HEWLETT PACKARD

A Low Bit Rate Video Codec

Wei Li*, Vasudev Bhaskaran
Media Technology Laboratory
HPL-93-102
November, 1993

email: li@ltssun2.epfl.ch
email: bhaskara@hplvab.hpl.hp.com

video coding, low bit rate, morphological segmentation, wavelet

A video coding/decoding system for applications requiring low bit rate has been developed. This coding scheme uses an intraframe coder for the initial frame in the video sequence and subsequent frames are coded using an interframe coding method. A wavelet-based technique is used for intraframe coding. For interframe coding, displaced frame differences are computed and coded using a segmentation based method wherein the displaced frame is segmented into active and inactive regions. To meet the low bit rate requirements, the motion vectors are processed so as to reduce their contribution to the overall bit rate. Preprocessing techniques are also employed to generate a smooth motion-vector field. To reduce coding artifacts, post-processing techniques are developed for use at the decoder. Coding results for 30 fps CIF resolution sequences at 60 kbits/sec and 15 fps QCIF sequences at 16 kbits/sec are discussed in this report.

*Signal Processing Laboratory, Swiss Federal Institute of Technology, Lausanne, Switzerland

1 Introduction

The advent of multimedia has evidenced a merger of computer technology and television technology. This merger has resulted in the emergence of several applications such as teleconferencing, videophone and video-on-demand. These applications would not be possible without an efficient video compression algorithm. Several international standardization activities are aiming at developing high performance video compression techniques for different applications, e.g. H.261 for video conferencing, MPEG1 for CD-ROM based applications, MPEG2 for brodacast TV etc. Currently the MPEG standardization group has started an investigative effort towards developing a standard (currently referred to as MPEG4) for low bit rate video compression. The objective of MPEG4 standard is to devise a video coding scheme that will deliver good quality video at frame rates around 10 fps or higher and at spatial resolutions of QCIF or higher and the compressed bit stream should use a bit rate lower than 64kbits/sec (as of the writing of this report, the requirements for MPEG4 have not been completely defined).

H.261, MPEG1 and MPEG2 essentially use the same framework(see Fig. 1). The intraframes are coded using a DCT based algorithm. Motion compensation is applied on the interframes, and the resulting displaced frame difference (DFD) is coded using a DCT based algorithm. Motion vectors are coded differentially using a variable length code (VLC).

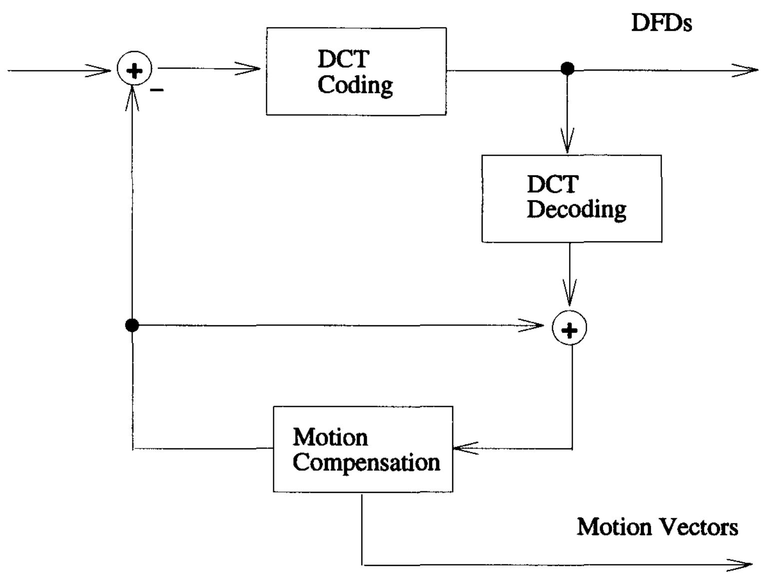


Figure 1: Generic Video Coder

In this report, a different approach will be described for coding video sequences. This scheme is particularly suitable for coding video at low bit rates and is capable of operating within the target suggested for MPEG4. In the proposed scheme, intraframes are coded using a wavelet transform method. A multi-grid motion estimation algorithm is used for estimating motion. The motion vectors are used in the computation of the DFD for the interframes. The DFDs are then coded using morphological segmentation (this is contrasted with the MPEG approach wherein the DFDs are coded using a DCT scheme). To reduce the bitrate contributions due to motion-vectors, we perform preprocessing of the video followed by motion-vector processing; the latter yields a more uniform motion-vector field. At the decoder, the video is postprocessed to reduce the visibility of coding artifacts.

In Section 2 we discuss the DFD's correlation characteristics. Due to the weak spatial correlation found in DFD's, we claim that the DCT is not an efficient method for coding the DFD's. In Section 3 we describe an improved coding scheme for the DFDs. Section 4 describes a complete coding scheme for low bit rate video. The performance of this coder is given in Section 5. Conclusions and future research directions can be found in Section 6.

2 DFD Statistics

As was pointed out in [1] and [2], correlation in DFDs are relatively small (less than 0.4) compared to typical natural images (0.9 to 0.95). A careful examination on data shows that the correlation is not isotropic and it might be *motion dependent.* A higher correlation may exist along the motion trajectory. It is also observed that de-interlacing procedure can result in higher vertical correlation.

