IOSR Journal of Engineering (IOSRJEN)
ISSN: 2250-3021 Volume 2, Issue 6 (June 2012), PP 1357-1373
www.iosrjen.org

IMAGE COMPRESSION ALGORITHM

M.S.Tamilselvi[1] and A. Rajiv Kannan[2]

***1.** Research Scholar, Dept of Computer Science&Engg., ANNA University, Coimbatore, India*
***2.** HOD., Dept of Computer Science&Engg., K.S.R. College of Engineering, Thiruchengode, India*

Abstract: *Existing Video Indexing Models are analyzed and a practical approach to the optimal Video Indexing is introduced. It is studied under all the phases of video indexing processes like segmentation, indexing, database storage, query based access, browsing and video clip retrieval, etc… The main aim is to easily parse the video stream into meaningful scenes, maintain them in an effective database with minimal data repetitions, efficient query handling and user friendly browsing capabilities. Conceptual Graph, Motion Estimation, Mean Absolute Frame Difference, Displaced Frame Difference, Dublin Core and other important existing techniques are utilized in this model. The main aim is to reduce the memory storage of video clippings without visible loss in quality by using a predictive video compression technique. Today almost all video clippings face a compromise between their quality and memory size. Even Video clippings are not advised to include in the web pages because of their downloading time and its memory size. To try to rectify it and to take positive measures to convert a video clipping file similar to *.swf (flash player's shockwave files) is the aim of this presentation.*

Keywords: *Frame Segmentation; Search Threshold; Block Matching; MotionVectorCorrection; Vector Coding; Prediction Error Coding*

1. INTRODUCTION

Requirements for Any Compression Algorithm

Any compression algorithm should satisfy the following requirements to standardize the compression-uncompress ion procedure:

- It should have a high compression rate at the same time maintain a good enough quality so that uncompressed file is not so much different from the original file and contains almost all of the important information
- The technique of compression should be simple, in fact the least complex technique is preferred
- The delay introduced in compression should be very small
- Based on application requirements, there are two kinds of codec's - symmetric and asymmetric Symmetric algorithms take the same amount of time to compress and uncompress files, while asymmetric codec's spend more time in one of the above processes. Asymmetric codec's are usually preferred, because the general user would like to compress the file once, and then uncompress it whenever needed, so that we need the decompression time to be small while the compression time can be allowed to be big because we are only going to compress the file once. This mode of the codec is called the *retrieval* mode, and it is usually expected to have the following properties:
- FFWD, REW should be possible with simultaneous display of the movie
- Random access with speeds should be less than 0.5 s
- The format should be independent of frame size and frame rate, and should support several rates - this means that users who are trying to stream movie previews, for example, should be able to choose between slow and high bandwidth connections, so that the website can provide to both slow and fast connections
- The audio and video tracks should be in sync. Nowadays there are programs which can fix audio-video sync, and are easily available over the internet.
-

Compression in Digital Video

A great deal of research has gone into image and video compression and indeed it is quite difficult to invent something new in this field. Let us have assumption that the input is always a PCM digitized signal in color components. The output of the compression process is a bit stream. Let's consider each technique briefly:

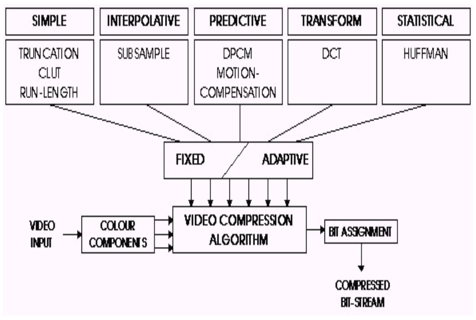


Figure 1.1. Compression in Digital Video

2. NEED FOR COMPRESSION & METHODS

2.1 Compression Techniques

Simple Compression Techniques

Various techniques exist, including:

- Truncation
  Reduces data by reducing the number of bits per pixel. This suffers from contouring (resolution loss) but has the advantage that processing is simple.
- Color Lookup Table (CLUT)
  Pixel values represent an index into a table of colors. The processing for this is non-trivial.
- Run length coding
  Blocks of repeated pixels are replaced with a pixel value and a count. This works well on images with blocks of single colors and can achieve a high compression ratio. However it not effective if images contain no repetitive areas.

Interpolative Techniques

This technique aims to send a subset of the pixels and use interpolation to reconstruct the intervening pixels. This technique is particularly useful for motion sequences, as certain frames are compressed by still compression; the frames between these are compressed by doing an interpolation between the other frames and sending only the data needed to correct the interpolation.

Predictive Techniques

This relies on the fact that there is nearly always some redundancy between frames in a sequence. There are two common methods:

- DPCM (Differential Pulse Code Modulation)
  This operates at the pixel level and sends only the difference between successive pixels. Since there is likely to be very little difference between adjacent pixels we can encode the value into smaller data widths. This technique suffers from slope-overload which causes smearing at high contrast edges in an image.
- ADPCM (Adaptive DPCM)
  This tries to reduce the slope-overload by using smaller steps for the difference values.

Transform Coding Techniques

A transform is a process that converts data into an alternate form which is more convenient for some particular purpose. Transforms are ordinarily designed to be reversible. Useful transforms typically operate on large blocks of data and perform some complex calculations. In general transform coding becomes more useful with larger blocks. The Discrete Cosine Transform (DCT) is especially important for video compression.

The DCT

The DCT is performed on a block of horizontally and vertically adjacent pixels (typically an 8 by 8 block of pixels). The outputs represent amplitudes of two dimensional spatial frequency components. These are called DCT coefficients. The coefficient for zero spatial frequency is called the DC coefficient and it is the average value of all the pixels in the block. The rest of the coefficients represent progressively higher horizontal and vertical spatial frequencies in the block.

Since adjacent pixel values tend to be similar or vary slowly from one to another, the DCT processing provides opportunity for compression by forcing most of the energy into lower spatial frequency components.

In most cases, many of the higher frequency coefficients will have zero or new-zero values and therefore can be ignored.

The decoder performs the reverse process, but due to the transcendental nature of the DCT the reverse process can only be approximated and hence some loss takes place. The trick is to use some cunning methods of keeping coefficients so that the loss is minimally visible.

Statistical Coding (or Entropy Coding)

This takes advantage of the statistical distribution of the pixel values. Some data values can occur more frequently then others and therefore we can set up a coding technique that uses less bits for these values. One widely used form of this coding is Huffman encoding. This technique has the overhead that syntax has to be pre-defined or sent for the decoder to work.

Motion Compensation

Consider the case of a video sequence where nothing is moving in the scene. Each frame of the video should be exactly the same as the previous one. In a digital system, it should be clear that, we only need to send one frame and a repetition count. Consider now, a dog walking across the same scene. The scene is the same throughout the sequence, but only the dog moves. If we could find a way of only sending the motion of the dog, then we can save a lot of storage space. This is an oversimplified case of motion video, but it reveals two of the most difficult problems in motion compensation:

- ➢ How can we tell if an image is stationery?
- ➢ How do we extract the part of the image that moves?

We can try to answer these questions by some form of comparison between adjacent frames of the sequence. We can assume that the current and previous frames are available for the comparison. The simple comparison technique is too simple and is like a frame-by-frame DPCM. This has a few problems:

- ➢ The pixel compare will rarely produce a zero difference, due to quantization noise in the system (this can be overcome with a threshold).
- ➢ Images are rarely stationery.

Therefore, more sophisticated techniques are needed. This problem is usually addressed by dividing the image into blocks. Each block is examined for motion.

If a block is found to contain no motion, a code is sent to the decompressed to leave the block the same as the previous one. If enough processing power is available, still more powerful techniques may be applied. For examples, blocks may be compared to previous block to see if there is a difference between the two. Only this difference (*motion vector*) is sent.

