Optical Engineering, Vol. 35, No. 1, pp. 128-136, Jan 1996

Multiresolution Motion Estimation Techniques for Video Compression

M. K. Mandal, E. Chan, X. Wang and S. Panchanathan
Visual Computing and Communications Laboratory
Department of Electrical and Computer Engineering
University of Ottawa, Ottawa
Ontario, Canada - K1N 6N5
e-mail : mandalm@marge.genie.uottawa.ca
Ph. No. - (613) 562-5800 Ext. 6233, Fax - (613) 562-5175

ABSTRACT

Wavelet transform is a valuable tool in video processing applications because of its flexibility in representing nonstationary signals. Wavelet-based compression has the advantages of efficient decorrelation of image frames and reduced complexity multiresolution motion estimation (MRME). In this paper, we propose three techniques to improve motion estimation in a wavelet-based coder. First, we propose to use an adaptive threshold (AMRME) for coding the motion vectors of the highpass subimages. Secondly, we propose a bi-directional motion estimation (BMRME) technique in the wavelet transform domain. In BMRME, we estimate the temporal (*i.e.* direction information) flags only for the blocks in the lowest resolution subimages and use the same information for the corresponding blocks in the higher resolution subimages. Finally, we propose a fast multiresolution motion estimation (FMRME) technique where the set of directional subimages at each level of the wavelet pyramid are combined together into a single subimage. Multiresolution motion estimation is then performed on the newly formed subimages. The proposed techniques improve the coding performance significantly over the baseline MRME tecnique. In addition, they further reduce the computational complexity of the MRME technique.

Key Words : Video Compression, Motion Estimation, Wavelets, Multiresolution Motion Estimation.

1. INTRODUCTION

Digital video transmission is becoming increasingly important with the advent of broadband networks such as Integrated Services Digital Network (ISDN), Asynchronous Transfer Mode (ATM), *etc*. We note that digital video data are voluminous and hence efficient video compression techniques are essential for video archival and transmission. The International Standard Organization (ISO) has recently proposed the MPEG standards for video compression [1]. MPEG-1 [2] has been developed for a targeted data rate of 1.5 Mbits/s. MPEG-2 [3] is an improved version of MPEG-1 and is expected to be used in a variety of applications. These standards employ a block-based motion estimation technique to reduce the temporal redundancy present in a video sequence. To further reduce the spatial redundancy present in the motion compensated frames, discrete cosine transform (DCT) is used. We note that motion estimation in MPEG is a compute intensive task. In addition, DCT has the drawbacks of blocking artifacts, mosquito noise and aliasing distortions at high compression ratios.

Recently, discrete wavelet transform (DWT) has become popular in video coding applications because of several reasons [4, 5]. First, it is efficient in representing videos which are in general nonstationary in nature. Secondly, wavelets have high decorrelation and energy compaction efficiency. Thirdly, the blocking artifacts and mosquito noise are absent in wavelet-based coder, resulting in subjectively pleasing reconstructed images. Fourthly, aliasing distortion can be reduced significantly with proper choice of wavelet filters. Finally, the basis functions match the human visual system (HVS) characteristics.

In DWT coding, the choice of the motion estimation algorithm is crucial and determines the coding performance and complexity of the coder [6]. Zhang *et al* [7] have proposed a multiresolution motion estimation (MRME) technique for estimating the motion vectors in the wavelet domain. This technique estimates the motion vectors hierarchically from lower resolution to higher resolution subimages and thus reduces the complexity significantly. However, there is a potential to further improve the coding performance and the complexity of the MRME technique.

In this paper, we propose several techniques to improve the coding performance of the baseline MRME technique. First, we propose to employ an adaptive threshold for coding the motion vectors in the MRME framework. Since the correlation between the corresponding highpass subimages of the neighbouring frames is not significant, AMRME improves the coding performance of the MRME technique. We then propose a bi-directional motion estimation (BMRME) technique in the wavelet domain. The proposed BMRME technique can be employed in the design of MPEG-like wavelet coder. Finally, we propose a fast multiresolution motion estimation (FMRME) technique which has a superior coding performance at a reduced complexity.

This paper is organized as follows. Section 2 provides a brief description of wavelets and MRME technique. In Section 3, the proposed algorithms are detailed. The simulation results are provided in Section 4 which is followed by conclusions in Section 5.

2. WAVELETS and MRME TECHNIQUE

Wavelet transform represents any arbitrary function as superposition of a family of basis functions called *wavelets*. A family of basis functions can be generated by translating and dilating the *mother wavelet* corresponding to that family. The main advantages of wavelets are : i) compactly supported basis functions, ii) adaptive time-frequency window, and iii) multiresolution capability. Forward DWT can be realized by using a two channel filterbank as shown in Fig. 1. The signal is passed through a lowpass (LPF) and a highpass filter (HPF) and the outputs of the filter are decimated by two. For reconstruction, the coefficients are upsampled and passed through another set of lowpass and highpass filters. We note that for orthonormal wavelets, the LPF and HPF are quadrature mirror filters (QMF).

