rate turns out to be quite a bit lower anyway. Thus, the picture consists of regions (sometimes quite large–background, sky, significant objects) of substantially uniform brightness, with object detail—the car, house, ship or whatever delineated by distinct borders (edges or sharp transitions). At the resolution given, even small objects frequently comprise several dozen very similar data points. A major feature of any video compression scheme is recognition of the similarity of neighboring picture elements and the use of this property to avoid transmitting each successive element in a picture scan at full (8 bits in our example) luminance resolution. Ways of doing this are detailed in what follows.

The other significant features of the image are those sharp transitions that exist between recognizable objects and their backgrounds. These are not nearly so easy to deal with but we can nevertheless still devise adaptive schemes that can vary their parameters to cope with this situation. What, then, of the response of the human eye to the image of the scene as reproduced by the system? It turns out that the eye is maximally sensitive to fairly small objects portrayed at the resolution of our example on a screen set at a reasonable viewing distance (say 4 to 6 times picture height). Larger objects, perhaps covering one-eighth of the screen or more produce somewhat less response and much smaller ones also substantially reduce sensation. We can thus afford to code very fine detail changes less accurately or, in some cases, ignore them completely, thereby achieving a reduction in the channel capacity necessary.

Another feature of the eye’s response that can be made use of in image and video compression schemes is that of the nature of the response as a function of the magnitude of the originating stimulus. Both the eye and the ear can usefully operate over an amplitude range of many orders of magnitude. This cannot be achieved by a linear transducer and so the perceptual response is logarithmic, implying that the annoyance value of a given error is not an absolute quantity but depends on its size as a fraction of that of the signal in error. Basically this means that large luminance changes can be coded with relatively large error for the same perceived imperfection as a small error in a region of substantially constant luminance. At the same time the existence of visual “masking” (the “covering-up” of a luminance error by a nearby large step-change or discontinuity) means that a small error in the vicinity of an edge will also be of reduced perceptual significance. It is worth making a general comment here. All techniques which achieve significant degrees of image or video compression do so by either omitting detail or coding it only approximately. The trick is to do this in such a way that the reconstructed picture is as little affected as possible. Extensive subjective testing of coding schemes is therefore necessary on large, representative, sets or sequences of data to ensure that the final result is acceptable. This is most important where the setting-up of worldwide standards (as has been the case over the past ten years or so) is concerned.

There is a third factor that contributes to the efficient operation of a digital video coding (compression) scheme; it relates to the final output data stream, which takes the form of a coded sequence of symbols ready to be passed to the final stage of the processor to be converted into a signal suitable for output to the channel (being correctly formatted, having error control information added, etc.). It turns out that the various symbols of the data stream are not used with equal probability (imagine coding a large, more or less uniform area, followed by a small transitional region) and we may thus benefit by arranging for the more frequently occurring symbols in the processor output to be converted to short codewords for transmission, and vice versa, resulting in so-called variable word-length coding (2,3); a good analogy here is that of the Morse Code, in which the most frequently occurring letter of the alphabet, e, is coded with the shortest output symbol, one dot. Doing this enables further useful gains in efficiency to be achieved.

So far we have restricted the discussion of compression ideas to source material consisting of still images. What now do we need to do to apply such ideas to video transmission, which, approximately at least, may be considered to be sequences of such still frames, presented to the eye at such a rate as (via the phenomenon of *persistence of vision*) to produce the illusion of image object motion? As the output of the image data processor will consist of a single symbol stream, whatever the format of the input, the third technique referred to in the preceding, variable word-length coding, is straightforwardly applicable in the video context without further comment. As far as the physical properties of the image sequence are concerned, the same principles apply, except that we now have an extra degree of freedom by being able to operate along the time axis also, relating the characteristics of one frame to the next. This is called *interframe* coding. It is, of course, possible to process the separate frames of a sequence (and, necessarily, individual still images too) on a one-by-one basis, that is, without making use of the properties which interrelate them. This is called *intraframe* coding.

Although there are many techniques for taking similarity properties into account, some of which will be described in what follows, two are so common and well-researched as to have been incorporated into the standards developed in the 1980s and 1990s—prediction (4) and transformation (5). In the first, use of the similarity property between neighboring picture elements allows us to make a prediction of the value of one element from those nearby, which have been previously processed on the same line, on the previous line or, indeed, in the previous frame. This latter case is particularly important, because for sequences in which the movement or action is not too violent, similarity from frame to frame can be very great and a prediction of a given element in one frame from the element in the "same" physical location in the previous frame is likewise efficient. Even the inevitable degree of motion (which, after all, makes moving pictures moving) can be allowed for by a technique known as motion compensation. Here we rely on the fact that an object changing its location between frames does so either by translation (simple lateral, diagonal, or other movement without other change–rotation perhaps) or, if the motion is more complex, it can be approximated on a small scale by such translation anyway. We then predict not from the same physical location in the previous frame but from the same location in the object, making a note at the same time of how much motion there has been from frame to frame in the form of a motion vector, which must be passed to the receiver. In this way an efficient prediction may be achieved and the prediction error (the actual value of the picture element minus what we predict it to be) is kept small. As the generation of the prediction proceeds sequentially at both coder (transmitter) and decoder (receiver), all we need to send now, together with the motion vector, is the error signal which, when added to the prediction produced at the decoder regenerates the value of the picture element.

Although the prediction process (in one, two or three dimensions) is all that is needed to produce a reasonable amount of video compression in practice it is found necessary to incorporate further, powerful means of efficient processing into the system. The other major technique is called transform processing or coding, which likewise employs similarity properties within the picture data stream but in a somewhat different way. Without introducing mathematical complexities into this introductory section it is intuitively obvious that any data containing large stretches of elements with similar values can be viewed as containing only low spatial frequencies (i.e., amplitude variations across or up and down the picture, analogous to more usual variations in time, which are defined by conventional frequency). Thus the detail only varies slowly as our viewpoint moves from one side of the screen to the other. On the other hand many rapidly varying values imply the presence of significant amounts of high frequency information. It turns out that image data does have, for the vast majority of the time, substantial low frequency content and relatively few large high frequencies. By operating on this frequency representation we can thus take care to process the former accurately while not being too particular about the latter. This is an extremely powerful technique in its own right and can be used to compress still images by factors of 10 to 20 : 1 while retaining good quality. Like the prediction operation it can be used on its own in one, two or three dimensions on video sequences, but it has found specific use when applied to the output of the frame to frame prediction operation described previously. This "hybrid" coding operation has become the cornerstone of all algorithms developed as video sequence coding standards since the mid-1980s.

HISTORICAL NOTE

Coding of still image data and video sequences has a longer history than is often supposed. Prediction of video material was investigated in the 1950s, following theoretical work on prediction carried out in the previous decade. Forms of variable length coding together with techniques allowing the construction of efficient codes date from the same period. Transformation of image data was first developed in the 1960s (basic notions of the transform processing of data coming into being some ten years earlier), and the study of the effects of motion on the processing of sequences and how to compensate for it dates from the 1970s. From that time also a multiplicity of other techniques for efficient image sequence processing has been developed, the most important of which are described here.