3. RECENT WORKS

General Purpose Compression Techniques

There are many popular general purpose lossless compression techniques that can be applied to any type of data.

Run Length Encoding

Run Length Encoding is a compression technique that replaces consecutive occurrences of a symbol with the symbol followed by the number of times it is repeated. For example, the string 111110000003355 could be represented by 15063252. Clearly this compression technique is most useful where symbols appear in long runs, and thus can sometimes be useful for images that have areas where the pixels all have the same value, cartoons for example.

Relative Encoding

Relative encoding is a transmission technique that attempts to improve efficiency by transmitting the difference between each value and its predecessor, in place of the value itself. Thus the values 15106433003 would be transmitted as 1+4-4-1+6-2-1+0-3+0+3. In effect the transmitter is predicting that each value is the same as its predecessor and the data transmitted is the difference between the predicted and actual values. Differential Pulse Code Modulation (DPCM) is an example of relative encoding.

The signal above can have one of 7 possible values (-3 to +3) and so would require 3 bits per sample. Each sample can also be described by the difference between it and the previous sample. Each sample is the same, one more, or one less than the previous sample. Only two bits are required to express the relationship between the samples. Coding the signal in this way results in a reduction of one third in the number of bits.

Huffman Coding

Huffman coding is a popular compression technique that assigns variable length codes (VLC) to symbols, so that the most frequently occurring symbols have the shortest codes. On decompression the symbols are reassigned their original fixed length codes. When used to compress text, for example, variable length codes are used in place of ASCII codes, and the most common characters, usually *space*, *e*, and *t* are assigned the shortest codes. In this way the total number of bits required to transmit the data can be considerably less than the number required if the fixed length representation is used. Huffman coding is particularly effective where the data are dominated by a small number of symbols.

Arithmetic Coding

Although Huffman coding is very efficient, it is only optimal when the symbol probabilities are integral powers of two. Arithmetic coding does not have this restriction and is usually more efficient than the more popular Huffman technique. Although more efficient than Huffman coding, arithmetic coding is more complex.

Lempel-Ziv Coding

Lempel-Ziv compressors use a dictionary of symbol sequences. When an occurrence of the sequence is repeated it is replaced by a reference to its position in the dictionary. There are several variations of this coding technique and they differ primarily in the manner in which they manage the dictionary. The most well known of these techniques is the Lempel-Ziv-Welch variation.

Intraframe Compression Techniques

Intraframe compression is compression applied to still images, such as photographs and diagrams, and exploits the redundancy within the image, known as *spatial redundancy*. Intraframe compression techniques can be applied to individual frames of a video sequence.

Sub-sampling

Sub-sampling is the most basic of all image compression techniques and it reduces the amount of data by throwing some of it away. Sub-sampling reduces the number of bits required to describe an image, but the quality of the sub-sampled image is lower than the quality of the original. Sub-sampling of images usually takes place in one of two ways. In the first, the original image is copied but only a fraction of the pixels from the original are used, as illustrated below. Alternatively, sub-sampling can be implemented by calculating the average pixel value for each group of several pixels, and then substituting this average in the appropriate place in the approximated image. The latter technique is more complex, but generally produces better quality images.

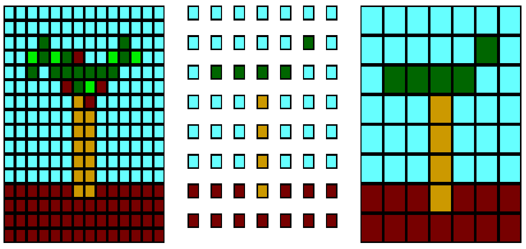


Figure 2.11 -

In this example the pixels in every second row and every second column are ignored. To compensate for this, the size of the remaining pixels is doubled. The same procedure is illustrated below.

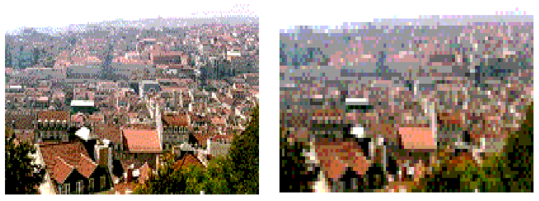


Figure 2.12 -

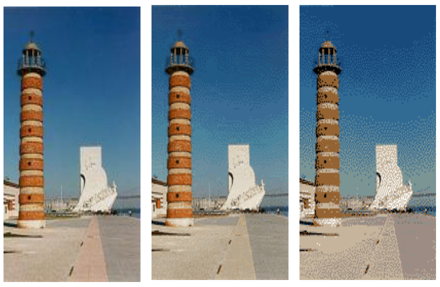
For example, an image might be sub-sampled by 2 in both the x and y directions, thus every second line and every second column of the image is completely ignored. When the sub-sampled image is displayed at the same size as the original image the size of the pixels is doubled. This is known as *pixel doubling*.

When coding color images, it is common to sub-sample the color component of the image by 2 in both directions, while leaving the luminance component intact. This is useful because human vision is much less sensitive to chrominance than it is to luminance, and sub-sampling in this way reduces the number of bits required to specify the chrominance component by three quarters.

Sub-sampling is necessarily lossy, but relies on the ability of human perception to fill in the gaps. The receiver itself can also attempt to fill in the gaps and try to restore the pixels that have been removed during sub-sampling. By comparing adjacent pixels of the sub-sampled image, the value of the missing in-between pixels can be approximated. This process is known as interpolation. Interpolation can be used to make a sub-sampled image appear to have higher resolution than it actually has and is usually more successful than pixel doubling. It can, however, result in edges becoming blurred.

Coarse Quantization

Coarse quantization is similar to sub-sampling in that information is discarded, but the compression is accomplished by reducing the numbers of bits used to describe each pixel, rather than reducing the number of pixels. Each pixel is reassigned an alternative value and the number of alternate values is less than that in the original image. In a monochrome image, Figure 2.4a for example, the number of shades of grey that pixels can have is reduced. Quantization where the number of ranges is small is known as *coarse quantization*.



These three images have different numbers of colors. The first has thousands of different colored pixels. The middle has 64 different colors and the rightmost uses only 8 different colors.

Photographic quality images typically require pixels of 24 bits, but can be reduced to 16 bits with acceptable loss. Images with 8 bits, however, are noticeably inferior to those with 16 bits per pixel. Coarse quantization of images, often called bit depth reduction, is a very common way of reducing the storage requirements of images.

Vector quantization

Vector quantization is a more complex form of quantization that first divides the input data stream into blocks. A pre-defined table contains a set of patterns for blocks and each block is coded using the pattern from the table that is most similar. If the number of quantization levels (i.e. blocks in the table) is very small, as in Figure 2.5, then the compression will be lossy. Because images often contain many repeated sections vector quantization can be quite successful for image compression.

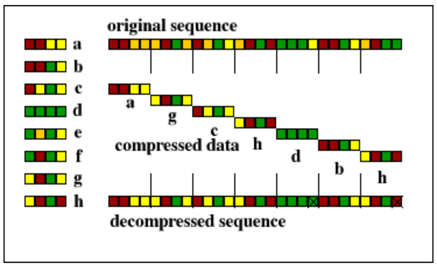


Figure 2.14 - Vector quantization

In this example of vector quantization a sequence of symbols is divided into blocks of four symbols and the blocks are compared to those in a table. Each block is assigned the symbol in the table entry it most resembles. These symbols form the compressed form of the sequence. On decompression an approximation of the original sequence is generated.

Transform Coding

Transform coding is an image conversion process that transforms an image from the spatial domain to the frequency domain. The most popular transform used in image coding is the Discrete Cosine Transform (DCT). Because transformation of large images can be prohibitively complex it is usual to decompose a large image into smaller square blocks and code each block separately.