The 2-D DWT is usually calculated using a separable approach [8]. Fig. 2 shows a 3-level wavelet decomposition of an image S_1 of size $a \times b$ pixels. In the first level of decomposition, one low pass subimage (S_2) and three orientation selective highpass subimages (W_2^H, W_2^V, W_2^D) are created. In second level of decomposition, the lowpass subimage is further decomposed into one lowpass and three highpass subimages (W_4^H, W_4^V, W_4^D). This process is repeated on the low pass subimage to form higher level wavelet decomposition. In other words, DWT decomposes an image into a pyramid structure of subimages with various resolutions corresponding to the different scales (see Fig. 3). The inverse wavelet transform is calculated in the reverse manner, *i.e.* starting from the lowest resolution subimages, the higher resolution images are calculated recursively.

A typical wavelet-based video coding scheme is shown in Fig. 4. The coder consists of three major modules : wavelet transform, motion compensation and quantization. Each frame is wavelet-decomposed upto 3-5 stages. The temporal redundancy that exist in a video sequence is removed by motion compensation. The error frames are then quantized for encoding. It should be noted that other wavelet-based coding schemes do exist. For example, motion compensation can be done before wavelet decomposition or, the residual error frames can be further decorrelated by applying DCT. However, it has been concluded in [7] that the coding scheme shown in Fig. 4, provides superior coding performance compared to the techniques just mentioned.

Several motion estimation techniques have been reported in the literature. However, block matching is widely used because of its simplicity. In the block matching process, the current frame (t) of a video sequence is divided into blocks of size $n \times n$ pixels as shown in Fig. 5. For each block (reference block) in the current frame (t), the previous frame $(t-1)$ is searched within a neighborhood (search area) in order to obtain the best match block with respect to a prespecified error criterion such as mean square error (MSE) defined as follows:

$$MSE\ (u,v) = \sum_{i=1}^{n}\sum_{j=1}^{n} | S(i+u, j+v) - R(i,j) |^2 \qquad -p \le u,v \le p \tag{1}$$

where $R(i,j)$ is a reference block's pixel in the current frame (t) and $S(i+u, j+v)$ is a candidate block's pixel within a search area in the previous frame $(t-1)$. We note that the search area consists of $(n+2p)\times(n+2p)$ pixels where p is the maximum allowed displacement, and the total number of possible candidate blocks is $(2p+1)^2$. The most intuitive approach for block matching is to use the full search algorithm (FSA). For each reference block, all possible $(2p+1)^2$ candidate blocks are searched to obtain the best match which is used as a prediction estimate for the reference block. The relative displacement between the reference block and the best match block constitutes the motion vector which is transmitted to the receiver. We note that the execution of the FSA is a computationally expensive procedure. Recently, several fast methods have been proposed for block-based motion estimation using logarithmic or hierarchical search [9]. However, these algorithms may converge to a local optimum which corresponds to the inaccurate prediction of the motion vectors resulting in a poor performance.

Recently, a multiresolution motion estimation scheme (MRME) has been reported for wavelet-based video compression [7]. This approach exploits the multiresolution property of the wavelet pyramid in order to reduce the computational complexity of the motion estimation process. In the MRME scheme, the motion vectors at the highest level of the wavelet pyramid are first estimated using the conventional block matching based motion estimation algorithm. Then the motion vectors at the next level of the wavelet pyramid are predicted from the motion vectors of the preceding level which are refined at each step. For example, the motion vectors in W_4^H, W_4^V and W_4^D are predicted from the motion vectors in W_8^H, W_8^V and W_8^D, respectively. The motion vectors of the lower level pyramid can be estimated as follows :

$$V_4^o(x,y) = 2V_8^o(x,y) + \Delta_4^o(x,y) \tag{2}$$

$$V_2^o(x,y) = V_4^o(x,y) + 2V_8^o(x,y) + \Delta_2^o(x,y) \tag{3}$$

where $V_i^o(x,y)$ represents the motion vector of the reference block centered at (x,y) for o -orientation subimage for various levels of pyramid. The incremental motion vector, $\Delta_i(x,y)$ is calculated within a reduced search area centered at $2V_8^o(x,y)$ and $\left(V_4^o(x,y) + 2V_8^o(x,y)\right)$ for level-2 and level-1 pyramids, respectively. The subimages of level-3, level-2 and level-1 pyramids, are divided into small blocks of size $n \times n$, $2n \times 2n$ and $4n \times 4n$, respectively. With this structure, the number of blocks in all subimages is identical. As a result, there is a one to one correspondence between the blocks at various levels of wavelet pyramid. The search window for level-3, level-2 and level-1 subimages are p, $p/2$ and $p/4$, respectively. The refinement of the motion estimation process is shown in Fig. 6. Table-1 compares the complexity of the FSA with the MRME technique. It is observed that the hierarchical prediction and refinement provides a superior motion compensation at a significantly reduced complexity. The complexity of the MRME technique has been calculated by assuming that the local motion vectors have been searched using FSA. The complexity of the MRME can further be reduced using a logarithmic or hierarchical search.

3. PROPOSED TECHNIQUES

The MRME technique described in the previous section reduces the complexity compared to the FSA. In this section, we propose several techniques to further improve the coding performance of the MRME technique. In Section 3.1, we propose to use an adaptive threshold while coding the motion vectors. A bi-directional MRME scheme is then proposed in Section 3.2. Finally, a fast MRME technique is proposed in Section 3.3. The performance evaluation of the proposed techniques are presented in the next section.