BASIC TECHNIQUES FOR VIDEO COMPRESSION

Prediction

In many areas of human activity the idea of prediction—attempting to make an estimate now of the likelihood of the occurrence of some possible future event or, more relevant to science and technology, of some future numerical value, has fundamental significance. In order for such an operation to succeed on anything more than an accidental basis, however, there must be some sort of regularity or structure within the process involved; few people make their livings out of predicting the outcomes of card games or horse races, and none out of applying prediction to the result of a lottery! This obvious requirement turns out to be satisfied to a surprisingly high degree where image compression is involved, and allows the design of a spectrum of coding schemes which will, at the one end, allow the transmission of digitized television material having source rates of hundreds of megabits per second over systems having bandwidths of only a few megahertz and, at the other, the transmission of low resolution videophone signals at rates of a few tens of kilobits per second. What is involved, then, in the application of prediction to image compression? The basic example of prediction occurs along a single image line, and takes the form of a weighted sum of previously scanned elements. Thus, if we label successive image elements as $I(n-3)$, $I(n-2)$, $I(n-1)$, $I(n)$, $I(n+1)$, and so on a prediction $I_p(n)$ of $I(n)$ will be of the form

$$I_p(n) = aI(n-1) + bI(n-2) + \cdots \tag{1}$$

where, somehow, we shall have to determine the weighting coefficients a, b etc., to make the prediction as efficient as possible. As far as still images are concerned, prediction is usually carried out using three or maybe a maximum of four nearby picture elements. It requires no mathematics at all to establish the principle that, to make a prediction as accurate as possible, it pays to stand as close to the value to be predicted as you can get, and so it is usual to include picture elements on the previous line also (recall that, given the conventional left to right and top to bottom line scan of the television process, and the fact that we can only predict from values we already have, to make up the prediction we only have available elements on previous lines and to the left-hand side of the element being predicted on the present line). Thus, if we label successive lines as $(m-2)$, $(m-1)$, m, and so on, a more general prediction could be

$$\begin{aligned} I_p(m,n) = {} & aI(m,n-1) + bI(m,n-2) + cI(m-1,n-1) \\ & + dI(m-1,n) + eI(m-1,n+1) \end{aligned} \tag{2}$$

where we have used the two preceding elements on the present line (m) and three symmetrically disposed elements on the previous line $(m-1)$. We now have five predictor coefficients to determine, which we do by noting that the prediction error (the difference between what the value of the predicted picture element actually is less the value of the prediction) is given by:

$$P(m,n) = I(m,n) - I_p(m,n) \tag{3}$$

Conventional prediction theory now minimizes the average value of $P^2(m,n)$ with respect to the various coefficients by setting the appropriate partial derivatives to zero to give us the optimum minimum mean square error prediction. It turns out that both the coefficient values themselves and the value of the minimum error so obtained are functions of the values of the interelement correlation within the image. To pursue the analytic approach these values must be obtained by measurements taken over a representative class of test images. Interestingly enough, it happens that the interelement correlation is quite high even for images containing significant amounts of fine detail (note that such a property will refer statistically to the image as a whole, in the absence of any attempt to adapt the coder parameters to specific kinds of picture content).

Such a scheme as that just described is very simple to implement, which accounts for its popularity when video compression schemes were first being researched, given the limited processing resources then available and even today more complex variants are still being investigated. One or two other details of the basic scheme are worthy of note here, the first being the exact structure of the system. The decoding operation is carried out at the receiver by taking the (small) prediction error as transmitted through the channel and adding it to a prediction generated at the decoder (which should be an exact copy of that generated by the coder). Thus, ideally (see Eq. 3),

$$I(m,n) = I_p(m,n) + P(m,n) \tag{4}$$

There is a problem, however: $P(m,n)$ will have been digitally transmitted and thus contain an inevitable amount of quantization error $q(m,n)$:

$$P_q(m,n) = P(m,n) + q(m,n) \tag{5}$$

thus the operation actually carried out at the decoder is

$$I_r(m,n) = I_p(m,n) + P_q(m,n) \tag{6}$$

giving the reconstructed signal $I_r(m,n)$. In order that coder and decoder should "track" exactly, the system must be arranged so that the prediction at the coder is also based upon a reconstructed signal containing the quantization error.

Then, from Eqs. (3), (5) and (6)

$$\begin{aligned} I_r(m,n) &= I_p(m,n) + P(m,n) + q(m,n) \\ &= I(m,n) + q(m,n) \end{aligned} \tag{7}$$

and the reconstruction is the same as the input signal apart from the addition of the unavoidable quantization error component $q(m,n)$. Figure 1 shows the basic operation of predictive coding.

Another significant detail is the exact form of the output signal. It is a fact that the input image as a whole may have widely varying statistical properties in terms of brightness, color, texture, etc. (after all, just about anything can qualify as a subject for a television shot). All we know, in practice, is that it can never be negative and that it is likely to have a significant overall mean value, at least for a daylight or studio scene. In contrast, the output signal as produced by predictive processing is very well behaved. It has a very small mean value (nominally zero) and its distribution is very highly peaked about that value, approximating to the analytical Laplacian case. An idea of why this should be so can be gained by considering the nature of the predictive process in conjunction with the properties of a typical image as discussed in the preceding. Over the reasonably uniform parts of the image we naturally expect the prediction to be good and the error signal therefore to be small. For a simple prediction of the kind discussed so far, whose properties are fixed, edges

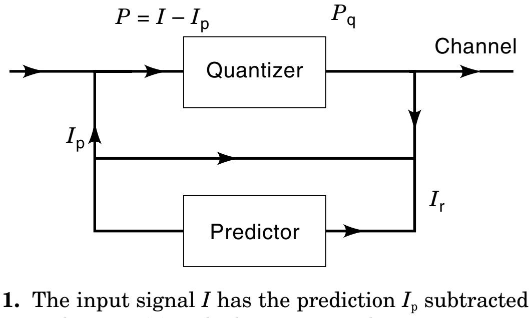

Figure 1. The input signal I has the prediction I_p subtracted to form an error signal P [Eq. (3)], which is quantized to $P_q = P + q$ [Eq. (5)]. I_p is also added to the quantized error signal (mimicking the decoder operation) to give $I_r = I_p + P_q = I + q$ [Eq. (7)], the reconstructed input signal save for the inevitable quantization term q.

(infrequent but often large) will, colloquially speaking, come as a complete surprise and therefore generate large prediction errors. Thus is the form of the error signal explained—many small values and a few large ones; this nonuniformity is significant in allowing us to achieve a reasonable amount of compression, for we can now quantize the signal with maybe only 10 to 15 intervals (fewer than 4 bits) compared with the 256 which our original 8 bit example contained. These intervals are narrow near the mean value (zero) to assure good reproduction of uniform areas, and wide for large values of error where, in the purely predictive coding process, visual masking helps to reduce their visibility anyway. Thus we have achieved our goal of compression, in spite of the paradoxical fact that the possible range of the error signal is greater than that of the input—an element to element transition from black to peak white (admittedly unlikely) produces a full range positive error signal and a white to black transition generates the reverse, giving a total 9 bit representation for an 8 bit input!