The autocorrelation function $\rho(x, y)$ for an image block $Z(i, j)$ of size $L \times L$ can be defined as follows:

$$\rho(x,y) = \frac{\sum_{i=1}^{L-x}\sum_{j=1}^{L-y} Z(i,j)\cdot Z(i+x,j+y)}{\sum_{i=1}^{L}\sum_{j=1}^{L} Z^2(i,j)} \quad (1)$$

Using Eq.(1) with $L = 8$, autocorrelation coefficients are computed along horizontal, vertical and diagonal directions for the first 20 de-interlaced frames of *Table Tennis* and *Flower Garden.* The average correlation coefficients are reported in Table 1. Only the first-lag coefficients are shown, the higher lag coefficients being negligible. Both sequences show higher vertical correlation due to the de-interlacing procedure.

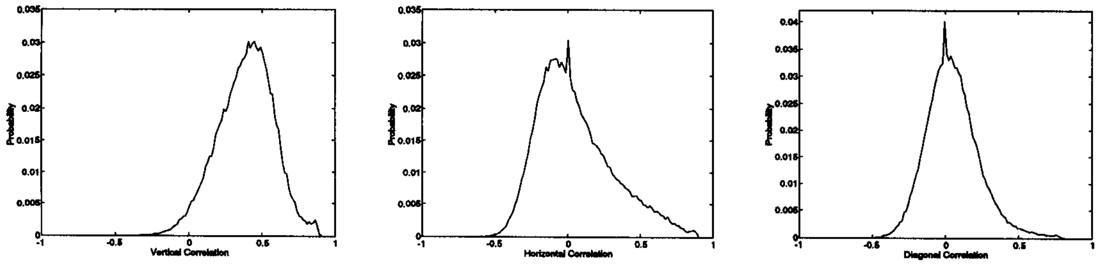
mean	$\rho(1,0)$	$\rho(0,1)$	$\rho(1,1)$
TableTennis	0.389	0.051	0.049
FlowerGarden	0.446	0.375	0.154

Table 1: Average autocorrelation coefficients. High vertical correlation is due to de-interlacing procedure.

Figure 2 and 3 show histograms of the correlation coefficients. It can be seen that although the correlation tends to be small, some image blocks do have high correlations.

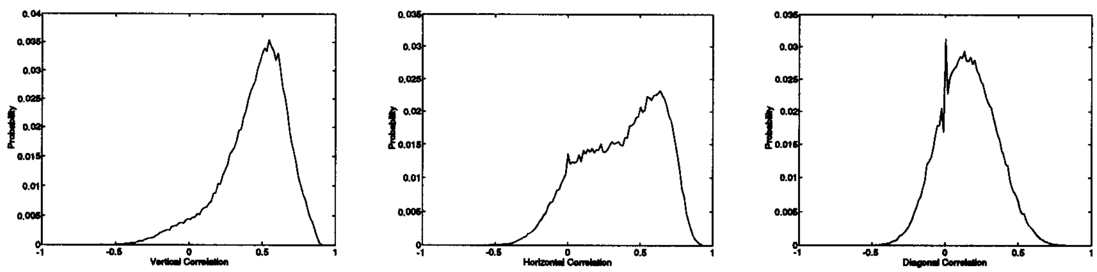
In Table 1 we notice that DFD of *Flower Garden* shows a high vertical correlation, which coincides with the heavy horizontal motion (panning). It is then reasonable to guess that if the sequences are not de-interlaced, then *Table Tennis* will show only vertical correlation (due to the vertical motion of the pingpong, the racket and the arm), while *Flower Garden* shows only horizontal correlation. Unfortunately, this is not generally true.

The first 20 even fields of each sequence are taken as data. The same computations give us Table 2 and Fig. 4 and 5. Notice that fields of *Flower Garden* still has a significant horizontal correlation, but fields of *Table Tennis* have little correlation in both horizontal and vertical directions. At this point, no general conclusions can be developed for the relation between the motion direction and the correlation direction. This is an area for further research.



(a) (b) (c)

Figure 2: The histogram of the three largest autocorrelation coefficients for *Table Tennis.* They are computed on 8×8 blocks. (a) $\rho(1,0)$; (b) $\rho(0,1)$; (c) $\rho(1,1)$.

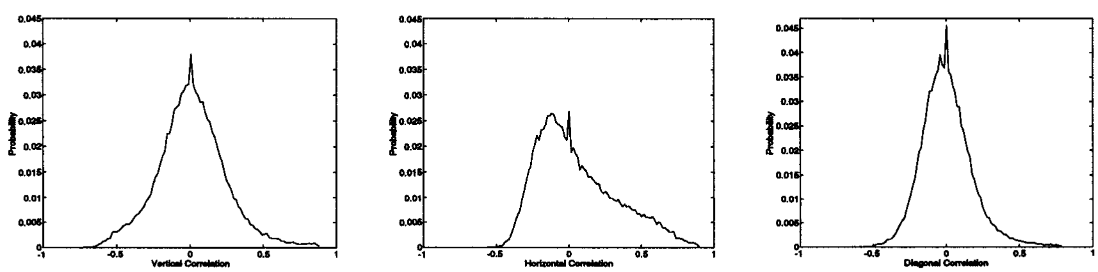


(a) (b) (c)

Figure 3: The histogram of the three largest autocorrelation coefficients for *Flower Garden.* They are computed on 8×8 blocks. (a) $\rho(1,0)$; (b) $\rho(0,1)$; (c) $\rho(1,1)$.