3.1 Adaptive Thresholding Technique

In the MRME technique, the motion vectors corresponding to all the subimages are calculated. However, we note that the high frequency subimages of consecutive frames are not correlated even though the consecutive frames are very similar. This is due to the following reasons. Firstly, DWT is not translational invariant, *i.e.* if an image is shifted by a pixel, the transform coefficients will not be shifted by 1 pixel (in the wavelet domain). As a result, the object motion in the spatial domain does not correspond to the translation of coefficients in the wavelet domain. This is especially true for highpass subimages. For lowpass subimages, a great similarity among neighbouring images have been observed mainly because the basis functions are much more smoother and thus the effects of translations are averaged. Secondly, the highpass subimages represent only the edge information and is hence expected to change rapidly from image to image even with a small change in scene. Although, the highpass subimages cannot be predicted well, the MRME approach performs well in practice as most of the information (or energy) is contained in the lowpass subimages.

We propose to use an adaptive thresholding (AMRME) approach for estimating motion vectors in the highfrequency subimages. If the dissimilarity between the reference block and the best match is greater than a threshold, the block is discarded and the motion vectors corresponding to that block are not coded. A zero block is initialized in the corresponding place at the decoder. This thresholding improves the coding performance in two ways. First, the number of motion vectors to be coded will be less, resulting in a reduced bit-rate. Secondly, the objective quality of the reconstructed will improve since we are discarding the mismatched block.

The choice of threshold is crucial in achieving a good performance. Since, we have assumed "PSNR" as the measure of quality, the adaptive threshold can be related to the energy of the reference block. If the threshold is made equal to the energy of the reference block, the image frames will be reconstructed with least distortion (in L^2 metric). However, we note that the rate-distortion function may not be optimal since the entropy of the motion vectors may be high. Usually, a threshold less than the energy of the block provides better result. We define *thresh_factor* as the ratio of the threshold and the energy of the block. Depending on the video sequence, a thresh_factor between 0.6-0.9 provides good coding performance. It should be noted that the complexity of the AMRME is marginally higher compared to MRME, since one has to calculate only the energy of the block.

3.2 Bi-directional Motion Estimation

To achieve a good coding performance, the MPEG standard [1] suggests that the frames be divided into three categories : I, P and B frames. I frames are coded independent of others. For P frames, the motion vectors are estimated by comparing the P frame with a previous I frame. The motion vectors and the prediction error frame are then sent to the decoder for reconstruction. B frames are compared with the previous and the next reference frame (I or P frame) and usually three types of predictions are used : forward prediction, backward prediction and average of two macroblocks [1]. After the motion estimation is performed, the motion vectors are sent to the decoder. We note that for a fast moving sequence, the motion vectors alone may not provide a good subjective quality of the reconstructed video-frames. Reducing the block-size, increasing the search window and sending the error frames help in improving the subjective quality. However, they also increase the bit-rate. Our simulation results indicate that it is more beneficial to send the quantized error frames. Hence, we have implemented an adaptive wavelet-based video coder which sends the error frames corresponding to B-frames, only when it is necessary, *i.e.* when the difference between the reconstructed I and B frames exceed a certain threshold.

Although, bi-directional motion estimation improves the coding performance, it also increases the complexity of the coder. More memory is needed in the decoder to store the future prediction reference frame (backward frame). An extra picture delay is introduced. Most of all, the computational complexity is twice that of unidirectional estimation. In the multiresolution framework, the motion vectors are estimated hierarchically from lower resolution to higher resolution subimages. Hence, there is a potential to reduce the computational complexity of bi-directional motion estimation with minimal degradation in performance. We assume a simple version of MPEG bi-directional motion estimation. Only the forward and backward motion estimation are considered in this paper. Although, this may degrade coding performance marginally, there are several advantages. First, the encoder sends only one motion vector (instead of two in case of averaging). Secondly, the decoder will have reduced complexity since it does not have to average the two macro blocks.

In the proposed bi-directional motion estimation, a temporal flag (TFLAG) will have one of two states : 0 or 1, depending on whether the best match is from the previous or from the future reference frame. Since, a small region of an image is represented by blocks from various directional subimages, it is highly probable that the temporal flags will be identical for all the corresponding blocks from the subimages of a particular orientation. Hence, in the BMRME technique, we propose to calculate the temporal flags only in the four lowest resolution subimages (*i.e.* S_8, W_8^H, W_8^V and W_8^D). The knowledge of the temporal flags of the highpass subimages at the first level is then used as an estimate for the higher resolution (lower level pyramid) subimages in the same orientation. The procedure can be summarized as follows :

1. For each block position (x,y), the motion vectors are calculated for all the subimages of the highest level pyramid with respect to both the previous and the next frame. Initialize $V_8^o(x,y)$, $o \in \{H,V,D\}$, with the better matching motion vectors (between two frames).

2. Set $TFLAG_8^o(x,y)=0$, $o \in \{H,V,D\}$, if the reference block matches better with the block from the previous frame. Else, intialize $TFLAG_8^o(x,y)=1$.