The scheme described here forms a simple basis for obtaining a moderate degree of data compression. It can be made more efficient by incorporating adaptation into either (or both) of the quantization and prediction processes. Adaptive quantization usually involves some sort of feature detection that can signal the presence of substantially uniform regions (for which closely spaced quantizer levels are appropriate) or those with rapidly varying luminance (when the quantizer levels can be much more widely spaced). Adaptive prediction frequently operates through the use of a set of different predictors on the basis of a comparison of the outputs of which the most efficient can be selected. Naturally, the decoder must know which quantizer or predictor to use, and this information must either be signaled to the decoder by the transmitter (so-called forward estimation) or, in more complex schemes, it may be derived from previously processed parts of the signal at the decoder only (backward estimation).

As far as video coding is concerned, the general predictive scheme [sometimes called differential pulse code modulation—(DPCM)] may be extended to three dimensions with the inclusion of a prediction term taken from the previous frame also (6,7). Such a scheme can provide moderate degrees of compression together with good reproduced quality, although, as might be expected, its performance is sensitive to changing image properties unless highly adaptive schemes are used. There are fundamental reasons why the predictive scheme cannot provide extreme degrees of compression, however, and thus its application to video coding is limited to a first step in the processing chain. The option taken is to make a simple prediction from the previous frame (interframe prediction), albeit with improved efficiency obtained through the use of motion compensation. In this case we may sometimes depart from the rules previously outlined as to the source of the prediction. Some video compression standards allow for what is called bidirectional prediction, in which case information is used for prediction that comes from frames that have yet to be processed. Of course such data cannot be used in its unaltered form, and such two-way prediction involves ordering the frame sequence so that the required information is actually available for the prediction.

Motion Compensation. Brief reference has already been made to the matter of efficiencies to be achieved in video compression by attempting to follow the motion of object detail from frame to frame. It has to be said that doing this accurately and consistently over a large number of frames and incorporating the results into compression algorithms is a task still in its infancy, but simple operations of this nature are included in commonly available compression systems. In any case, there is a complex trade-off to be considered when attempting to account for the motion of objects within a sequence. On the one hand, we could ignore all interframe motion and expend all channel capacity on coding the frequently large signals that resulted. On the other we could use up more and more on signaling to the decoder the details of the motion of objects within the scene, leaving less and less capacity for the actual object, which parameters would have been coded in a prior frame. It is obvious that the situation will be strongly dependent upon scene content and degree of motion and at present only relatively simple schemes are in use that are, nevertheless, capable of reducing the coding requirement by one-third or even one-half as compared with the noncompensated case.

What approaches are available, then, for motion estimation and compensation? Much work has been carried out since the late 1970s on the minimization of the interframe prediction error using recursive steepest descent algorithms, which are also able to allow for changes in incident scene illumination (8). It is also possible to use frequency domain techniques to estimate motion (9). The scheme that has become of major importance in practical video compression schemes so far, however, is known as block-matching (10). This is a type of correlation technique, and one which fits in well with more general block processing of image data to be discussed in the section on image transforms). A small block of picture elements, 8×8 or 16×16 in extent, is considered in the present frame and then, somewhere not too far away in the previous frame, there is a block of the same size that contains very similar luminance, edge, and other detail. This detail will in all probability be at the same location within the object that has moved in the interframe interval, but slightly modified, by translation, rotation, or scale change within the block. A (larger) search area is then defined in the previous frame and the present block superimposed on every possible location (may be 200 or so) within that region. A distance function is then decided upon, possibly mean square error or, to ease computing requirements, mean absolute error between all the elements in the block and those within the search area covered by them at any one displacement, and location at which a minimum in this function is reached is recorded. This location (of the present block relative to the area covered in the search window) is noted and sent to the decoder as a "motion" vector. The block of prediction errors now forms a motion compensated frame difference signal that can be further processed in ways that will be described. It is important to note that, because the only criterion involved is a minimum of some error function, the motion vector does not necessarily represent true motion as would be needed in a motion tracking exercise.

As can be imagined, the computational requirements of such a full-search method are onerous (especially as, with the use of interpolation, a further small advantage may be gained by moving to a resolution of one-half of a picture element) and this has led to the reporting of many reduced search techniques, usually involving an initial rough search that is progressively refined in multiple recursions. One successful way of doing this is to operate on a hierarchy of image planes of varying resolutions, generated by successive 2×2 or 4×4 averaging. An initial rough search on the lowest resolution level is then used as an initial estimate for subsequent improvement (11). Such methods are becoming increasingly unattractive, however, as improvements in computing power allow us to carry out full-search estimation over increasingly large areas that provide the only way to guarantee that the true minimum has been reached. There is a further application of motion compensation in video coding that is particularly useful at the lowest rates where videophone and videoconference data are transmitted. Because channel capacity is at an absolute premium here it is possible to drop every other frame (or even two frames out of every three) in the coding process and reconstruct them at the decoder before display. This turns out to be unsatisfactory if done simply by interpolation because of object motion but the result can be improved considerably if such motion is taken into account (12). This naturally requires reliable segmentation of the scene into object and background and also that proper consideration be given to areas both covered up and uncovered by the motion of objects between coded frames.

Transformation

Predictive coding is a data, or space domain, process, operating directly on the picture elements making up the image. It is an algorithm that, at least in its basic form, requires something to be transmitted to the decoder about each separate element and it is thus unable to attain very high levels of compression (and so transmit at very low data rates—significantly below one bit per picture element). As a very rough indication of what is involved, transmission of a so-called quarter Common-Intermediate-Format sequence (176 elements on a line with 144 lines forming the picture) at 15 frames/s over a 64 kb/s link requires that we allocate no more than $64{,}000/15 \times 176 \times 144$ = approximately 0.2 bit to each picture element. Such a value is quite outside the bounds of possibility for predictive coding. What is needed is a system that allocates coding bits to blocks of picture elements rather than single ones so that, in this example, we could allocate 0.2×64, that is, approximately 13 bits to an 8×8 image block (on average) to represent its detail. One powerful way we know of representing waveform shapes is through the use of the Fourier transform, although here it turns out that an alternative transform is more efficient and better suited to image processing. The principle is the same, however. The transform consists of a set of basis vectors, each of predetermined shape and representing, generally speaking, more rapidly varying detail as the order increases. When we multiply the basis set sequentially into a data vector we obtain a set of coefficients, each of which indicates how closely that particular basis vector is mirrored in the data. Thus, for an N length data vector $\boldsymbol{X}$ we have N, N element basis vectors as the rows of an $N \times N$ basis matrix $[T]$. The coefficient vector $\boldsymbol{C}$ is then given by

$$\boldsymbol{C} = [T]\boldsymbol{X} \tag{8}$$

For image data compression the transform now universally used is the discrete cosine transform (DCT) proposed in 1974 (13). It is defined in one dimension as

$$\begin{aligned} &T(m,n) = T(0)(2/N)^{1/2}\cos(m(2n+1)\pi/2N) \\ &m,n \quad 0 \rightarrow N-1 \end{aligned} \tag{9}$$

with $T(0) = 1/\sqrt{2}$ if $m = 0$, and 1 otherwise.