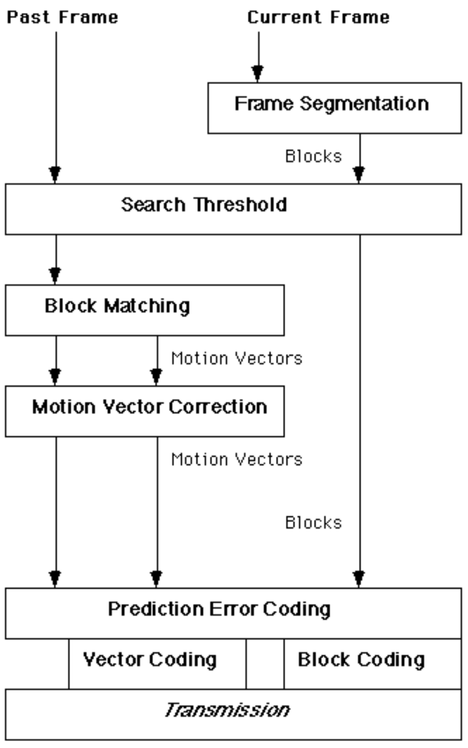
Instead of representing the data as an array of 64 values arranged in an 8x8 grid, the DCT represents it as a varying signal that can be approximated by a collection of 64 cosine functions with appropriate amplitudes. The DCT represents a block as a matrix of coefficients. Although this process does not in itself result in compression, the coefficients, when read in an appropriate order, tend to be good candidates for compression using run length encoding or predictive coding.

The most useful property of DCT coded blocks is that the coefficients can be coarsely quantized without seriously affecting the quality of the image that results from an inverse DCT of the quantized coefficients. It is in this manner that the DCT is most frequently used as an image compression technique.

4. IMPLEMENTATION

4.1 COMPRESSION APPROACH

Motion Compensated Video Compression Overview



Flow of information through the motion compensation process. As described in the section on Interframe Compression techniques, block based motion compensation uses blocks from a past frame to construct a replica of the current frame. The past frame is a frame that has already been transmitted to the receiver. For each block in the current frame a matching block is found in the past frame and if suitable, its motion vector is substituted for the block during transmission. Depending on the search threshold some blocks will be transmitted in their entirety rather than substituted by motion vectors. The problem of finding the most suitable block in the past frame is known as the block matching problem.

Block based motion compensated video compression takes place in a number of distinct stages. The flow chart above illustrates how the output from the earlier processes forms the input to later processes. Consequently choices made at early stages can have an impact of the effectiveness of later stages. To fully understand the issues involved with this type of video compression it is necessary to examine each of the stages in detail. These stages can be described as:

- ❖ Frame Segmentation
- ❖ Search Threshold

- ❖ Block Matching
- ❖ Motion Vector Correction
- ❖ Vector Coding
- ❖ Prediction Error Coding

Frame Segmentation

The current frame of video to be compressed is divided into equal sized non-overlapping rectangular blocks. Ideally the frame dimensions are multiples of the block size and square blocks are most common. Chan etc. used rectangular blocks of 16 x 8 pixels, claiming that blocks of this shape exploit the fact that motion within image sequences is more often in the horizontal direction than the vertical [C].

Block size affects the performance of compression techniques. The larger the block size, the fewer the number of blocks, and hence fewer motion vectors need to be transmitted. However, borders of moving objects do not normally coincide with the borders of blocks and so larger blocks require more correction data to be transmitted [L]. Small blocks result in a greater number of motion vectors, but each matching block is more likely to closely match its target and so less correction data is required. Lallaret et al. found that if the block size is too small then the compression system will be very sensitive to noise [L]. Thus block size represents a tradeoff between minimizing the number of motion vectors and maximizing the quality of the matching blocks. The relationship between block size, image quality, and compression ratio has been the subject of much research and is well understood.

For architectural reasons block sizes of integer powers of 2 are preferred and so block sizes of 8 and 16 pixels predominate. Both the MPEG and H.261 video compression standards use blocks of 16x16 pixels.

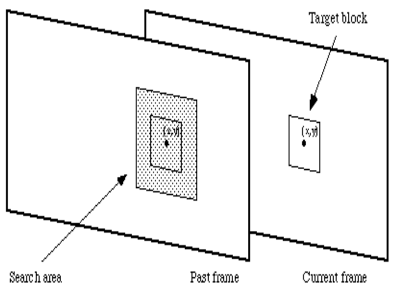
Search Threshold

If the difference between the target block and the candidate block at the same position in the past frame is below some threshold then it is assumed that no motion has taken place and a zero vector is returned. Thus the expense of a search is avoided. Most video codes employ a Threshold in order to determine if the computational effort of a search is warranted.

Block Matching

Block matching is the most time consuming part of the encoding process. During block matching each target block of the current frame is compared with a past frame in order to find a matching block. When the current frame is reconstructed by the receiver this matching block is used as a substitute for the block from the current frame.

Block matching takes place only on the luminance component of frames. The color components of the blocks are included when coding the frame but they are not usually used when evaluating the appropriateness of potential substitutes or *candidate blocks.*



Corresponding blocks from a current and past frame, and the search area in the past frame.

The search can be carried out on the entire past frame, but is usually restricted to a smaller *search area* centered on the position of the target block in the current frame (see above figure). This practice places an upper limit, known as the *maximum displacement*, on how far objects can move between frames, if they are to be coded effectively. The maximum displacement is specified as the maximum number of pixels in the horizontal and vertical directions that a candidate block can be from the position of the target block in the original frame.

The quality of the match can often be improved by interpolating pixels in the search area, effectively increasing the resolution within the search area by allowing hypothetical candidate blocks with fractional displacements.

The search area need not be square. Because motion is more likely in the horizontal direction than vertical, rectangular search areas are popular. The CLM460x MPEG video encoder (CCube 1992), for example, uses displacements of -106 to +99.5 pixels in the horizontal direction, and -58 to +51.5 pixels in the vertical. The half pixel accuracy is the result of the matching including interpolated pixels. The cheaper CLM4500, on the other hand, uses ±48 pixels in the horizontal direction, and ±24 in the vertical, again with half pixel accuracy.

If the block size is b and the maximum displacements in the horizontal and vertical directions are dx and dy respectively, then the search area will be of size $(2dx + b)(2dy + b)$. Excluding sub-pixel accuracy it will contain $(2dx + 1)(2dy+1)$ distinct, but overlapping, candidate blocks. Clearly the larger the allowable displacement the greater the probability of finding a good match. The number of candidate blocks in the search area, however, increases quadratic all as the displacement increases, which can result in a large number of candidate blocks being compared to the target block. Considering every candidate block in the search area as a potential match is known as an *Exhaustive* Search, *Brute Force* Search, or *Full* Search

Matching Criteria

In order for the compressed frame to look like the original, the substitute block must be as similar as possible to the one it replaces. Thus a *matching criterion*, or *distortion function*, is used to quantify the similarity between the target block and candidate blocks. If, due to a large search area, many candidate blocks are considered, then the matching criteria will be evaluated many times. Thus the choice of the matching criteria has an impact on the success of the compression. If the matching criterion is slow, for example, then the block matching will be slow. If the matching criterion results in bad matches then the quality of the compression will be adversely affected. Fortunately a number of matching criteria are suitable for use in video compression.

Sub-Optimal Block Matching Algorithms

The exhaustive search is computationally very intensive and requires the distortion function (matching criteria) to be evaluated many times for each target block to be matched. Considerable research has gone into developing block matching algorithms that find suitable matches for target blocks but require fewer evaluations. Such algorithms test only some of the candidate blocks from the search area and choose a match from this subset of blocks. Hence they are known as sub-optimal algorithms. Because they do not examine all of the candidate blocks, the choice of matching block might not be as good as that chosen by an exhaustive search. The quality-cost trade-off is usually worthwhile however.[There is a lot more information is available on block matching at this site.]