3. For subimages corresponding to higher level of the pyramid, TFLAG is no more calculated. The motion vectors $V_4^o(x,y)$, $V_2^o(x,y)$ are estimated using Eq. 2 and 3 with respect to the previous frame if $TFLAG_8^o(x,y)=0$ else the motion vectors are estimated with respect to the next reference frame.

The complexity of the proposed algorithm is twice the complexity of the MRME technique (which is unidirectional) for the first four subimages and identical to that of MRME for the other six subimages. The overall complexity of MRME/BMRME is compared in Table-1. The complexity is given in operations/pixel (for MSE criteria, an *operation* includes one subtraction, one multiplication and one addition). It is observed that the complexity of BMRME is marginally higher than that of MRME.

To compare the bitrate of MRME and BMRME, we calculate the number of blocks in each subimage as

$$m = \frac{M}{2^L * n} * \frac{N}{2^L * n} \tag{4}$$

where M and N are, respectively, the number of rows and columns of an image frame and $n \times n$ is the block-size at the highest level of the pyramid. The number of motion vectors (each vector has two components : horizontal and vertical) and temporal flags required to be transmitted, are shown in Table-2. The dynamic range of the motion vectors (each component) and temporal flags are shown in column 2. We observe that the number of motion vectors in MRME and BMRME are identical. However, BMRME has small overhead of transmitting the temporal flags (for the first four subimages).

3.3 Fast MRME

We recall that DWT decomposes an image into a pyramid structure of subimages. The motion vectors for different orientation subimages at each level of the wavelet pyramid actually describe the same part of an object in a scene. In other words, the motion activities of the different wavelet subimages at the same pyramid level are highly correlated because they represent the motion in the same scale. In the proposed scheme (FMRME), the set of wavelet components at each level of the pyramid are combined into a single all-orientation subimage. In other words, W_8^H, W_8^V, W_8^D are combined into W_8^A, whereas W_4^H, W_4^V, W_4^D are combined into W_4^A, *etc*. This process is illustrated in Fig. 7. We note that the motion estimation is performed only on the all-orientation subimages (W_8^A, W_4^A, W_2^A, *etc.*). This contrasts with the MRME scheme where the motion estimation is separately performed on all the individual wavelet subimages (W_8^H, W_8^V, W_8^D, *etc.*). Hence the FMRME scheme exploits the correlation among the motion vectors of the subimages of a pyramid level.

The motion activities at different levels of the pyramid are highly correlated since they actually characterize the same motion structure at different scales. Hence, in FMRME, the motion vectors of all-orientation subimage of a lower level pyramid, are predicted and refined from the motion vectors of all-orientation subimage of higher level pyramid. For a three stage decomposition, the motion vectors of differents levels of pyramid can be expressed as

$$V_4^A(x,y) = 2V_8^A(x,y) + \Delta_4^A(x,y) \tag{5}$$

$$V_2^A(x,y) = V_4^A(x,y) + 2V_8^A(x,y) + \Delta_2^A(x,y) \tag{6}$$

At the receiver, the motion vectors of the all-orientation subimage W_i^A are assigned as the motion vectors of W_i^H, W_i^V and W_i^D in order to reconstruct each wavelet subimage, *i.e.* :

$$V_i^H(x,y) = V_i^V(x,y) = V_i^D(x,y) = V_i^A(x,y) \qquad for\ i = 2, 4, 8 \tag{7}$$

Table 2 compares the number of motion vectors (each vector has a horizontal and vertical component), required to be sent to the decoder. We observe that the number of motion vectors are almost 40% of that required for MRME.

4. PERFORMANCE OF THE PROPOSED TECHNIQUES

Performance of the proposed techniques have been evaluated with three test sequences - i) Miss America (CIF format, 360×288), ii) Salesman (360×288) and iii) Pingpong (360×240). Miss America and Salesman are typical video conferencing sequences with slow motion and low spatial details. On the other hand, Pingpong has high spatial details and fast motion. The basic video coder as shown in Fig. 4 has been employed in the simulations. The full search algorithm with *mean square error* (MSE) as the matching criterion has been used to obtain the motion vectors. The block size of the level-1, level-2 and level-3 subimages have been chosen 3×3, 6×6 and 12×12, respectively. The maximum allowed displacement for level-1 (S_8 and W_8) subimages is 4 pixels and the maximum allowed refinements allowed in level-2 (W_4) and level-3 (W_2) subimages are 2 and 1 pixel, respectively. The *peak signal to noise ratio* (PSNR) which is defined as

$$PSNR \ (in\ dB) = 10 \log_{10}\left(\frac{255^2}{MSE}\right)$$

has been employed as a measure of the quality of the reconstructed images. The coding performance of various techniques have been compared with respect to bit-rate versus PSNR. The bit-rate and PSNR have been calculated by averaging the bit-rate and MSE for all (I, P and B) the fames. In order to calculate the bit-rate, a frame rate of 24 frames/second has been assumed. The GOP has been taken as IBBBBPBBBBPBBBBI (*i.e.* four B-frames between consecutive reference frames).

The choice of wavelets and quantization scheme are crucial in achieving good performance. In all our simulations, we have used the *Daubechies-8* tap wavelet, as it provides a good coding performance [10]. The wavelet coefficients (or the motion compensated error coefficients) have been quantized with a fairly simple uniform quantizer [12]. For each level of the pyramid (4 subimages for the lowest resolution level and 3 subimages for higher resolution level) one quantizer has been used. The ratio of the quantization step-sizes of the successive levels have been chosen as 2.0 (increasing from lower resolution to higher resolution subimages). The quantized coefficients are encoded with an arithmetic coder [11] to achieve superior coding performance.