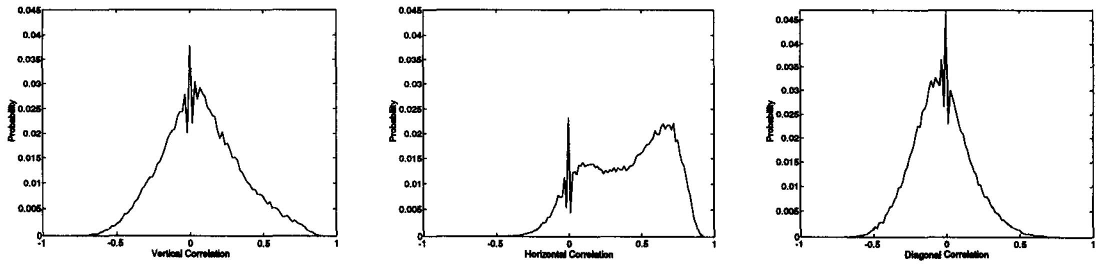
mean	$\rho(1,0)$	$\rho(0,1)$	$\rho(1,1)$
TableTennis	0.0130	0.0556	0.0025
FlowerGarden	0.0767	0.405	0.0025

Table 2: Average autocorrelation coefficients computed on fields only.Vertical correlation is very small now.



(a) (b) (c)

Figure 4: The histogram of the three largest autocorrelation coefficients for *Table Tennis.* Only even fields are used in computation. They are computed on 8×8 blocks. (a) $\rho(1,0)$; (b) $\rho(0,1)$; (c) $\rho(1,1)$.



(a) (b) (c)

Figure 5: The histogram of the three largest autocorrelation coefficients for *Flower Garden*. Only even fields are used in computation. They are computed on 8×8 blocks. (a) $\rho(1,0)$; (b) $\rho(0,1)$; (c) $\rho(1,1)$.

3 Segmentation Based DFD Coding

As was discussed in the previous section, DFDs possess little correlation. Therefore, DCT-based coding will not be efficient. In [2] and [3], a segmentation based method using mathematical morphology is proposed to code the DFDs. The major advantages of this method include:

1. preservation of edges in moving objects;
2. amenable to an efficient implementation as most of the morphological operators have fast algorithms;
3. reduced operations count compared with the conventional approach which requires performing a two-dimensional transform such as the 2D DCT; the proposed method uses simple spatial quantization and coding of the DFDs.

Basic operations of the algorithm is shown in Fig. 6.

Fig. 7 shows a simple example of the algorithm.

In [2] and [3], adaptive arithmetic coder with zeroth order Markov model [4] was used. In this work, we'll show that the coding performance can be significantly improved by using more powerful arithmetic coders. Two aspects are pursued: (1) using higher order models; (2) establish a good initial statistical model.

Using Higher-Order Statistical Models

Using the arithmetic coder described in [5], any order of the Markov model can be used. From simulations we found that using first order model instead of zeroth order will save up to 10% the coding cost (see Table 3); increasing the order further order will increase the coding cost. This is mainly due to the limited context which is insufficient to establish a good high order statistical model. The problem is even more profound for low bitrate coding applications, because the number of contexts available is extremely limited.