This transform has the advantage over the discrete Fourier transform in that it ameliorates the effects of discontinuities at the ends of data blocks, which occur when processing finite length sequences. As far as implementation is concerned, there exists a variety of extremely efficient fast algorithms (as in the case of the Fast Fourier transform) available for this purpose (14). We invoke the idea of data uniformity (introduced when discussing the prediction operation). A large proportion of picture data will be reasonably uniform over a data vector length (considering first of all the one-dimensional case) of typically 8 elements. In such cases the coefficient vector $\boldsymbol{C}$ will have one or two low-order terms of significant amplitude and the remainder will be small or even zero. We may thus allocate our available bits preferentially to those few significant coefficients and, upon inverse transformation (having filled other locations in the received coefficient vector with zeros), obtain an approximate reconstruction of the original input picture vector. Should the approximation be unsatisfactory in quality, more bits must naturally be allocated. Of course, blocks with significant amounts of fine detail will have more nonnegligible coefficients and will, in general, need more bits for satisfactory coding. There is thus a distinct trade-off between picture detail, number of bits allocated and output quality. This procedure is effective in removing the correlation that exists between picture elements but operates, in distinction to the predictive algorithm, in the frequency (or, more accurately, in a frequency-like domain, the Fourier transform being the only one that possesses a strict space/frequency correspondence). As there is just as much correlation vertically within the image as horizontally, it is advantageous to employ a two-dimensional transform in which the vector $\boldsymbol{X}$ in Eq. (8) now becomes a data block (matrix) and $\boldsymbol{C}$ an equivalent block of coefficients. The two-dimensional transform is given by the relation

$$[C] = [T][X][T]^{\mathrm{T}} \tag{10}$$

where $[T]^{\mathrm{T}}$ is the transpose of $[T]$.

It should be noted that, before the substitution of zeroes for negligibly small coefficients (or any other rounding or approximation of values) the transform is exactly invertible and no data compression has occurred. Compression comes from the strong nonuniformity imparted to the coefficient set by the transform and which is not present in the original data. As noted earlier this is a consequence of the strongly lowpass nature of much image data. As an extreme example of this consider a completely uniform region of image data. Although all picture values are the same, in the transform domain only the lowest-order coefficient (called the dc coefficient) is finite and all others are identically zero! In our 8×8 example we need, therefore, retain only 1 out of 64 coefficients to be able to recover our data.

Over the years transform coding has developed into a powerful means of compressing still pictures down to, say, 0.25 to 0.5 bits per element with acceptable quality. Many schemes for processing the coefficients have been devised, usually incorporating large degrees of adaptability—splitting the blocks to be coded into various categories, from those with highly visible active detail to those that are substantially uniform, each having its own optimally designed minimum mean square error quantization strategy (15). Nowadays, however, the preferred scheme is to threshold the coefficient set and then apply a uniform quantizer followed by variable word-length coding of the kind discussed earlier (16). The thresholding is conveniently carried out by having a quantizer with constant stepsize save for a dead-zone of greater extent around zero. Coefficients are processed in a run length/level format that requires explanation. As we have seen, many data blocks will produce only a few significant coefficients, which will be situated within the low-order region of the coefficient block (matrix). Given the conventional row/column structure of the transform and data matrices this region will lie in the upper left part of the coefficient matrix. Some blocks, however, will contain image structure, which results in significant coefficient values in other parts of the array. On the assumption that the low-order terms will, nevertheless, exercise a dominating influence a zig-zag scan path is defined, moving gradually away from the top left and towards the bottom right of the coefficient array (where, on average, the smallest coefficients may be expected to reside). Coefficients are then identified on the basis of a run-length/level pair, where the former indicates the distance, along the scan path, of the present nonzero coefficient from the last such.

Two-dimensional transform coding is the basis of the Joint Photographic Experts Group (JPEG) still picture standard (17). As far as video coding is concerned, there is the option of extending the technique (just as in the case of the prediction algorithm) into three dimensions, and this has indeed been investigated (18). There is the problem of delay, however; for an $8 \times 8 \times 8$ transform we must wait eight frames before the transform along the time axis can begin and this, combined with the problems of including motion compensation in the transform domain, has meant that other approaches have been sought. For all standards at present in existence for video compression, two-dimensional transform coding has been combined with the predictive technique described earlier into a scheme called hybrid coding—specifically, applying motion-compensated interframe prediction to 8×8 image blocks and then two-dimensionally transforming the blocks of error signals so produced.

Hybrid Coding

This technique combines the extreme simplicity of the prediction algorithm (discounting, for the moment, complexities of implementation caused by the introduction of motion compensation) with the power of the transform approach in reducing intraframe correlation between elements. Thus, it can produce significant levels of video compression (19). It is worth noting at this point that, in the absence of quantization, both predictive and transform coding algorithms are totally invertible (within the accuracy of the word lengths used)—the error signal added to the prediction reconstitutes the input value in the former; an inverse transform regenerates the input data stream in the latter. Compression results in both cases from the sweeping change in the distribution of amplitude values, together with their appropriate quantization (it has been said more than once that the secret of all data compression lies not so much in the actual technique used but in the design of the following quantizer). Furthermore, it does not matter whether we implement the prediction operation first followed by the transform or vice versa. One method of carrying out hybrid coding is, therefore, to transform individual video frames (producing equivalent frames of transform coefficients) and then employ a predictive algorithm operation between coefficient frames to produce the final coded output. A problem then arises with respect to the implementation of motion compensation. There is no question that it is desirable, in the search for improved efficiency, to attempt to account for object motion in the coding process. Neither is there any doubt than this can be done using frequency (or transform) domain information, as would be necessary in this case. At the time that standards for video compression were being developed (20), the one technique that had been exhaustively researched and was guaranteed to work (albeit approximately) was that of full search block matching (described in the preceding). Less optimal reduced search techniques were also available. As this algorithm works in the data (space) domain it has become conventional to implement hybrid coding by carrying out motion compensated prediction to generate blocks of error terms, which are then two-dimensionally intraframe transform coded to generate the output signal.

It is this approach, as illustrated in Fig. 2, that has been built into almost every video compression standard for coding, at rates between tens of kilobits per second and those appropriate for high definition television; there are many sources of related software and hardware. There is a theoretical, if not a practical, problem with the technique, however, and that is—why does it work at all? Prediction theory tells us that the aim should be to generate as good a prediction as possible in order that the error signal be not only small but also uncorrelated. In such a case the error block looks just like random noise and its transformation is then of no further benefit in terms of data compression. It seems likely that there are two reasons for the success of hybrid transform coding. First, block matching is at best an approximate technique for the compensation of object motion through a video sequence. While some motion compensated error blocks, when examined, do indeed appear noise-like, many still contain structured image detail resulting from the approximations involved. Secondly, intraframe transform coding is a very powerful technique for the reduction of interelement correla-

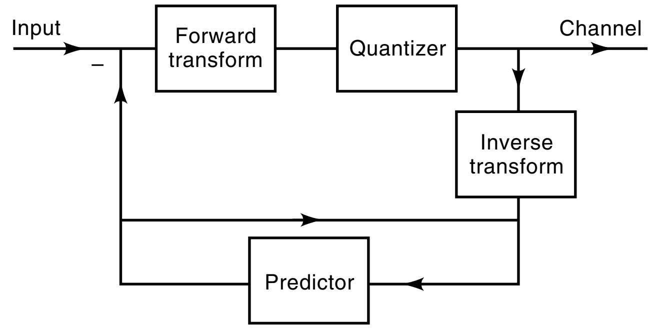

Figure 2. Note the similarity to Fig. 1, with the inclusion of the forward transform to produce sets of coefficients for quantization and transmission, and the inverse transform to regenerate a spatial signal for the predictor.