The coding performance of AMRME depends on the choice of the threshold. Fig. 8 compares the coding performance for various *thresh_factor*. It is observed that the best *thresh_factor* is in the range 0.6-0.8 for most sequences. For Miss America (and also for Salesman which is not shown here), the performance variation is sensitive on the choice of threshold. Miss America (and Salesman), being a low entropy sequence, requires less number of bits for encoding. Hence the bit-saving due to thresholding is able to improve the overall coding performance. However, Pingpong being a high entropy sequence, requires a large number of bits for encoding. Hence, the bit saving due to thresholding is not very significant compared to overall bit-rates. In all our subsequent simulation, we have used a threshold with *thresh_factor* equal to 0.7 for all the three sequences.

The coding performance of AMRME (with *thresh_factor*=0.7) is shown in Fig. 9. It is observed that AMRME provides an improvement of more than 1 dB compared to MRME. The improvement is more significant for Pingpong sequence, since the highpass subimages of the consecutive frames are less correlated due to its fast motion.

Fig. 10 compares the coding performance of B-frames for the MRME and BMRME techniques (using adaptive threshold in both the cases). We observe that BMRME provides a superior coding performance compared to MRME. In addition, BMRME addresses the uncovered areas better, since an uncovered area can be predicted only from the future reference frame. Hence, BMRME technique has a potential to provide a superior performance when there is a scene change in video sequence.

The relative performance of MRME and FMRME techniques is shown in Fig. 11. In general, FMRME is expected to provide poor performance compared to MRME technique because of its coarse motion estimation. However, we note that the FMRME requires less number of motion vectors to be encoded and thus save some bits. In Fig. 11, it is observed that for the *Pingpong* sequence, MRME provides a better coding performance. Since, Pingpong is a sequence with fast motion, the corresponding motion vectors from different subimages are less correlated and hence superior performance is achieved by estimating motion individually for all the subimages. However, it is observed that FMRME provides a coding performance comparable to MRME for *Miss America* and *Salesman* sequences. For slow sequences such as *Salesman* and *Miss America*, there is correlation among the motion vectors of the three subimages at any level of the wavelet pyramid. Hence the bit-rate saving due to the lower number of motion vectors compensates the degradations due to coarse motion estimation. It is also observed that FMRME works better for very low bit-rate coding. At lower bit-rate, FMRME spends fewer bits in coding motion vectors and spends the remaining bits in encoding the error coefficients. In summary, for slow motion sequences, the overall coding performance of FMRME is comparable to that of MRME at a significantly reduced complexity.

Finally, the overall coding performance, combining all the three techniques, is shown in Fig. 12. We observe that an improvement of 1-2 dB in coding performance is obtained with the proposed techniques compared to MRME for all the video sequences. We also note that the overall complexity of the proposed techniques is significantly less compared to MRME.

5. CONCLUSIONS

In this paper, we have first presented the use of adaptive threshold for coding the motion vectors in highpass subimages. This threshold improves the coding performance of the MRME technique as there is little correlation among the highpass coefficients of two neighbouring frames. The improvement is more appreciable for fast motion sequences. Secondly, we have proposed a bi-directional multiresolution motion estimation technique for a wavelet transform based video coder. The BMRME technique provides a superior coding performance (improvement of more than 1 dB) as compared to the MRME technique. The performance improvement due to BMRME over MRME is more pronounced for a fast motion sequence. The proposed technique can be employed in the design of a wavelet-based MPEG coder. Finally, we have proposed a fast multiresolution motion estimation (FMRME) technique. This technique exploits the correlation among the motion vectors of the subimages of a particular wavelet pyramid and significantly reduces the computational complexity (about 60%). In addition, the number of motion vectors are also reduced. As a result, FMRME provides a superior coding performance compared to MRME, especially for a slow motion sequence.

Further work can be carried out to extend the MRME/BMRME techniques to a generalized wavepacket decomposition scheme. In wavepackets, the decomposition is image adaptive and irregular in nature, resulting in a superior coding performance [10]. The extension of MRME techniques to wavepackets has a potential to further improve the coding performance.