Motion vector correction

Once the best substitute, or *matching block*, has been found for the target block, a motion vector is calculated. The motion vector describes the location of the matching block from the past frame with reference to the position of the target block in the current frame. Motion vectors, irrespective of how they are determined, might not correspond to the actual motion in the scene. This may be due to noise, weaknesses in the matching algorithm, or local minima. The property that is exploited in spatially dependent algorithms can be utilized after the vectors have been calculated in an attempt to correct them. Smoothing techniques can be applied to the motion vectors that can detect erratic vectors and suggest alternatives. The alternative motion vectors can be used in place of those suggested by the BMA. Usually the candidate blocks to which they refer are first evaluated as potential matches and the corrected motion vector used only if the block is suitable.

Smoothing motion vectors, however, can add considerable complexity to a video compression algorithm and should only be used where the benefits outweigh these costs. If frames are going to be interpolated by the receiver then motion vector correction is likely to be worthwhile. Smoothing can also reduce the amount of data required to transmit the motion vector information, because this information is subsequently compressed and smooth vectors can be compressed more efficiently.

Vector smoothing causes problems of its own. Smoothing can cause small objects to be coded badly because their motion vectors might be considered erroneous when they are in fact correct. Smoothing such motion vectors can adversely affect the quality of the compressed image. Stiller found that averaging schemes blurred sharp discontinuities in image sequences. Because these discontinuities might be the result of object boundaries they should be preserved. To correct these problems Kim & Park developed a sophisticated correction process that yielded better results than other methods. Lallauret & Barba devised a technique that reconsidered every motion vector that did not concur with its immediate neighbors to the left and above.

Vector coding

Once determined, motion vectors must be assigned bit sequences to represent them. Because so much of the compressed data will consist of motion vectors, the efficiency with which they are coded has a great impact on the compression ratio. In fact up to 40% of the bits transmitted by a codec might be taken up with motion vector data. Fortunately, the high correlation between motion vectors and their non-uniform distribution makes them suitable for further compression. This compression must be lossless.

Efficient coding of motion vectors is a subject of research in its own right and many authors have offered suggestions on which techniques work best. Any one of the lossless general purpose compression algorithms is suitable for coding vectors. Kansu et al investigated the relative efficiency of Arithmetic, Adaptive Huffman, and Lempel-Ziv Coding. They found that the arithmetic and Huffman techniques performed best and that adaptive techniques using short term statistics performed better than those using long term statistics. The ISO/IEC video compression standard known as MPEG specifies variable length codes to be used for motion vectors. The zero vectors, for example, has a short code, because it is the most frequently occurring.

Lallauret & Barba tested two methods of coding motion vectors. The first was a predictive method where a prediction of the motion vector was calculated based on its predecessors in the same row and column. The prediction errors were then Huffman coded. The second technique grouped the motion vectors into blocks. If all the vectors in a block were the same, then only one was transmitted. Blocks that did not contain a homogenous set of vectors were labeled and the vectors described as per as their first method.

Prediction Error coding

Although the battery of techniques described thus far can code video very successfully, they rarely generate perfect replicas of the original frames. Thus the difference between a predicted frame and the original uncompressed frame might be coded. Generally this is applied on a block by block basis and only where portions of the coded frame are significantly different from the original. Transform coding is most frequently used to achieve this and completely lossless coding is rarely a goal.
Matching criteria

A *matching criterion*, or *distortion function*, is used to quantify the similarity between the target block and candidate blocks. A number of matching criteria are suitable for use in video compression.

Mean Absolute Difference (MAD)

The mean absolute difference (MAD) is the most popular block matching criterion. Corresponding pixels from each block are compared and their differences summed, as described by this equation.

$$\frac{1}{mn} \sum_{p=1}^{m}\sum_{q=1}^{n}\left|A[p,q]-B[p,q]\right|$$

Mean Absolute Difference function for two blocks A and B of size *nxm*. A[p,q] is the value of the pixel in the pth row and qth column of block A.

The lower the MAD the better the match and so the candidate block with the minimum MAD should be chosen. The function is alternatively called Mean Absolute Error (MAE).

Mean Square Difference (MSD)

The mean square difference function is similar to the mean absolute difference function, except that the difference between pixels is squared before summation.

$$\frac{1}{mn} \sum_{p=1}^{m}\sum_{q=1}^{n}\left(A[p,q]-B[p,q]\right)^2$$

Mean Squared Difference function

The mean square difference is more commonly called the mean square error (MSE) and the lower this value the better the match. This criterion is said to result in better matches than the MAD criterion although

Dionysian and Baker found the difference between it and Mean Square Difference (MSD) criterion was too slight to be perceived.

Pel Difference Classification (PDC)

The Pel Difference Classification (PDC) distortion function compares each pixel of the target block with its counterpart in the candidate block and classifies each pixel pair as either matching or not matching. Pixels are matching if the difference between their values is less than some threshold and the greater the number of matching pixels the better the match.

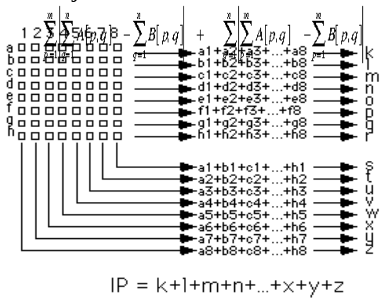
$$\sum_{q=1}^{n}\sum_{p=1}^{m}\left[ord\left(\left|A[p,q]-B[p,q]\right|\le t\right)\right]$$

Pel Difference Classification function. Ord (*e*) evaluates to 1 if *e* is true and false otherwise.
Chan et al. used a modified form of the PDC in which blocks were ruled out from the search by varying the threshold, beginning with an arbitrarily high threshold and halving it until only a few candidate blocks remain.

Integral Projection

In 1992 Kim and Park proposed using integral projections (IP) as a matching criterion. Integral projections are calculated by summing the values of pixels from each column and each row of a block. The most attractive feature of this criterion is that values calculated for a particular candidate block can be reused in calculating the integrals for overlapping candidate blocks. This feature is of particular value during an exhaustive search, but less useful in the case of sub-optimal searches. Kim and Park claimed that this choice of matching criterion is less susceptible to noise than others.

Integral Projection function



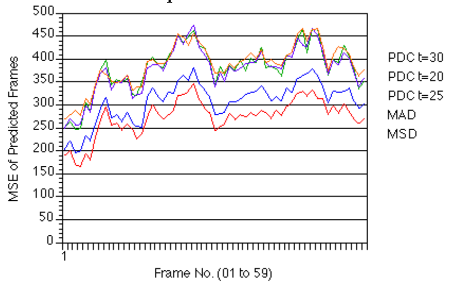
Integral Projection function

Matching Criteria

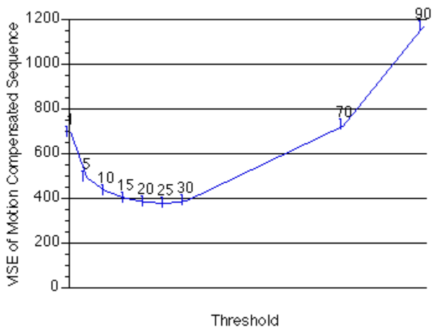
The choice of matching criteria is important since block matching might require the distortion function to be evaluated many thousands of times. Block matching algorithms vary in complexity. The MSD, for example, requires many multiplications whereas the MAD primarily uses additions. While multiplication might not have too great an impact on a software code, a hardware coder using MSE could be significantly more expensive than a hardware implementation of the MAD function.

If all matching criteria resulted in compressed video of the same quality then, of course, the least complex of these would always be used for block matching. However matching criteria often differ on the choice of substitute for the target block, with consequent variation in the quality of the coded frame.