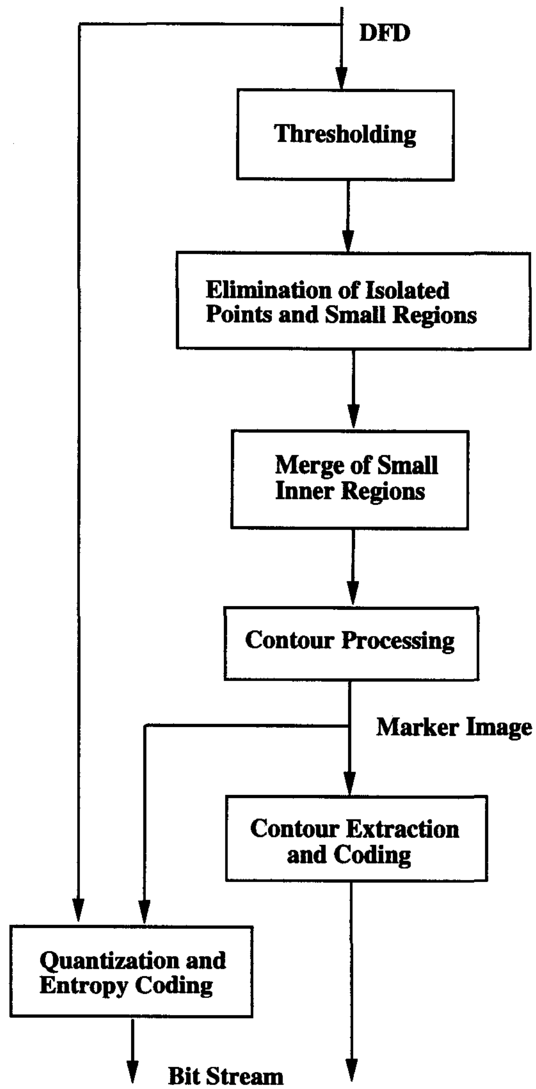
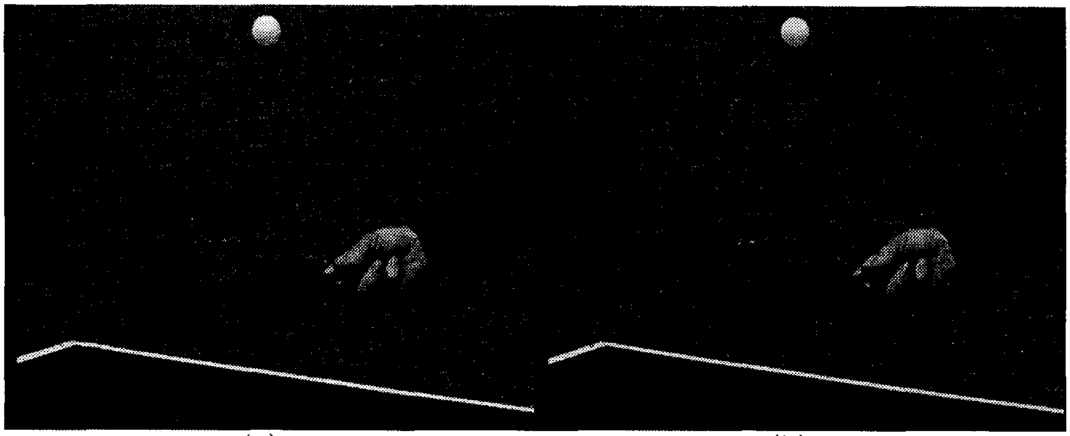
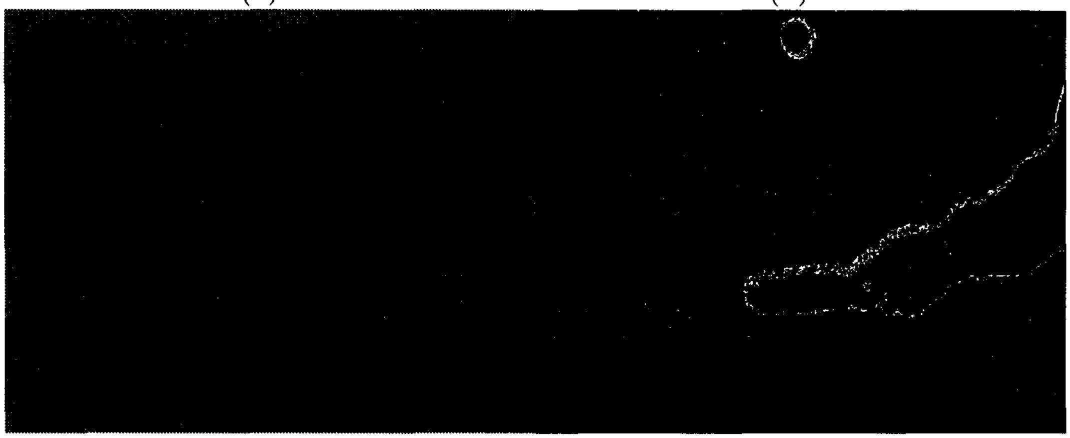


Figure 6: Major steps of the segmentation algorithm.



(a) (b)

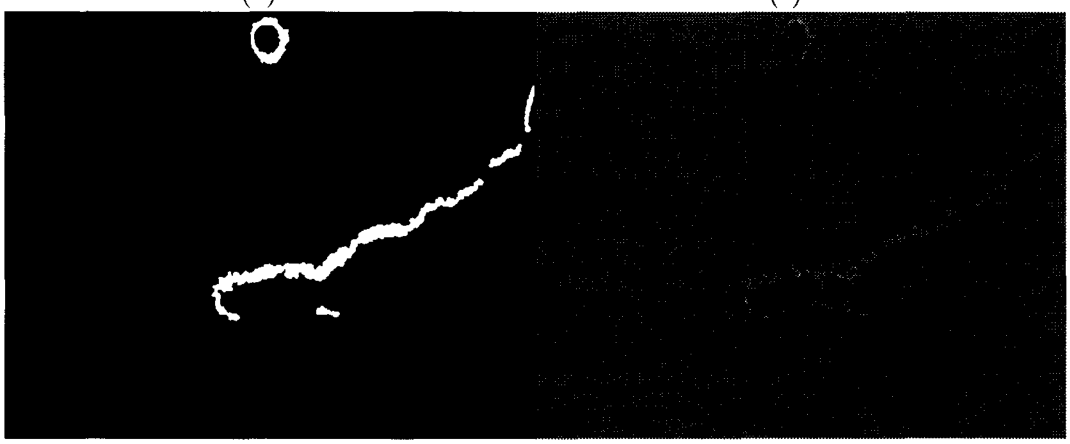


(c) (d)

frame	0^{th} order	1^{st} order	saving
2	22648	21864	3.5%
3	29960	27856	7.0%
4	11672	10504	10.0%
5	32840	28928	11.9%
6	88216	83864	4.9%
7	77296	71032	8.1%
8	13400	12208	8.9%
9	59064	50240	14.9%
10	35848	30024	16.2%
11	36744	30416	17.2%

Table 3: Improvement by using first order Markov model rather than the zeroth order model in the adaptive arithmetic coder. Results are computed using the first 12 frames of the sequence *Table Tennis* in CCIR 601 format. Coding cost unit is *bit*.