6. REFERENCES

1. D. L. Gall, "MPEG : A Video Compression Standard for Multimedia Applications," *Communications of the ACM*, Vol. 34, No. 4, pp. 46-58 (April 1991).
2. MPEG (Moving Pictures Expert Group), Final Text for ISO/IEC 11172, *Information Technology - Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s*, ISO/IEC (1993).
3. ISO/IEC JTC1/SC29/WG11/N0400, "Test Model 5", Version 2 (April 1993).
4. W. R. Zettler, J. Huffman and D. C. P. Linden, "Application of Compactly Supported Wavelets to Image Compression," *Proc. of SPIE : Image Processing Algorithms and Techniques*, Vol. 1244, pp. 150-160 (1990).
5. A. N. Akansu and R. A. Haddad, *Multiresolution Signal Decomposition : Transform, Subbands and Wavelets*, Academic Press (1993).
6. A. Puri, H. M. Hang and D. L. Schilling, "An Efficient Block Matching Algorithm for Motion-Compensated Coding," *Proc. of the ICASSP*, pp. 1063-1066 (1987).
7. Y. Q. Zhang and S. Zafar, "Motion-Compensated Wavelet Transform Coding for Color Video Compression," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 2, No. 3, pp. 285-296 (Sept 1992).
8. S. G. Mallat, "A Theory for Multiresolution Signal Representation: The Wavelet Decomposition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 7, pp. 674-693 (July 1989).
9. E. Chan, A. A. Rodriguez, R. Gandhi and S. Panchanathan, "Experiments on Block-Matching Techniques for Video Coding," *Multimedia Systems*, Vol. 2, No. 3, pp. 228-241 (1994).
10. M. K. Mandal, S. Panchanathan and T. Aboulnasr, “Wavelets for Image Compression,” *Proc. of IEEE International Symposium on Time-Frequency and Time-Scale Analysis*, pp. 338-341 (Oct 1994).
11. I. H. Witten, R. M. Neal and J. G. Cleary, "Arithmetic Coding for Data Compression," *Communications of the ACM*, Vol 30, No. 6, pp 520-540 (June 1987).
12. R. A. Devore, B. Jawerth and B. J. Lucier, "Image Compression through Wavelet Coding," *IEEE Trans. on Information Theory*, Vol. 38, No. 12, pp. 719-746 (March 1992).

Table 1 : Computational complexity of various motion estimation algorithms (in operations/pixel)

Full Search B-ME		Multiresolution ME (p=4)			
p=8	p=16	MRME	FMRME	BMRME	BFMRME
$8p^2$	$8p^2$	$0.6p^2+1.5p$	$0.25p^2+0.5p$	$0.9p^2+1.5p$	$0.4p^2+0.6p$
512	2048	15.6	6.0	20.4	8.8

Table 2 : No. of motion vectors in MRME/BMRME algorithms (m = number of blocks in a subimage)

Level	Dynamic range of vectors	MRME	FMRME	BMRME	BFMRME
1	-4, 4	4*m	2*m	4*m	2*m
2	-2, 2	3*m	1*m	3*m	1*m
3	-1, 1	3*m	1*m	3*m	1*m
overhead[1]	0, 1	Nil	Nil	2*m	1*m