tion and so can still work effectively on those blocks where motion compensation has (partially) failed. Two other factors enter the picture in a practical context. As time goes by, we shall naturally expect improvement in our capability to perform object tracking through a video sequence. This will make the error block sequence overall more noise-like and thus further reduce the efficiency of the transform. It is likely, though, that the cost of this advance will be a need to spend more channel capacity on sending motion parameters to the decoder and the result will be an even more complex and subtle trade-off between the transmission of motion information and active picture detail. The other factor relates to the need to update decoded information at all, given that large areas of even quite detailed video sequences remain unchanged over a span of many frames. In such cases where all error signals within a (motion compensated) block are small, we may simply flag the decoder to reproduce the same block in the present frame, and so not involve the transform operation at all. Such conditional replenishment is a very useful way of obviating the transmission of unnecessary information.

OTHER TECHNIQUES FOR VIDEO DATA COMPRESSION

Since its first application to still pictures in the 1960s, transform coding has constituted the mainstream of both still and moving image compression activity. This is far from saying that it had no competitors, however, or indeed critics, who remarked upon the technique's relative complexity (both forward and inverse transforms need to be implemented using a suitable fast algorithm), the large dynamic range of the coefficients generated (21) and thus the wordlengths needed, etc. Many other techniques were therefore developed, all with distinct advantages and disadvantages. Some of the more significant are described next.

Vector Quantization

Arguably the technique having the highest profile after transform coding is vector quantization (22). This is a space domain approach, first applied to still pictures in the early 1980s. It is the vector analog of scalar (one-dimensional) quantization—instead of comparing data element-by-element with decision regions on a line and selecting the appropriate reproduction word from a quantization table this is done on a vector basis. Typically, image data is divided into 4×4 blocks and the elements reordered, for convenience, into a vector of length 16. From a training sequence consisting of similar blocks (and so similarly reordered vectors) from many somehow typical images, a representative codebook is derived containing, say, only 512 of the enormous number of possible combinations of element values within a 4×4 block. Using some sort of distance measure (perhaps mean square or mean absolute error) the closest of these to the vector being coded is chosen and the corresponding nine bit ($\log_2 512$) index (label) transmitted. The decoder also possesses the codebook and simply replicates, on the screen, the corresponding entry. We thus have a simple scheme (the decoder is no more than a look-up table, with perhaps a trivial scaling operation or two) which, in this example, sends nine bits to represent a 4×4 block, giving an equivalent rate of 9/16, that is, about 0.5 bit/element. As it stands, gross errors will be present in the reproduced image due to the small number of reproduction vectors available (corresponding to very coarse quantization in the one-dimensional, scalar, equivalent) and coding will take a significant time as a consequence of the need to generate the codebook in the first place and search it for the closest entry. Most research into the application of vector quantization as an image compression technique has been directed towards solving these two basic problems, which are naturally intensified where video compression is concerned.

One way in which a codebook may be generated is as follows (23). We first determine the multidimensional centroid (center of gravity) of the totality of training sequence vectors and then add a small perturbation to this centroid vector to create a further, nearby, vector. All training vectors are then allocated to the nearest one of these two initial approximations and the centroids of the two resulting distributions determined. Iteration of this operation produces an optimum two reproduction vector codebook. Subsequent splitting of the two vectors into four (then 8, 16, etc.) allows the generation of a codebook of the required size. There is a multitude of technique available for the optimization of this procedure according to particular circumstances, as indeed there is for obviating the necessity of fully searching the codebook in order to find the nearest reproduction vector, and the technique has been refined to the point where its performance rivals that of transform coding for still pictures at low rates. Two obvious ways to apply vector quantization to video sequence compression are by extending the two-dimensional block into three dimensions and by processing separate frames singly in an intraframe manner. Schemes for adapting the codebook entries to keep track of variations in image detail in a sequence can be advantageous and have also been reported. The most common application of vector quantization for video signal processing, however, has been its use to process either the motion compensated/predicted frame difference (i.e., as a direct substitute for the transform) (24) or sometimes as a postprocessing step to code the transform coefficient arrarys in an otherwise conventional hybrid coder. Generally, speaking, although vector quantization has the advantages that it does not involve the complexity inherent in forward and inverse transformation, and it concentrates the need for processing power at the coder (allowing a trivially simple decoder), it is unlikely that it will supersede the hybrid transform approach in video compression applications.

Subband and Wavelet Coding

It is always open to us to process/analyze signals in either the (original) data domain or the frequency domain and developments in image data compression have, at various times, emphasized one or the other. Thus vector quantization is an example of the first while transform coding and the techniques to be described in this section utilize frequency, or frequency-like, transformations. Subband coding (25) is the culmination of an idea with a long history in image coding—that of splitting the signal into different frequency bands and then processing each subband according to its own individual statistical properties. Where video signals are concerned we have already seen that the frequency response is highly nonuniform and we may expect to see a large (and thus relatively important) signal at low frequencies and a smaller signal at high frequencies. A very simple (one-dimensional) scheme splits the frequency spectrum into two components, each occupying half of the original bandwidth and each of which can be subsampled by a factor of two, the total number of samples remaining constant (in the general case, the factor two may be replaced by K). After efficient processing of the subsampled filter outputs they are transmitted to the decoder, interpolated by inserting a zero between the sample locations (in the general case $K - 1$ zeros) and then filtering, and finally combined to obtain the reconstructed data. This is illustrated in Fig. 3. As usual, with images it is normal to process both dimensions in the same way, and filter design and its extension into two-dimensions has been extensively reported in the literature (26). Furthermore, it is not necessary for all bands to have the same frequency extent and some benefit may be obtained by making the lower band(s) relatively smaller as compared with the higher frequency ones, consistent with the reduced acuity of the eye at higher frequencies. As with transform coding, apart from filter imperfections the filtering/subsampling/interpolation operations have no direct part to play in compression, this resulting from whatever processing is applied to the subsampled values (we may, indeed, look on transform coding as a kind of transform processing in which each subband has only one coefficient). Typically, then, subband coefficients may be processed using predictive coding or vector quantization, or the lowest band (simply a lowpass filtered version of the original image) transform coded.

Again, subband coding is a technique that can be applied for the compression of video signals. Once more, three-dimensional schemes are possible but not popular, and other schemes have been proposed that decompose individual frames into their constituent subbands followed by predictive coding or vector quantization. Subband coding may also be employed to process the error signal in a conventional hybrid structure, in place of the transform operation, once more with predictive coding (27) or vector quantization (28) following. One advantage of subband schemes is that they process the whole of the image or error signal field and thus do not suffer from the appearance of edge degradation effects in the reconstructed image characteristic of block-based transform or vector quantization schemes (note, however, that this is not necessarily true if block-based motion compensation is incorporated into the algorithm).