The football sequence was used to compare matching criteria. MAD, MSD and PDC (with a variety of thresholds ranging from 1 to 90) were used as matching criteria to produce motion compensated approximations of each frame using the experimental method described. Each of the approximations was then compared to its counterpart in the actual sequence and the Mean Square Error calculated.



The results of the experiments, illustrated here, show different MSE values for approximated frames from the Football sequence produced by the MSD, MAD, PDC (thresholds of 20, 25, and 30) matching. Some frames proved more difficult to code than others. Perhaps this was because good matches for the target blocks in these frames were not available in the frames' predecessors and so less than ideal blocks were used to build each frame's approximation. It is noteworthy that all the matching criteria had problems with the same frames, but some matching criteria produced better approximated frames. It is clear that the MSD has the best matching capabilities and that the MAD is superior to the PDC criterion.



Above: PDC threshold versus MSE of Compressed Football Sequence

The graph above shows the results for all the thresholds tested with PDC. Of thresholds tested, 20 and 25 were the best choices. It is possible that some value between these was optimal, but the mean square errors for these values were so close that further investigation was unwarranted.

The choice of threshold has a great impact on the effectiveness of the Pel Difference Classification distortion function, but the best threshold for one sequence is not necessarily optimal for another sequence. Nelson tested a variety of thresholds on three sequences and found 5, 15, and 25 were optimal depending on the sequence. Chan et al. experimented with this PDC and found its performance varied depending on the type of motion in the video sequence. They suggested that smaller threshold values perform better than larger ones where the motion in the sequence is slow and that the inverse is also true.

Gharavi & Mills found that PDC performed marginally worse than MSD, but significantly outperformed MAD. The results from the experiments conducted for this thesis did not bear this out. This could be the result of a number of factors. Firstly every sequence is different and can produce different results. The two sequences used by Gharavi & Mills produced quite different results, although PDC outperformed MAD in both cases. Gharavi & Mills' results are based on images of sizes 360x288 and 256x240, whereas the football sequence is of size 160x120 pixels. The two most significant factors however are threshold and block size. Since Gharavi & Mills did not provide a mechanism for determining the optimal threshold *t* their results are difficult to reproduce. As above Figure shows, the value of *t* cannot be arbitrary. The difficulties PDC has in distinguishing good matches from bad are compounded by small block sizes. Block size is perhaps the most significant difference between the experiments using the football sequence (with block size of 8x8) and Gharavi & Mills' sequences (compressed using blocks of 16x16 pixels). The MAD and MSD matching criteria are not sensitive to the block size in the same way.

Advanced Block Matching Algorithms

There are a number of complex motion compensation techniques that have the potential to offer performance improvements over conventional block matching techniques. They are based on the same principles as the motion compensation techniques discussed in the previous chapter but they take a more general view of the motion compensationproblem.

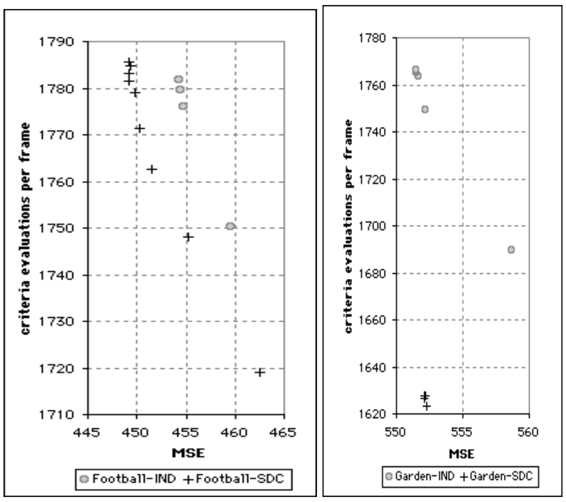
5. EXPERIMENTAL RESULTS AND DISCUSSIONS

Spatial Dependency (Type C)

Results

Type C spatial redundancy was more restrictive than type A. It exploited spatial redundancy only when the blocks above and to the left of the target had identical motion vectors. In such cases the algorithms began their searches at the position corresponding to these vectors. Otherwise the algorithms behaved normally and began their searches at the centre of the search area.

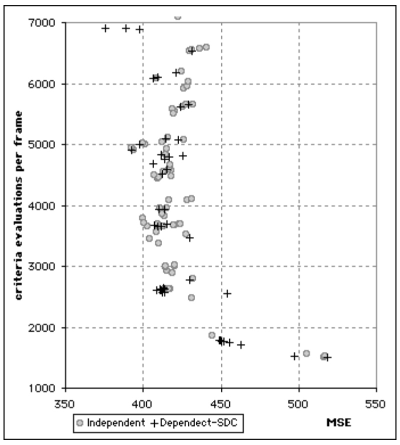
Spatial dependency of Type C was more successful than Type A. More importantly, however, although this spatial redundancy failed to improve the performance of some algorithms, it did not significantly deteriorate the performance of any.



Performance of Independent and Spatially Dependent OTS when coding the Football (above left) and Garden (above right) sequences.

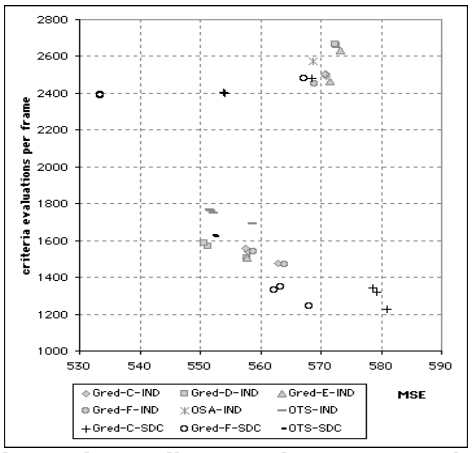
Spatial Dependency C improved the performance of the OTS on the Garden and Football sequence, but adversely affected the performance on the OTS when coding the Tennis sequence.

As can be seen from the graph below, the data for the spatially dependent algorithms approach the cost-quality curve of the independent algorithms. On only a few occasions did the dependent algorithms outperform the independent ones? The dependent algorithms at an MSE of about 420 that appear to have outperformed the independent algorithms are Greedy Algorithms C and F. The dependent data points beginning at an MSE of 450 are the result of the dependent OTS. Clearly, however, the spatial dependency did not significantly improve the performance of the algorithms.



Performance of Independent and Spatially Dependent (Type C) algorithms during coding of Football sequence.

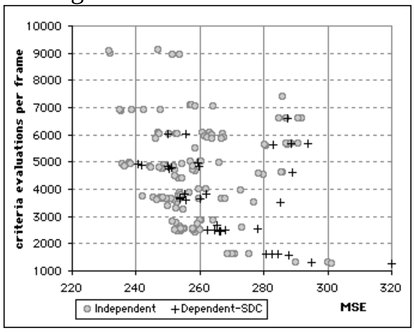
Spatial dependency (Type C) proved to be of assistance when coding the Garden sequence. Spatial dependent variations of Greedy Algorithms C and F generated images with competitive cost-quality performance. This can be seen from this graph:



Performance of Independent and Spatially Dependent (Type C) algorithms used to code the Garden sequence.

Spatial Dependency of this type did not assist the algorithms coding the Tennis sequence. Unlike the Type A spatial dependency, however, Type C did not cause the algorithms to perform disastrously.

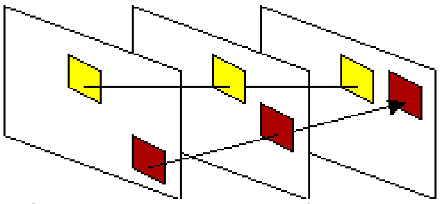
Thus spatial dependency type C failed to improve the performance of the algorithms. The concurring neighbor’s restriction (the difference between types A and C) improved the performance of the algorithms and it is likely that other dependent algorithms might also benefit from this modification.