[1]Overhead due to time flags

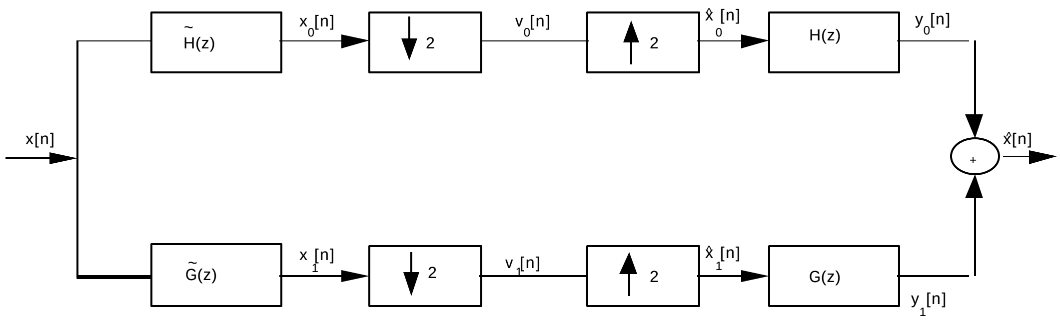


Figure 1 : 1-D Wavelet Decomp osition and Reconstruction

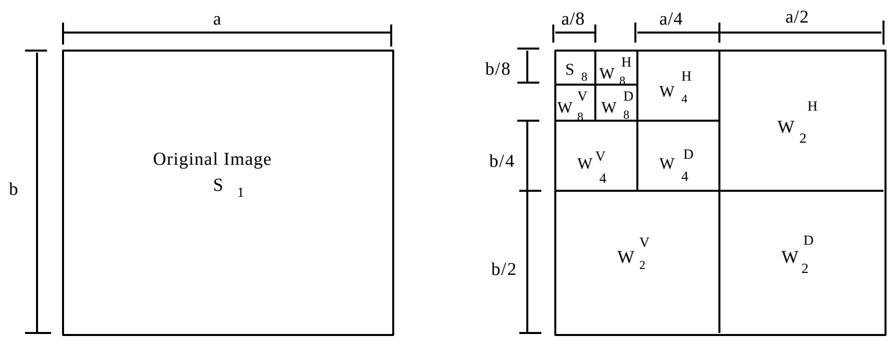


Figure 2 : Wavelet Transformed Image

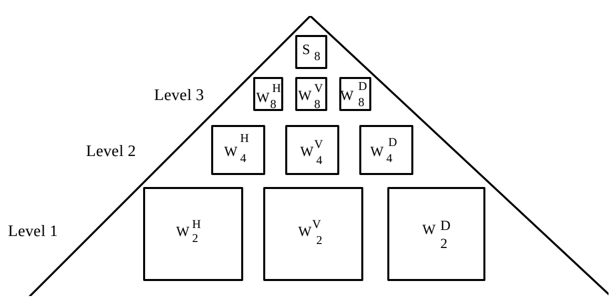


Figure 3 : Wavelet Pyramid

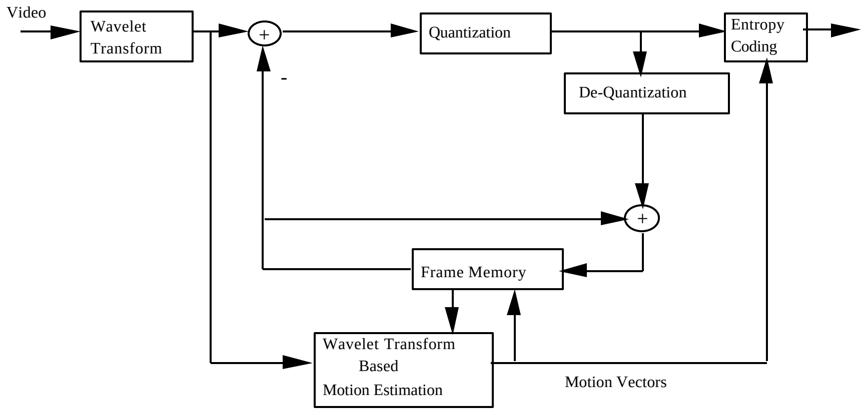


Figure 4 : A Simple Wavelet-based Video Coder

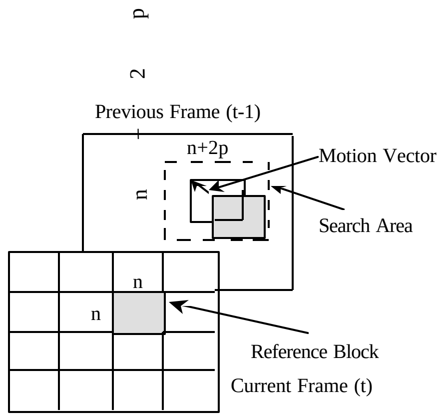


Figure 5 : Block Matching Motion Estimation Process

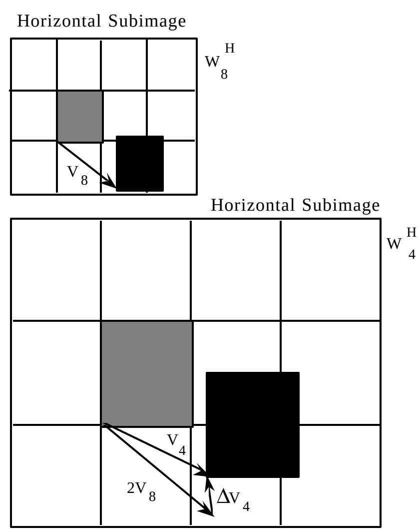


Figure 6 : Multiresolution Motion Estimation

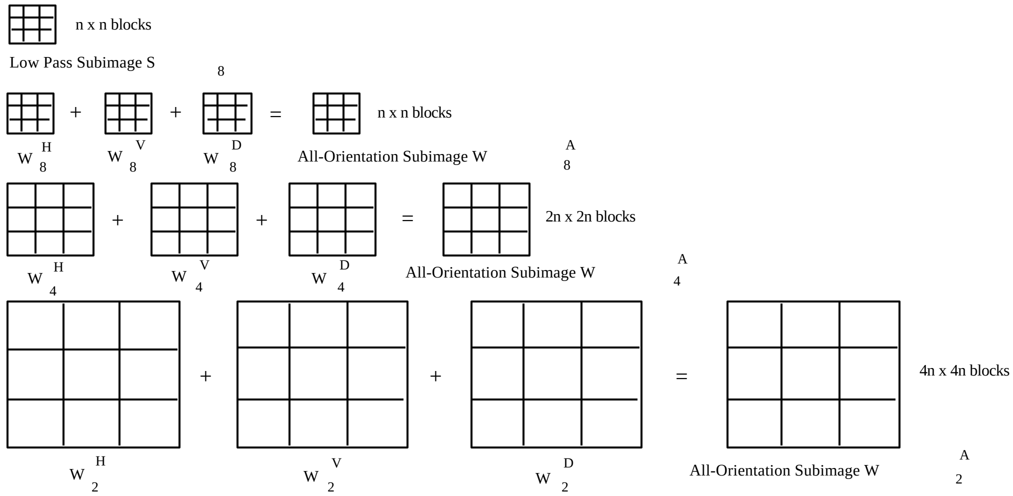
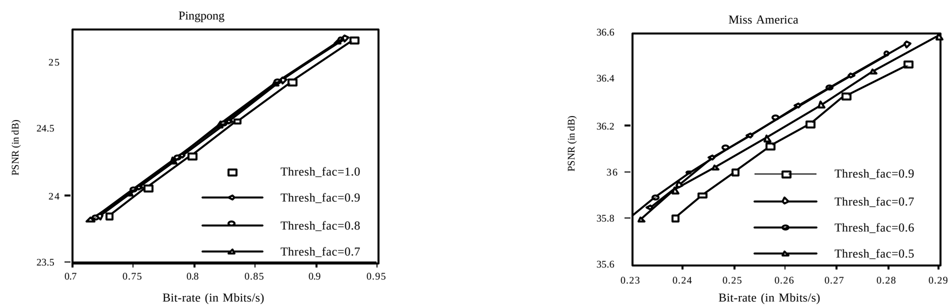
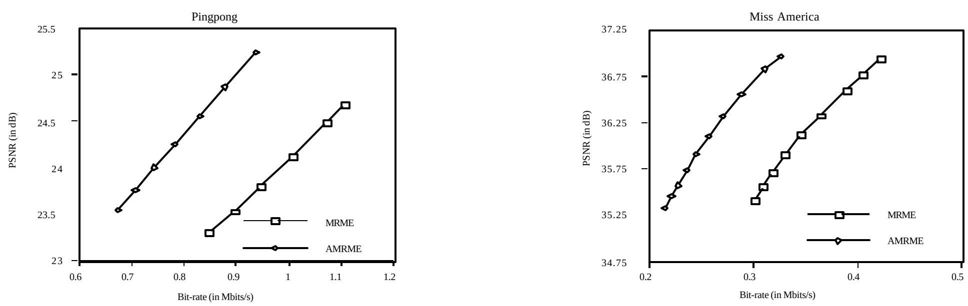