Conventional Fourier analysis employs an integral formulation with limits at plus and minus infinity; this raises important questions in connection with the interpretation of any results so obtained, which are especially important where rapidly changing (nonstationary) signals are concerned. In brief, this implies that such an analysis averages out any changes taking place within the analysis window, and so cannot be used to localize significant detail. This has led to an interest over the past decade or so in so-called wavelet analysis (29). Here a trade-off between frequency and time (in image terms, space) is available, which parallels the logarithmic form of human perception (i.e., of a constant fraction of some stimulus magnitude, rather than of a constant absolute value). Thus, varying levels of resolution may be achieved—wide bandwidth at high frequencies giving good spatial resolution for fine detail, long analysis windows at low frequencies giving good frequency resolution. As far as image compression is concerned, such processing can be carried out in a way superficially similar to subband coding. We start by carrying out a simple 2 : 1 subband decomposition horizontally and vertically, resulting in four frequency bands L_h L_v; H_h L_v; L_h H_v; H_h H_v, each 1/4 the extent of the original image, where L and H represent low and high frequencies respectively, and subscripts h and v refer to horizontal and vertical terms. All signals but L_h L_v (the lowest resolution one) are so-called detail signals, and will play, if retained, a part in the subsequent image reconstruction process. L_h L_v is now decomposed in the same way but at one level of resolution lower, three more detail signals and another low-pass version, now all 1/16 of the area of the original appearing. This may be continued to lower levels of resolution if desired. The lowest lowpass image may now be sent to the decoder needing, relatively, only very few bits. The detail signals all have small energies and a highly peaked amplitude distribution and may be efficiently coded using prediction or vector quantization. For reconstruction, lower level images are interpolated horizontally and vertically, the decoded detail signals from the next level up are added, and so on.

Wavelet decomposition can be applied to video compression in a manner similar to that used with subband coding. We may carry out a motion compensated prediction first and apply wavelet decomposition to the error signal, followed perhaps by vector quantization of the wavelet terms. Alternatively, we can perform the wavelet analysis first on the actual incoming frames and then use motion compensated prediction on the wavelet image fields followed by vector quantization. This has the advantage that multiresolution (hierarchical) motion compensation (as referred to earlier) can be built into the algorithm (30).

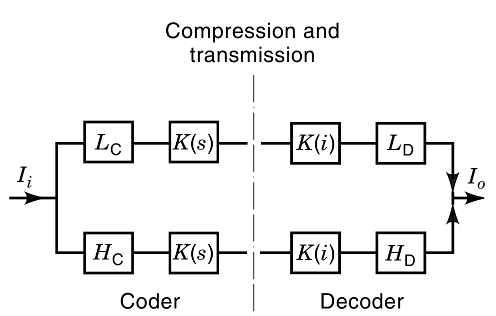

Figure 3. Low- and high-pass filters L_C and H_C split signal I_i into, respectively, low and high frequency components. These are then subsampled by factor K (block $K(s)$), coded and transmitted. At the decoder they are interpolated by factor K (block $K(i)$) in conjunction with low- and high-pass filters L_D and H_D and added to generate the output signal I_o.

Multiresolution

Because video material is available today in a wide variety of formats and resolution levels, an important design criterion for any image communication system is its ability to code/decode over a range of such parameters. For example, it could be useful to display television material at reduced resolution over a videoconference network, or similarly to generate a reduced area picture insert. In this respect some algorithms perform better than others. One of the drawbacks, indeed, of transform coding is the lack of flexibility offered by the coefficient set for this purpose. Thus if a crude, low resolution image is all that is required we might just decode a few coefficients (in the limit only the dc term), or, maybe, use rounded approximations to the original values. To obtain higher resolution more often, or more accurately determined, values can be used. In this respect it is preferable to use a properly hierarchical scheme such as wavelet coding described above, where a series of different levels is available from the start, together with additional detail signals to enhance resolution wherever necessary.

To conclude this section, we may note that a multiplicity of schemes has been developed for video compression over the past few decades, details of many of which it has not been possible to include here (31). Furthermore, considerable ingenuity has been exercised in an effort to combine almost all algorithms in almost every way possible, in an attempt to achieve better and better picture quality at ever lower transmission rates. Out of all of this research motion compensated prediction, transform coding, and variable length coding have emerged; vector quantization and subband/wavelet techniques are close runners-up. In spite of the establishment of standards, however, research has not abated; indeed, some of the more interesting and fundamental problems still remain to be solved. More advanced and conjectural approaches form the subject of our final section.

ADVANCED TECHNIQUES AND IMAGE TRANSMISSION

By the mid 1980s concern was beginning to be expressed that the algorithms that had by then been developed were running out of steam and that their continual refinement was not, in fact, producing worthwhile gains in image/video compression (32). There was, furthermore, a feeling that perhaps the topic as a whole was proceeding along the wrong lines. What were the underlying reasons for this disquiet? We have already seen why predictive coding cannot, in its basic form, produce extreme values of compression, but what about transform coding? Undoubtedly one of the factors contributing to the extensive development of this technique was the introduction of the fast Fourier transform (FFT) in the mid 1960s, enabling large transforms to be carried out in fractions of the time previously required. Initially implemented by using the FFT, the DCT soon had efficient fast variants of its own. Again, this approach was underpinned by the fact that Eq. (9) is a matrix manipulation, and transform coding thus has the massive foundation of matrix mathematics at its disposal to provide new concepts and insights for its development.

On the other side of the argument is the fact that the transform operation is applied to, typically, 8×8 image blocks taking no note of where, with respect to image detail, block edge transitions may lie. These blocks are then processed independently and, when attempts are made to achieve the lowest possible data rates, allocation of insufficient bits to deal with block to block luminance profiles can easily result in a very visible, and annoying, repetitive block structure being overlaid on the reconstructed image. This form of degradation is characteristic of all block-based schemes—even those which may not use a block coding algorithm as such but only depend on block-based motion compensation, for example, to improve overall performance; it is, however, most troublesome with algorithms such as transform coding and vector quantization. Again, when restricting bit allocations to try to improve compression performance, it is natural to delete more and more high frequency coefficients, and the resulting reconstruction looks out of focus and blurred as a result.

Might it be that other schemes would be subject to different sources of degradation at low data rates, which could be more acceptable to the viewer? Alongside these more specific, technical, considerations is the question—what, in any case, are we coding? All those algorithms such as predictive and transform coding depended upon the correlation properties of the waveforms, which the (maybe multidimensional) source presented to the coder. These could originate from anything—video sources, audio sources, instrumentation transducers, etc.; all that the coder needed to operate was a signal amplitude distribution. It began to be felt more and more strongly that, in the case of image or video compression, scenes processed were of actual *objects*—humans, buildings, cars, trees etc., and thus we should be coding these and not simply interelement variations or 8×8 blocks.