Performance of Spatially Dependent (Type C) algorithms compared to independent algorithms used to code the Tennis sequence.

Temporal Dependency

If the assumption is made that moving objects in a scene move with constant velocity then a large degree of temporal redundancy can be expected. This redundancy can be exploited in a manner similar to spatial redundancy by providing predictions to block matching algorithms as to likely positions of matching blocks. By examining the motion of blocks in the previous frame(s), predictions about their behavior in the current frame can be made.

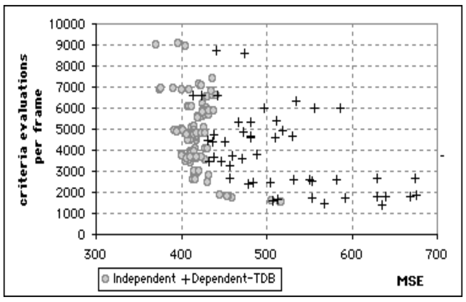


The distance and direction of object motion between frames tends to remain the same from frame to frame. Temporal redundancy exploits this tendency by assuming that motion vectors from previous frames indicate good starting points for searches in subsequent frames.

Ninomiya and Ohtsuka implemented a temporally dependent system that took into account the motion vector of the block at the same position in the previous frame. Puri et al devised a temporally dependent variation of the orthogonal search algorithm that also examined the motion vector of the same position in the previous frame. They found that the temporal dependency allowed the algorithm to achieve results similar to the independent variation but required fewer searches. Both spatial and temporal dependency can be exploited simultaneously.

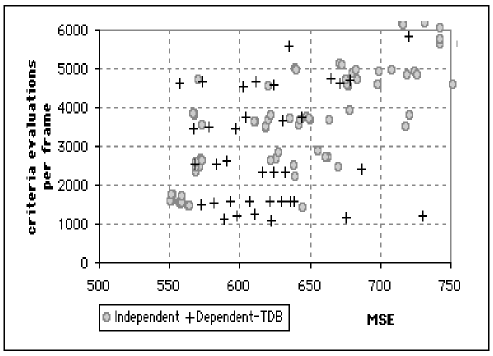
The kinds of block matching algorithms used on this page are frequently used when coding MPEG. There are however more advanced block matching techniques that are also very interesting. They are not suitable for use with MPEG however.

Football Sequence – TDB



Performance of Independent algorithms compared to Dependent algorithms when coding the Football sequence.

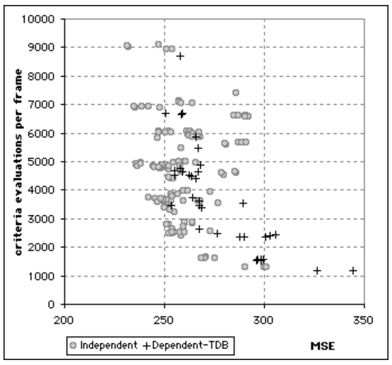
Garden Sequence – TDB



Performance of Independent algorithms compared to Dependent algorithms when coding the Garden sequence.

As the graphs above illustrate, temporal redundancy of the kind investigated failed to assist the block matching algorithms in finding blocks more economically. Some temporally dependent algorithms coded the Garden sequence using fewer criteria evaluations than their independent counterparts, but the resulting sequences were of lower quality. However they were not unreasonably lower.

Tennis Sequence – TDB



Performance of Independent algorithms compared to Dependent algorithms when coding the Garden sequence.

6. CONCLUTION

Finally, when coding each block of the predicted frame, the motion vector detailing the position (in the past frame) of the target block's match is encoded in place of the target block itself. Because fewer bits are required to code a motion vector than to code actual blocks, compression is achieved.

During decompression, the decoder uses the motion vectors to find the matching blocks in the past frame (which it has already received) and copies the matching blocks from the past frame into the appropriate positions in the approximation of the current frame, thus reconstructing the image. In the example used above, a perfect replica of the coded image can be reconstructed after decompression. In general this is not possible with block based motion compensation and thus the technique is lossy.

The effectiveness of compression techniques that use block based motion compensation depends on the extent to which the following assumptions hold.

- Objects move in a plane that is parallel to the camera plane. Thus the effects of zoom and object rotation are not considered, although tracking in the plane parallel to object motion is.
- Illumination is spatially and temporally uniform. That is, the level of lighting is constant throughout the image and does not change over time.
- Occlusion of one object by another, and uncovered background are not considered.

Bidirectional motion compensation uses matching blocks from both a past frame and a *future frame* to code the current frame. A future frame is a frame that is displayed after the current frame. Considering the chess board example, suppose that a player is fortunate enough to have a once lost queen replace a pawn on the board. If the queen does not appear on the board before the current move then no block containing the queen can be copied from the previous state of play and used to describe the current state. After the next move, however, the queen might be on the board. If in addition to the state of play immediately before the current move, the state of play immediately following is also available to the receiver, then the current image of the chess board can be reproduced by taking blocks from both the past and future frames.

Bidirectional compression is much more successful than compression that uses only a single past frame, because information that is not to be found in the past frame might be found in the future frame. This allows more blocks to be replaced by motion vectors. Bi-directional motion compensation, however, requires that frames be encoded and transmitted in a different order from which they will be displayed

REFERANCES

[1]. **[IrCo93]**Towards more picturesque speech - Irish Computer, p 24 - 27, November 1993

[2]. **[Ahma90]** M.H. AHMAD FADZIL, T.J. DENNIS A hierarchical motion estimator for interframe coding IEE Electronics Division Colloquium on 'Applications of Motion Compensation' on Monday, 8 October 1990 Digest No. 1990/128

[3]. **[Akan90]** Ali N. AKANSU, Jung Hui CHIEN, M.S. KADUR Lossless compression of block motion information in motion compensated video coding Proceedings of the International Society for Optical Engineering (SPIE), Vol. 1199, p 30 - 38, 1989 ISBN: 0-8194-0238-9

[4]. **[Bier88]** M. BIERLING Displacement estimation by hierarchical block matching Visual Communications and Image Processing, Proceedings of the SPIE, Volume 1001, part 2, p 942 - 951, 1988

[5]. **[Bowl85]** Carl D. BOWLING, Richard A. JONES Motion compensated image coding with a combined maximum a posteriori and regression algorithm IEEE Transactions on Communications, Vol. COM-33, No. 8, p 844 - 857, August 1985

[6]. **[Chan94]** Eric CHAN, Arturo A. RODRIGUEZ, Rakeshkumar GHANDI, Sethuraman PANCHANATHAN Experiments on block-matching techniques for video coding Multimedia Systems, Volume 2, Number 5, p 228 - 241, December 1994

[7]. **[Chen88]** T.C. CHEN, Kou-Hu TZOU, P.E. FLEISCHER A hierarchical HDTV coding system using a DCPM-PCM approach Visual Communications and Image Processing 88, Proceedings of the SPIE, Volume 1001, part 2, p 804 - 811

[8]. **[Choi89]** Woo Young CHOI, Rae-Hong PARK Motion vector coding with conditional transmission Signal Processing, Vol. 18, No. 3, November 1989

[9]. **[Cook90]** David A. COOK A history of narrative film 2nd Edition, p2 W.W. Norton & Compnay Inc. New York, 1990 ISBN: 0-393-95553-2

[10]. **[DelR90]** Vincenzo DEL RE, Giovanni ZARONE A modified 2D-logarithmic search procedure for a motion compensation and presegmented predictive coding Signal Processing V : Theories and Applications, Proceedings of EUSIPCO-90, Fifth European Signal Processing Conference, Barcelona, September 18-21, 1990, Volume II, p 789- 792 Editors L. TORRES, E. MASGRAU, M.A. LAGUNAS Elsevier Science Publishers B.V. ISBN : 0-444-88636-2