Figure 7 : Wavelet All-Orientation Subimages



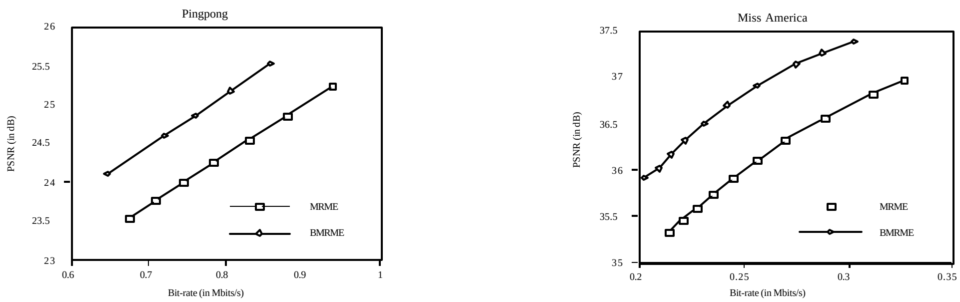
(a) (b)

Figure 8 : Performance comparison of various thresholds.



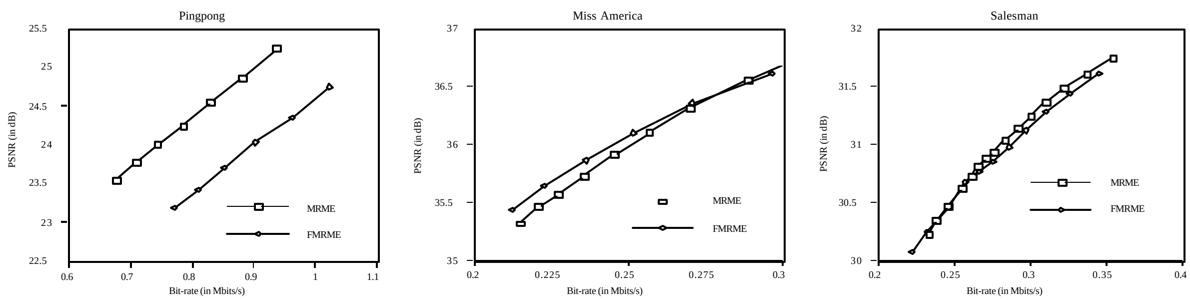
a) b)

Figure 9 : Comparison of MRME and AMRME (with thresh_factor=0.7) Techniques



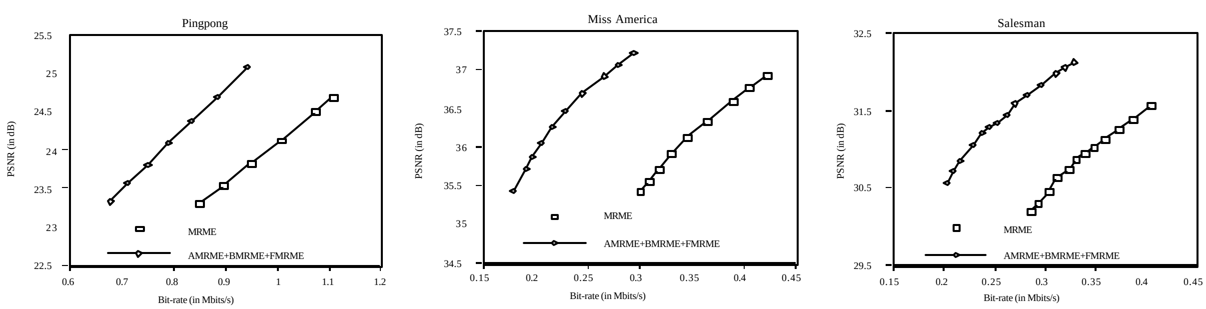
(a) (b)

Figure 10 : Comparison of MRME and BMRME techniques.
In all cases, a thresh_factor of 0.7 has been used.



(a) (b) (c)

Figure 11 : Comparison of MRME and FMRME Techniques.
In all cases, a thresh_factor of 0.7 has been used.



(a) (b) (c)

Figure 12 : Performance of the combined three techniques.