At the same time, the development of compression algorithms, which had predominantly an electrical engineering flavor (associated with signal statistical and prediction theory and Fourier transformation) began to be influenced by techniques of a more computer science-based nature; this has continued to the present day. One such was the application of the 'quadtree' structure (33) to compression algorithms. Although still a block-based technique, it offered a simple, highly adaptive, and efficient method for the processing of the motion compensated difference signal. The 8×8 or 16×16 blocks of error differences are divided into smaller (4×4, 2×2, etc.) blocks, at each step being tested for some coherence property, perhaps amplitude uniformity. If the test is satisfied, subdivision ceases. Alternatively the algorithm can start at the lowest level, successively merging larger and larger blocks, maybe on the basis of similarity of mean values (34). Another was the use of neural techniques to aid the speedy design of codebooks for vector quantization and also to help in optimizing predictive coding algorithms to take advantage of both linear and nonlinear correlation in the data being processed. A more recent technique that has achieved a level of notoriety rarely encountered in technical areas is that of fractal coding (35). Once heralded as the answer to all compression problems (apparently allowing image reproduction at vanishingly small transmission rates), it is now, after much intensive development, producing results of similar quality to those of transform and vector quantization approaches. This is done by taking account of the self-similarity property of image data, either between regions suitably translated, scaled, and rotated within the same image field or between corresponding fields with different resolutions in the sort of hierarchical structure mentioned previously to define an affine transform whose coefficients form the basis for the reconstruction of the image at the receiver. It should be noted, however, that the technique has nothing to do with notions of fractional dimension curves inherent in the more mathematical interpretation of the term and it is more usefully considered as a variety of vector quantization. It also has the disadvantage of being a block scheme. Of more importance to future developments in image compression are two techniques at present under active investigation—segmentation and model (object) based coding.

Segmentation (36) has a long history in image processing. From the earliest days a body of knowledge has steadily built up relating ways of separating object detail from other items in a scene, or from the surrounding background. These can be based upon two quite different principles—of difference or of similarity. In the first case, sharp luminance, color, etc., discontinuities signal the change from object to object (or background), and conventional edge detectors can be used to detect and delineate significant objects in scenes. In the second case, region growing techniques can be used to define areas, all elements within which have (within a prespecified tolerance) the same property. In both cases the object is to define regions with closed contours that may subsequently be coded at low data rates. Following segmentation, regions can be coded using polynomial fits of various orders, together with some sort of (differential) chain code (37) applied to the object boundaries. Defects are present in the reconstructed picture and take the form of lack of realistic luminance/color profiling and shading. An important benefit, however, is that the edge detail remains sharp even at very low rates, and this aids object definition (in contrast to transform coding). For video compression, the segmentation process is carried out on the motion compensated frame difference signal (38). Two further refinements improve the efficiency of the algorithm. First, adaptive (variable threshold) segmentation can be used to enhance detail in important areas of the picture. Second, there is an intimate connection between segmentation and motion compensation—objects usually move without too much internal deformity from frame to frame and thus a region of motion vectors all with approximately the same amplitude and orientation will usually indicate a coherent object. Although our competence in this area is gradually increasing, there is no way yet in which any computational algorithm can carry out the segmentation operation with even a tiny fraction of the competence of the human eye in such a situation, let alone define more or less significant objects on which to concentrate coding capacity. Nevertheless, in some situations of importance in low rate transmission of images, especially those of head and shoulder views presented against a static background (as used in videophone interchange), a reasonable result can be obtained. Sophisticated segmentation techniques allowing true object tracking and coding through a video sequence will undoubtedly play a more and more significant role as video compression develops.

Earlier, a technique known as conditional replenishment was mentioned, in which only changing luminance detail was updated from frame to frame at the decoder/receiver. In a much more all-embracing way, it is possible to envisage such a scheme operating upon the characteristics responsible for higher-level object properties; this is the mechanism of operation of model, or object based, coding (39). In the basic scheme, interest has concentrated upon head and shoulders images as employed in interpersonal (videophone) communications to produce systems that can operate at a few kilobits per second to produce realistic representation of the changes in facial expressions and head attitudes characteristic of face-to-face interchange. The method operates by establishing a three-dimensional model of the head (usually a polygonal wire frame with several hundred facets) together with one set of parameters that characterize the basic structure of an individual face (eye location, nose length, etc.) together with another that represents expression—eye direction, or mouth corner location, for example. At the coder feature identification techniques are employed to localize not only the basic expressive properties of the face and head but also the way in which these move during the video sequence (analysis). Signals representing the changes are transmitted to the decoder and are then used to move the vertices of the polygon model (synthesis), the overall representation being made realistic by the use of some form of computer graphics shading. In this way surprisingly lifelike moving head reconstructions may be generated at extremely low data rates. Obviously the scheme as it stands is very object-specific, and research has been carried out into ways of overcoming this disadvantage (40). In this case a model world is generated consisting of objects having parameter sets defining motion, shape, and color derived from the actual input scene. Checks are made during processing to ensure that the model reflects the real world to a sufficiently accurate degree; the scheme can be made to work well, but as yet still must be used in association with more conventional techniques that can deal with those parts of the scene the model fails to represent closely enough. It is undoubtedly the case, however, that a move toward the use of more advanced segmentation/computer graphics/modelling techniques for the representation of actual objects will eventually lead to much more flexible techniques for the efficient transmission or storage of image and video material. In this connection we might mention that the brain gains significant information about the surrounding world by the use of stereoscopic vision. Curiously, only recently has a start been made on using this effect in object tracking for efficient coding. It is likely that further investigations in this direction will yield useful improvements in compression strategies.

In conclusion, a word may be said about the actual mechanism of image transmission. Traditionally, the output of the coder has always been sent over a fixed rate channel. Because any efficient coder inevitably has an adaptive structure, the rate at which bits are generated will be very variable—high when coding regions of significant image activity and low when processing uniform background areas. Buffering is thus necessary between the coder output and the channel, together with some means of preventing overflow, usually by arranging for a signal dependent upon the degree of fullness of the buffer to be fed back to increase the coarseness of quantization or to initiate the process of data subsampling and so reduce the rate at which coded bits are generated (41). In addition, means must be employed to prevent the incidence of transmission errors from corrupting the reconstructed picture. In fixed systems this will be carried out by the addition of error control codes to the channel data stream. Where the error rate is likely to be much higher and possibly very variable (in wireless systems, for example) more powerful error control is needed, especially for the more important parts of the transmission. Thus, in a wavelet or subband scheme, for example, motion vectors and low frequency coefficients can be highly protected while the higher frequency (detail) bands are not so essential and can be protected to a lesser degree (42). Again, many schemes for trading-off available coding capacity between error protection and actual coding of video detail have been proposed (43). Nowadays there is much interest in the use of the asynchronous transfer mode (ATM) for digital transmission (44). Where images are concerned, this changes dramatically the service quality/bit-rate relation. With fixed rate working, when very detailed material is coded, coarse quantization must be invoked and the quality of the output drops. With variable rate transmission, if the system can accept the demand at such periods for increased capacity, service quality will remain substantially unchanged and, of course, the packet hierarchy can be arranged to guarantee delivery of important overhead information, coefficients, etc. (45). With efficient multiplexing, then, the latter mode of communication offers the possibility of more reliable quality of service overall.