[11]. **[DeWi92]** Peter N. H. DE WITH A simple recursive motion estimation technique for compression of HDTV signals 4th International Conference on Image Processing and its Applications, Maastricht, Netherlands, 7 - 9 April 1992 Published by Institution of Electrical Engineers (IEE), London ISBN 0 85296 543 5, ISSN 0537-9989

[12]. **[Fros90]** T.M.E. FROST, C.J. THEAKER Moving object detection and motion estimation IEE Electronics Division Colloquium on 'Applications of Motion Compensation' on Monday, 8 October 1990 Digest No. 1990/128

[13]. **[Fuh91]** Chiou-Shann FUH, Petros MARAGOS Motion displacement estimation using an affine model for image matching Optical Engineering, Vol. 30, No.7, p 881 - 886, July 1991

[14]. **[Gerk89]** P. GERKEN, D. ADOLPH Improved motion-compensating prediction with rotational block matching Third International Conference on Image Processing and its Applications , Warwick, UK, 18-20 July 1989. p 290 - 294 London: IEE, Conf. Publ. No.307

[15]. **[Ghan90]** M. GHANBARI The cross-search algorithm for motion estimation IEEE Transactions on Communications, Volume 38, No 7, p.950-953, July 1990

[16]. **[Gilg88]** Michael GILGE A high quality videophone coder using hierarchical motion estimation and structure coding of the prediction error Visual Communications and Image Processing 88, Proceedings of the SPIE, Volume 1001, part 2, p 864 - 874

[17]. **[Hsie90]** C.H. HSIEH, P.C. LU, J.S. SHYN, E.H. LU Motion estimation algorithm using interlock correlation Electronics Letters, Vol. 26, No. 5, p 276 - 277, 1 March 1990

[18]. **[Huff52]** D.A. HUFFMAN A method for the construction of minimum redundancy codes. Proc. IRE 40, p 1098 - 1101, 1952

[19]. **[IrCo93]** Towards more picturesque speech Irish Computer, p 24 - 27, November 1993

[20]. **[Jain81]** Jaswant R. JAIN, Anil K. JAIN Displacement measurement and its application in interframe image coding IEEE Transactions on Communications, Volume COM-29, Number 12, p 1799 - 1808, December 1981

[21]. **[Jain91]** S.H. JANG, S.D. KIM Backward predictive block matching algorithm for low bit rate video coding Electronics Letters, Vol. 27, No. 18, p 1674 - 1676, 29 August 1991

[22]. **[Jaur90]** Fernando JAUREGUÍZAR, José I. RONDA, Narciso GARCÍA Motion compensated prediction on digital HDTV Signal Processing V : Theories and Applications, Proceedings of EUSIPCO-90, Fifth European Signal Processing Conference, Barcelona, September 18-21, 1990, Volume II, P 753 - 756 Editors L. TORRES, E. MASGRAU, M.A. LAGUNAS Elsevier Science Publishers B.V. ISBN : 0-444-88636-2

[23]. **[Jaur92]** Fernando JAUREGUÍZAR, José I. RONDA, Narciso GARCÍA Statistical analysis of the motion compensated hybrid transform encoding of HDTV signals Signal Processing of HDTV III Edited by H. YASUDA, L. CHIARLIONE Elsevier Science Publishers B.V.,1992 ISBN:0-444-89491-8

[24]. **[Kim92]** Joon-Seek KIM, Rae-Hong PARK A fast feature-based block matching algorithm using integral projections IEEE Journal on Selected Areas in Communications, Volume SAC-10, Number 5, p 968 - 971 , June 1992 ISSN 0733-8716

[25]. **[Kim93]** Joon-Seek KIM, Rae-Hong PARK Local motion-adaptive interpolation technique based on block matching algorithms Signal Processing : Image Communication, Volume 4, Issue 6, p 519 - 528, 1992 ISSN : 0923-5965

[26]. **[Koga81]** T. KOGA, K. IINUMA, A. HIRANO, Y. IIJIMA, T. ISHIGURO Motion-compensated interframe coding for video conferencing Proceedings NTC'81 (IEEE), p G.5.3.1 - G.5.3.4

[27]. **[Lall91]** Fabrice LALLAURET, Dominique BARBA Motion compensation by block matching and vector post-processing in sub-band coding of TV signals at 15 Mbit/s Proc. SPIE, Vol. 1605, p 26 - 36 ISBN: 0-8194-0742-9

[28]. **[Lee93]** Liang-Wei LEE, Jhing-Fa WANG, Jau-Yien LEE, Jung-Dar SHIE Dynamic search-window adjustment and interlaced search for block-matching algorithm IEEE Transactions on Circuits and Systems for Video Technology, Vol. 3, No. 1, p 85 - 87, February 1993

[29]. **[Lin84]** C.M. LIN, S.C. KWATRA Motion compensated interframe color image coding International Conference on Communications, Part 1, p 516 - 520 1988 Amsterdam ISBN : 0-444-87522-0

[30]. **[Lin95]** David W. LIN, Cheng-Tie CHEN, T. Russel HSING Video on phone lines : technology and applications Proceedings of the IEEE, Volume 83, Number 2, p 175 - 193, February 1995

[31]. **[Nels93]** Giles J NELSON The Block Matching Approach to Motion Estimation M.Sc. Dissertation, Department of Computer Science, University of Warwick, UK. September 1993

[32]. **[Netr88]** Arun N. NETRAVALI, Barry G. HASKELL Digital pictures: representation and compression Plenum Publishing Corporation, New York ISBN: 0-306-42791-5

[33]. **[Puri87]** A. PURI, H.-M. HANG, D.L. SCHILLING An efficient block-matching algorithm for motion compensated coding Proceedings IEEE ICASSP (International Conference on Acoustics, Speech and Signal Processing), p 25.4.1 - 24.4.4, 1987

[34]. **[Reut89]** T. REUTER A modified block matching algorithm with vector reliability checking and adaptive smoothing Third International Conference on Image Processing and its Applications , Warwick, UK, 18-20 July 1989. p 295 - 299 London: IEE, Conf. Publ. No.307

[35]. **[Sefe92]** V. SEFERIDIS, M. GHANBARI Hierarchical motion estimation using texture analysis 4th International Conference on Image Processing and its Applications, Maastricht, Netherlands, 7 - 9 April 1992 Published by Institution of Electrical Engineers (IEE), London ISBN 0 85296 543 5, ISSN 0537-9989

[36]. **[Shel92]** Kenneth M. SHELDON The VideoPhone goes home computer Byte, Volume 17, Num. 4, p 51, April 1992

[37]. **[Stew95]** Daniel Robert STEWART Department of History, University of California at Berkeley Unpublished Communication, 20 September 1995

[38]. **[Stil90]** Christoph STILLER Motion-estimation for coding of moving video at 8 Kbit/s with Gibbs modeled vector field smoothing Proceedings of the International Society for Optical Engineering (SPIE), Vol. 1360, p 468 - 476, 1990 ISBN: 0-8194-0421-7

[39]. **[Toba95]** Fouad A. TOBAGI Distance learning with digital video IEEE Multimedia, p 90 - 93, Spring 1995

[40]. **[Wang92]** Q.WANG, R.J. CLARKE Motion estimation and compensation for image sequence coding Signal Processing: Image Communication, Volume 4, Number 2, p 161 - 174, April 1992

[41]. **[IrCo93]**Towards more picturesque speech - Irish Computer, p 24 - 27, November 1993

[42]. **[Ahma90]** M.H. AHMAD FADZIL, T.J. DENNIS A hierarchical motion estimator for interframe coding IEE Electronics Division Colloquium on 'Applications of Motion Compensation' on Monday, 8 October 1990 Digest No. 1990/128