(e) (f)



(g) (h)

Figure 7: The major steps of segmentation: (a) Original frame, (b) Reconstructed frame, (c) Original DFD, (d) After thresholding, (e) After removing isolated points, (f) Marker image after elimination of small segments, (g) Final marker image, (h) Reconstructed DFD.

seq		without context	with context	saving
MA1	R_{dfd}	178120	165016	7.4%
	R_{mv}	102792	76600	25.5%
MA2	R_{dfd}	237288	223080	6.0%
	R_{mv}	62872	49248	21.7%
MA3	R_{dfd}	59952	51992	13.3%
	R_{mv}	29552	19120	35.3%
SM	R_{dfd}	689104	665968	3.4%
	R_{mv}	58528	44296	24.3%

Table 4: Using statistical model established by coding previous frame as an initial model to code the current frame improves the coding performance. This technique is valid for both DFD coding and motion vector coding. R_{dfd} is the number of bits to code all DFDs of a sequence, R_{mv} is the number of bits to code all motion vectors of a sequence. Coding cost unit is *bit*.

Establish A Good Initial Statistical Model

The adaptive arithmetic coder establishes the statistical model by assuming equi-probable distribution of each symbol and incrementally adapts the model (here is a table of joint probability between each two successive symbols) after coding each symbol. This method has the advantage that it can adapt quickly to the nonstationary statistics; however it requires actually coding some number of symbols so as to establish a good model. In the low bitrate coding applications, only a small percentage of pixels in DFD (say, 3%) will be coded in each frame. Many bits would therefore have to be used for the development of a good model. The modelling problem is significantly alleviated by the fact that that successive DFDs tend to have similar statistics; thus, one can actually use the statistical model established during the coding of a previous frame as an initial model for coding the current frame. This is equivalent to a statistical model with one frame-length memory. Statistical history beyond one frame duration is not considered. Table 4 shows the efficiency of this technique. As seen from this table, a small improvement in compression ratio results from using this limited context. A similar technique is used in the coding of motion vectors; In this case, the bit savings is much higher because two successive motion vector fields show much higher coherence than two successive DFDs.

One might tend to think that using more than one previous frame to establish an initial statistical model might be beneficial. In the low bitrate coding context, the quantization may change from one frame to another and furthermore, a scene change may make the model very inefficient. Thus, using only one previous frame is a good trade-off between flexibility of quantization and coding efficiency.

An area for further research is in determining how many symbols are required to establish a good model.

4 Codec Details

System Description

The challenge of low bitrate coding is the requirement for a very high compression ratio. Typically, CIF resolution (352×288) sequences with 30fps coded at 64 kbits/sec corresponds to a compression ratio of 390:1; QCIF resolution (176 $\times$ 144) sequences with 15 fps coded at 16 kbits/sec corresponds to a compression ratio of 190:1. To achieve such a high compression ratio, each block of the system should be well designed. For the low bitrate codec, we propose a hybrid motion-compensating coding system. Fig. 8 illustrates the schema of the system. A wavelet transform method is used in intraframe coding. For subsequent frames interframe coding is employed. Here, motion estimation is performed and the DFD is computed and the DFD is coded using the method described in previous section. To reduce the complexity of the input sequence, a preprocessing scheme is used. At the decoder, to reduce the visibility of the coding artifacts a simple post-processing scheme is employed. The following sections provide details on some of the modules in the proposed codec.

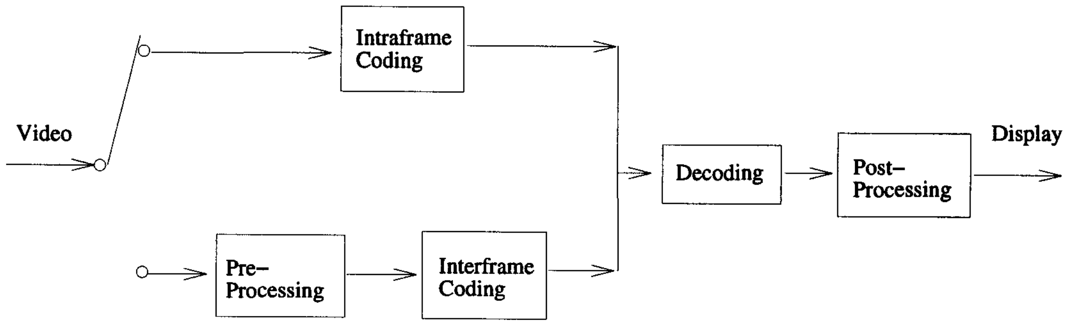


Figure 8: Schema of the codec.

Intraframe Coding

Fig. 9 shows the procedures used in intraframe coding. An image is decomposed using a three level wavelet transform; this transform partitions the frequency domain as shown in Fig. 9(b). Uniform quantization is applied for each level's transformed images, and the output of the quantizers is coded by an arithmetic coder with a first order Markov model. The wavelet used here is specially designed for image compression [6]. Even with simple quantization strategies, it gives better PSNR performance and leads to less objectionable artifacts than the current international still image coding standard JPEG, especially at high compression ratios. Fig. 10 illustrates the PSNR performance. Fig. 11 shows reconstructed images at a high compression ratio.