BIBLIOGRAPHY

1. *Encoding Parameters of Digital Television for Studios.* CCIR Recommendation 601, **XI**: 95–104. *Interfaces for Digital Component Video Signals in 525 and 625-line Television Systems.* CCIR Recommendation 656, **XI**: 105–117, 1990.
2. D. A. Huffman, A method for the construction of minimum redundancy codes, *IRE Proc.,* **40**: 1098–1101, 1952.
3. I. H. Witten, R. M. Neal, and J. G. Cleary, Arithmetic coding for data compression, *Commun. ACM,* **30**: 520–540, 1987.
4. N. S. Jayant and P. Noll, *Digital Coding of Waveforms,* Englewood Cliffs, NJ: Prentice-Hall, 1984.
5. R. J. Clarke, *Transform Coding of Images,* San Diego: Academic Press, 1985.
6. L. H. Zetterberg, S. Ericsson, and H. Brusewitz, Interframe DPCM with adaptive quantization and entropy coding. *IEEE Trans. Commun.,* **COM-30**: 1888–1899, 1982.
7. L. H. Zetterberg, S. Ericsson, and C. Coturier, DPCM picture coding with two-dimensional control of adaptive quantization, *IEEE Trans. Commun.,* **COM-32**: 457–461, 1984.
8. C. R. Moloney and E. Dubois, Estimation of motion fields from image sequences with illumination variation, *ICASSP Proc.,* 2425–2428, 1991.
9. R. W. Young and N. G. Kingsbury, Frequency domain motion estimation using a complex lapped transform, *IEEE Trans. Image Process.,* **IP-2**: 2–17, 1993.
10. J. R. Jain and A. K. Jain, Displacement measurement and its application to interframe image coding, *IEEE Trans. Commun.,* **COM-29**: 1799–1808, 1981.
11. Q. Wang and R. J. Clarke, Motion compensated sequence coding using image pyramids, *Electron. Lett.,* **26**: 575–576, 1990.
12. H. G. Musmann, P. Pirsch, and H-J. Grallert, Advances in picture coding, *Proc. IEEE,* **73**: 523–548, 1985.
13. N. Ahmed, T. Natarajan, and K. R. Rao, Discrete cosine transform, *IEEE Trans Comput.,* **C-23**: 90–93, 1974
14. K. R. Rao and P. Yip, *Discrete Cosine Transform. Algorithms, Advantages, Applications,* San Diego: Academic Press, 1990.
15. W. H. Chen and C. H. Smith, Adaptive coding of monochrome and color images, *IEEE Trans. Commun.,* **COM-25**: 1285–1292, 1997.
16. W. H. Chen and W. K. Pratt, Scene adaptive coder, *IEEE Trans. Commun.,* **COM-32**: 225–232, 1984.
17. Joint Photographic Experts Group ISO/IEC JTC1/SC2/WG8 CCITT SGVIII. JPEG Technical Specification Revision 8, 1990.
18. T. Akiyama, T. Takahashi, and K. Takahashi, Adaptive three-dimensional transform coding for moving pictures, *Proc. Picture Coding Symp.,* Cambridge, MA, 26–28 March 1990.
19. S. Ericsson, Fixed and adaptive predictors for hybrid predictive/transform coding, *IEEE Trans. Commun.,* **COM-33**: 1291–1302, 1985.
20. K. R. Rao and J. J. Hwang, *Techniques & Standards for Image, Video and Audio Coding,* Englewood Cliffs, NJ: Prentice-Hall, 1996.
21. R. J. Clarke, On the dynamic range of coefficients generated in the transform processing of digitised data, *IEE Proc., Part F: Commun. Radar Signal Process,* **132**: 107–110, 1985.
22. R. M. Gray, Vector quantization, *IEEE Acoust. Speech Signal Process. Mag.,* 4–29, April 1984.
23. Y. Linde, A. Buzo, and R. M. Gray, An algorithm for vector quantizer design, *IEEE Trans. Commun.,* **COM-28**: 84–95, 1980.
24. Q. Wang and R. J. Clarke, A new motion-compensated image sequence coding scheme at 64 kb/s, *IEE Proc. Part I, Commun. Speech, Vision,* **139**: 219–223, 1992.
25. J. W. Woods and S. D. O'Neil, Sub-band coding of images, *IEEE Trans. Acoust. Speech, Signal Process.,* **ASSP-34**: 1278–1288, 1986.
26. M. Vetterli, Multi-dimensional sub-band coding, some theory and algorithms, *Signal Process.,* **6**: 97–112, 1984.
27. H. Gharavi, Differential sub-band coding of video signals, *ICASSP Proc.,* 1819–1822, 1989.
28. M. H. Ahmad Fadzil and T. J. Dennis, Sample selection in sub-band vector quantization, *ICASSP Proc.,* 2085–2088, 1990.
29. C. K. Chui, *Wavelets: A Tutorial in Theory and Applications,* San Diego: Academic Press, 1992.
30. S. Yao and R. J. Clarke, Motion-compensated wavelet coding of colour images using adaptive vector quantisation, *Proc. Conf. Image Process.: Theory Appl.,* San Remo, Italy, June 1993, pp. 99–102.
31. R. J. Clarke, *Digital Compression of Still Images & Video,* San Diego: Academic Press, 1995.
32. M. Kunt, A. Ikonomopoulos, and M. Kocher, Second generation image coding techniques, *Proc. IEEE,* **73**: 549–574, 1985.
33. H. Samet, The quadtree and related hierarchical structures, *Comput. Surveys,* **16**: 187–260, 1984.
34. P. Strobach, Tree-structured scene adaptive coder, *IEEE Trans. Commun.,* **COM-38**: 477–486, 1990.
35. A. E. Jacquin, Image coding based on a fractal theory of iterated contractive image transformation, *IEEE Trans. Image Process.,* **IP-1**: 11–30, 1992.
36. R. M. Haralick and L. G. Shapiro, Survey: Image segmentation techniques, *Comp. Vis. Graph. Image Process.,* **29**: 100–132, 1985.
37. C. C. Lu and J. G. Dunham, Highly efficient coding scheme for contour lines based on chain code representations, *IEEE Trans. Commun.,* **COM-39**: 1511–1514, 1991.
38. M. Soryani and R. J. Clarke, Segmented coding of digital image sequences, *IEE Proc., Part I, Commun., Speech, Vision,* **139**: 212–218, 1992.
39. H. Li, A. Lundmark, and R. Forchheimer, Image sequence coding at very low bitrates, A Review, *IEEE Trans. Image Process.,* **IP-3**: 589–609, 1994.
40. H. G. Musmann, M. Hotter, and J. Ostermann, Object-oriented analysis-synthesis of moving images, *Signal Process.: Image Commun.,* **1**: 117–138, 1989.
41. J. Zdepski, D. Raychaudhuri, and K. Joseph, Statistically based buffer control policies for constant rate transmission of compressed digital video, *IEEE Trans. Commun.,* **COM-39**: 947–957, 1991.
42. R. Stedman et al., Transmission of sub-band coded images via mobile channels, *IEEE Trans. Circuits Syst. Video Technol.,* **3**: 15–26, 1993.
43. M. G. Perkins and T. Lookabaugh, Combined source-channel DCT image coding for the Gaussian channel, *EUSIPCO Proc.,* **1360**: 865–868, 1993.

44. J. Lane, ATM knits voice, data on any net, *IEEE Spectrum,* **31**: 42–45, February 1994.

45. M. Ghanbari and V. Seferidis, Cell-loss concealment in ATM video codecs, *IEEE Trans. Circuits Syst. Video Technol.,* **3**: 238–247, 1993.

ROGER J. CLARKE
Heriot-Watt University