[43]. **[Akan90]** Ali N. AKANSU, Jung Hui CHIEN, M.S. KADUR Lossless compression of block motion information in motion compensated video coding Proceedings of the International Society for Optical Engineering (SPIE), Vol. 1199, p 30 - 38, 1989 ISBN: 0-8194-0238-9

[44]. **[Bier88]** M. BIERLING Displacement estimation by hierarchical block matching Visual Communications and Image Processing, Proceedings of the SPIE, Volume 1001, part 2, p 942 - 951, 1988

[45]. **[Bowl85]** Carl D. BOWLING, Richard A. JONES Motion compensated image coding with a combined maximum a posteriori and regression algorithm IEEE Transactions on Communications, Vol. COM-33, No. 8, p 844 - 857, August 1985

[46]. **[Chan94]** Eric CHAN, Arturo A. RODRIGUEZ, Rakeshkumar GHANDI, Sethuraman PANCHANATHAN Experiments on block-matching techniques for video coding Multimedia Systems, Volume 2, Number 5, p 228 - 241, December 1994

[47]. **[Chen88]** T.C. CHEN, Kou-Hu TZOU, P.E. FLEISCHER A hierarchical HDTV coding system using a DCPM-PCM approach Visual Communications and Image Processing 88, Proceedings of the SPIE, Volume 1001, part 2, p 804 - 811

[48]. **[Choi89]** Woo Young CHOI, Rae-Hong PARK Motion vector coding with conditional transmission Signal Processing, Vol. 18, No. 3, November 1989

[49]. **[Cook90]** David A. COOK A history of narrative film 2nd Edition, p2 W.W. Norton & Compnay Inc. New York, 1990 ISBN: 0-393-95553-2

[50]. **[DelR90]** Vincenzo DEL RE, Giovanni ZARONE A modified 2D-logarithmic search procedure for a motion compensation and presegmented predictive coding Signal Processing V : Theories and Applications, Proceedings of EUSIPCO-90, Fifth European Signal Processing Conference, Barcelona, September 18-21, 1990, Volume II, p 789- 792 Editors L. TORRES, E. MASGRAU, M.A. LAGUNAS Elsevier Science Publishers B.V. ISBN : 0-444-88636-2

[51]. **[DeWi92]** Peter N. H. DE WITH A simple recursive motion estimation technique for compression of HDTV signals 4th International Conference on Image Processing and its Applications, Maastricht, Netherlands, 7 - 9 April 1992 Published by Institution of Electrical Engineers (IEE), London ISBN 0 85296 543 5, ISSN 0537-9989

[52]. **[Fros90]** T.M.E. FROST, C.J. THEAKER Moving object detection and motion estimation IEE Electronics Division Colloquium on 'Applications of Motion Compensation' on Monday, 8 October 1990 Digest No. 1990/128

[53]. **[Fuh91]** Chiou-Shann FUH, Petro's MARAGOS Motion displacement estimation using an affine model for image matching Optical Engineering, Vol. 30, No.7, p 881 - 886, July 1991

[54]. **[Gerk89]** P. GERKEN, D. ADOLPH Improved motion-compensating prediction with rotational block matching Third International Conference on Image Processing and its Applications , Warwick, UK, 18-20 July 1989. p 290 - 294 London: IEE, Conf. Publ. No.307

[55]. **[Ghan90]** M. GHANBARI The cross-search algorithm for motion estimation IEEE Transactions on Communications, Volume 38, No 7, p.950-953, July 1990

[56]. **[Gilg88]** Michael GILGE A high quality videophone coder using hierarchical motion estimation and structure coding of the prediction error Visual Communications and Image Processing 88, Proceedings of the SPIE, Volume 1001, part 2, p 864 - 874

[57]. **[Hsie90]** C.H. HSIEH, P.C. LU, J.S. SHYN, E.H. LU Motion estimation algorithm using interlock correlation Electronics Letters, Vol. 26, No. 5, p 276 - 277, 1 March 1990

[58]. **[Huff52]** D.A. HUFFMAN A method for the construction of minimum redundancy codes. Proc. IRE 40, p 1098 - 1101, 1952

[59]. **[IrCo93]** Towards more picturesque speech Irish Computer, p 24 - 27, November 1993

[60]. **[Jain81]** Jaswant R. JAIN, Anil K. JAIN Displacement measurement and its application in interframe image coding IEEE Transactions on Communications, Volume COM-29, Number 12, p 1799 - 1808, December 1981

[61]. **[Jain91]** S.H. JANG, S.D. KIM Backward predictive block matching algorithm for low bit rate video coding Electronics Letters, Vol. 27, No. 18, p 1674 - 1676, 29 August 1991

[62]. **[Jaur90]** Fernando JAUREGUÍZAR, José I. RONDA, Narciso GARCÍA Motion compensated prediction on digital HDTV Signal Processing V : Theories and Applications, Proceedings of EUSIPCO-90, Fifth European Signal Processing Conference, Barcelona, September 18-21, 1990, Volume II, P 753 - 756 Editors L. TORRES, E. MASGRAU, M.A. LAGUNAS Elsevier Science Publishers B.V. ISBN : 0-444-88636-2

[63]. **[Jaur92]** Fernando JAUREGUÍZAR, José I. RONDA, Narciso GARCÍA Statistical analysis of the motion compensated hybrid transform encoding of HDTV signals Signal Processing of HDTV III Edited by H. YASUDA, L. CHIARLIONE Elsevier Science Publishers B.V.,1992 ISBN:0-444-89491-8

[64]. **[Kim92]** Joon-Seek KIM, Rae-Hong PARK A fast feature-based block matching algorithm using integral projections IEEE Journal on Selected Areas in Communications, Volume SAC-10, Number 5, p 968 - 971 , June 1992 ISSN 0733-8716

[65]. **[Kim93]** Joon-Seek KIM, Rae-Hong PARK Local motion-adaptive interpolation technique based on block matching algorithms Signal Processing : Image Communication, Volume 4, Issue 6, p 519 - 528, 1992 ISSN : 0923-5965

[66]. **[Koga81]** T. KOGA, K. IINUMA, A. HIRANO, Y. IIJIMA, T. ISHIGURO Motion-compensated interframe coding for video conferencing Proceedings NTC'81 (IEEE), p G.5.3.1 - G.5.3.4

[67]. **[Lall91]** Fabrice LALLAURET, Dominique BARBA Motion compensation by block matching and vector post-processing in sub-band coding of TV signals at 15 Mbit/s Proc. SPIE, Vol. 1605, p 26 - 36 ISBN: 0-8194-0742-9

[68]. **[Lee93]** Liang-Wei LEE, Jhing-Fa WANG, Jau-Yien LEE, Jung-Dar SHIE Dynamic search-window adjustment and interlaced search for block-matching algorithm IEEE Transactions on Circuits and Systems for Video Technology, Vol. 3, No. 1, p 85 - 87, February 1993

[69]. **[Lin84]** C.M. LIN, S.C. KWATRA Motion compensated interframe color image coding International Conference on Communications, Part 1, p 516 - 520 1988 Amsterdam ISBN : 0-444-87522-0

[70]. **[Lin95]** David W. LIN, Cheng-Tie CHEN, T. Russel HSING Video on phone lines : technology and applications Proceedings of the IEEE, Volume 83, Number 2, p 175 - 193, February 1995

[71]. **[Nels93]** Giles J NELSON The Block Matching Approach to Motion Estimation M.Sc. Dissertation, Department of Computer Science, University of Warwick, UK. September 1993

[72]. **[Netr88]** Arun N. NETRAVALI, Barry G. HASKELL Digital pictures: representation and compression Plenum Publishing Corporation, New York ISBN: 0-306-42791-5