Pre-processing By Applying Intraframe Coding

Fig. 12 shows the interframe coding. A block matching full-search motion compensation algorithm is used. The motion vectors are computed usually between the current original frame and the previous reconstructed frame. As the previous reconstructed frame contains considerable quantization error, the motion estimation procedure results in a very noisy motion field (which is expensive to code) and can cause the still background to move unnaturally. To avoid this problem, the intraframe coding/decoding method is applied to each interframe prior to motion compensation so that the current

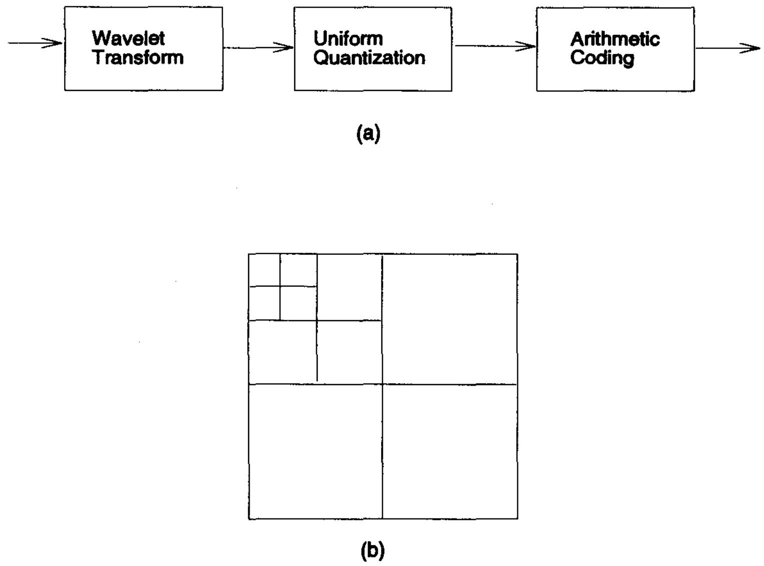


Figure 9: Intraframe coding by wavelet transform. (a) block diagram; (b) frequency partition.

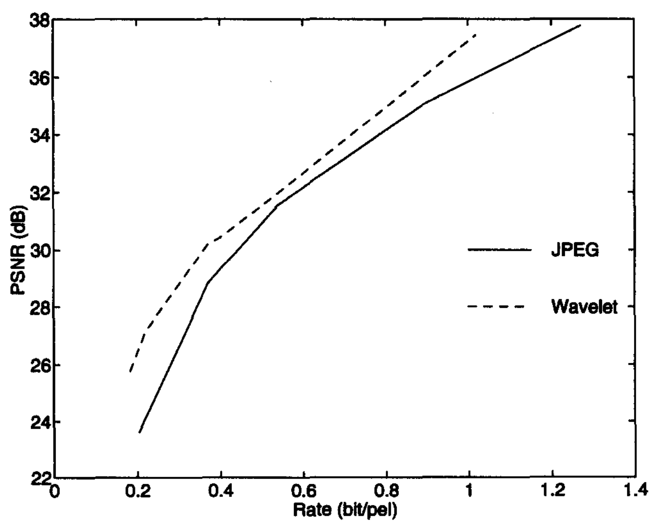


Figure 10: PSNR performance comparison between JPEG and the proposed method.

Figure 11: Reconstructed images. (a) JPEG coded at 0.20 bit/pel; (b) Wavelet-based coded at 0.18 bit/pel.

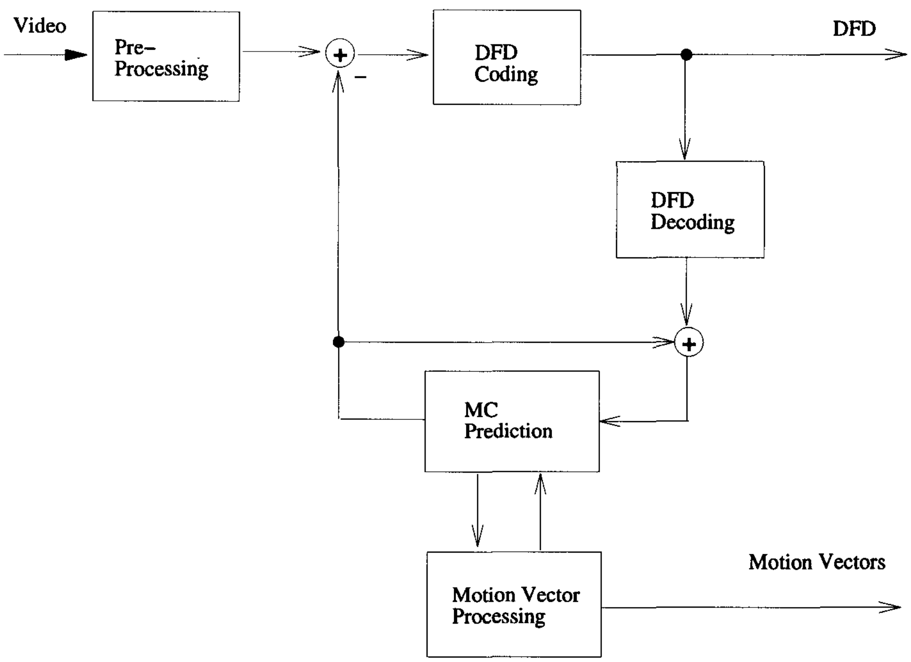


Figure 12: Block diagram of interframe coding.

frame contains the same amount of quantization noise for the still background. The effectiveness of this pre-processing is shown in Fig. 13.

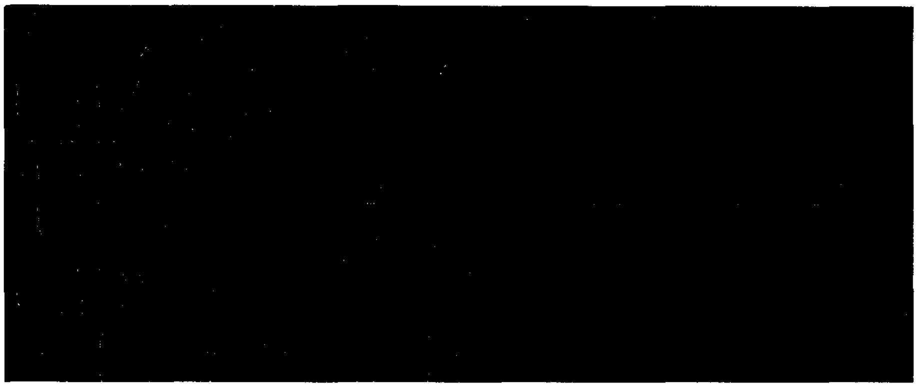


Figure 13: Pre-processing improves the uniformity of motion vectors. (a) without pre-processing; (b) with pre-processing. Motion vectors are more uniform in the latter case.

Motion Vector Processing

In Fig. 13(b), many of the motion vectors corresponding to the background have non-zero values, which are expensive to code. A motion vector processing procedure is developed to set the motion vectors corresponding to the still background or regions with low contrast to zero. The procedure can be stated formally as follows:

1. compute the direct difference between two frames (assuming motion is zero) for each block;
2. compute its mean m and variance var;
3. if $m < T_m$, and $var < T_{var}$, where T_m and T_{var} are preset thresholds, then set motion vectors corresponding to the block to zero.

Note that it is important to check both the mean and variance. For example, if one bright uniform object is moving in front of a dark uniform background, then the motion vector should not be set to zero.

Fig. 14 shows the efficiency of this procedure. It reduces significantly the number of bits needed to code the motion vectors with little overhead on DFD coding.

Post-processing

The segmentation based DFD coding quantizes spatial domain samples; hence, artificial contours can occur in reconstructed sequences. The image may appear "dirty" and patches might be visible in smooth regions. Lowpass filtering can reduce these artifacts but will blur the image. An edge-preserving lowpass filter is devised as a post-processing device. It is basically an iterated lowpass filter, as shown in Fig. 15. Repeated application of this lowpass filter results in a smooth image;however the edges will be preserved. The edge preserving lowpass filtering process can be described as follows:

(1) Consider an image block of 3×3 (Fig. 16(a)), compute its variance var. Let the quantization step size used in coding a DFD be Δ_d, then the quantization noise will have a variance approximately

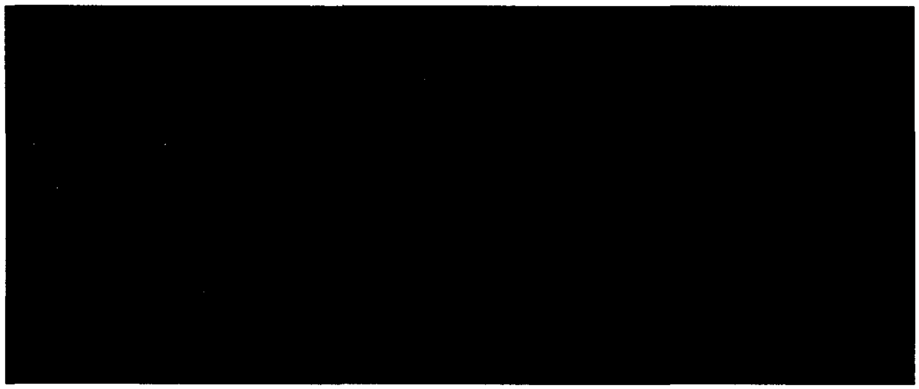


Figure 14: Pre-processing improves the uniformity of motion vectors. (a) before motion vector processing; (b) after motion vector processing.

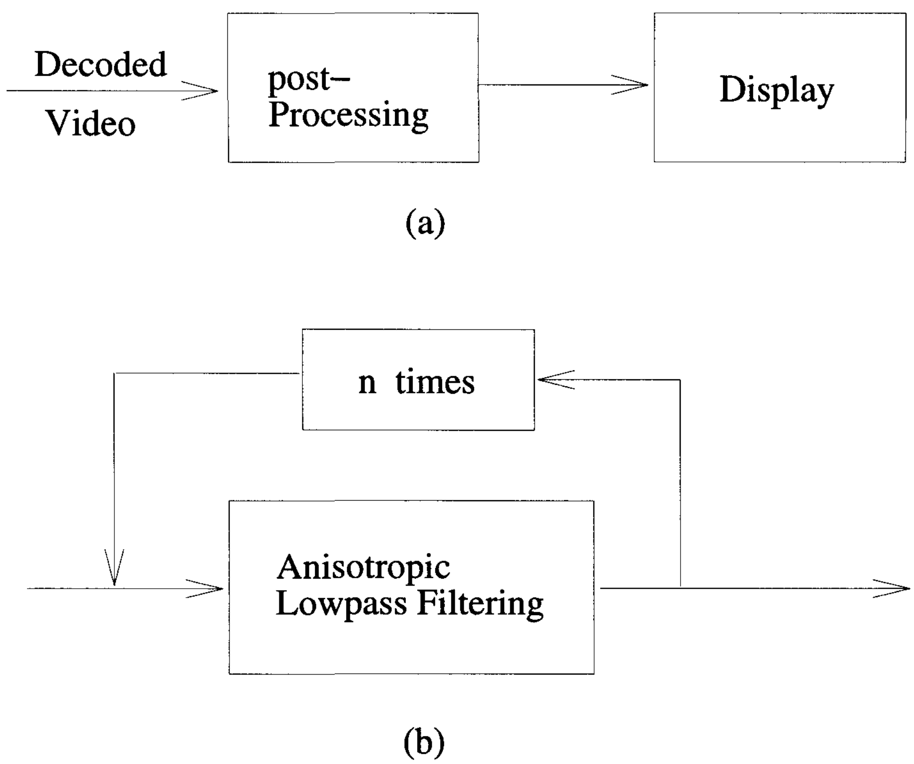


Figure 15: Post-processing by edge-preserving lowpass filters.

equal to $\Delta_d^2/12$. If $var < \Delta_d^2/12$, then the variance is most probably caused by quantization and hence we perform a 2-D separable lowpass filter with the kernel 1/4[1,2,1], otherwise, go to step (2).

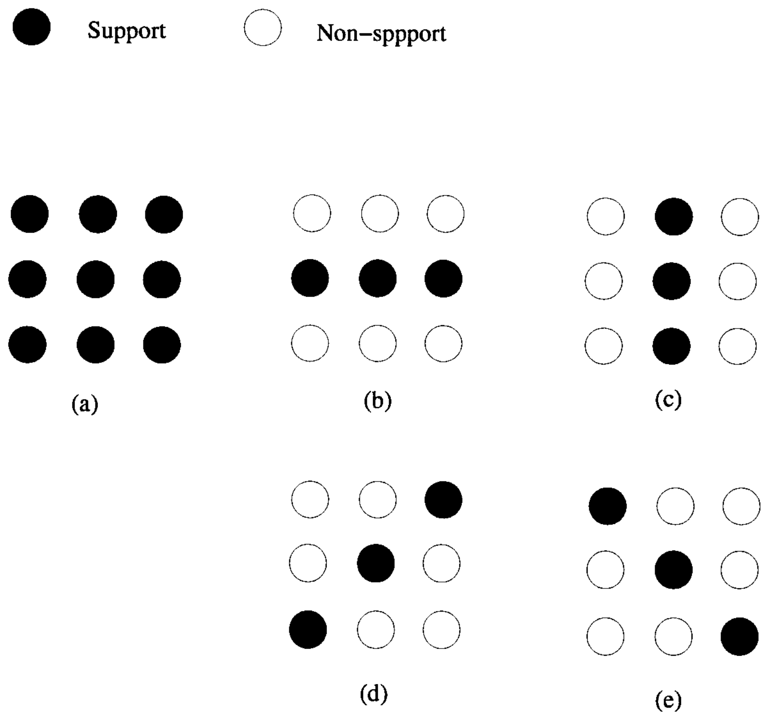


Figure 16: Support region of the edge-preserving lowpass filters.

(2) Compute the variances on the one dimensional support region along 4 different directions (Fig. 16(b),(c),(d),(e)). Perform 1-D lowpass filtering along the direction with least variance using the same kernel 1/4[1,2,1]. In this way, lowpass filtering is performed only along the edge but not across it.

Fig. 17 shows an example of the post-processing.

Figure 17: An example of the post-processing, number of iterations equals to 2. (a) before post-processing; (b) after post-processing.

parameters	MA1	MA2	MA3	SM
Δ_1	30	30	30	20
Δ_2	9	9	11	6
Δ_3	3	3	5	2
Δ_4	4	4	4	3
Δ_d	12	8	17	10
β	3.5	4.5	1.8	4.0
T_m	15	15	10	15
T_{var}	50	50	40	50

Table 5: Parameters used for each simulation.

5 Coding Performance

The coding system was tested using 4 sequences MA1, MA2, MA3 and SM.

- MA1 is 150 frames of *Miss America* test sequence at 30 fps and at CIF resolution.
- MA2 is 75 frames of *Miss America* test sequence at 15 fps and at CIF resolution. MA2 is obtained by temporal subsampling of MA1.
- MA3 is 75 frames of *Miss America* test sequence at 15 fps and at QCIF resolution. It is obtained by spatial subsampling of MA2.
- SM is 75 frames of *Sales Man* test sequence at 15 fps and at CIF resolution.

The proposed coding system involves the following parameters:

- Δ_1, Δ_2, Δ_3 – quantization stepsize for each level of wavelet transform in intraframe coding. The stepsizes are used in order of high to low resolution levels.
- Δ_4 – quantization stepsize for the lowest resolution (DC) band in intraframe coding.
- Δ_d – quantization stepsize for DFD coding.
- β – percentage of the pixels kept in DFD segmentation.
- T_m and T_{var} – thresholds for motion vector processing.

Table 5 shows parameter values used in each simulation. Although they are not fully optimized, the choice reflects some insights as to how the system should be tuned to obtain a certain target bitrate.

Figs. 18, 19, 20 show simulation results of MA1 coded at 52.7 kbit/sec.

Fig. 18 shows the number of bits to code DFD and motion vectors for each frame. It can be observed that motion vectors are less expensive to code than the DFDs due to motion-vector processing. The number of bits to code DFDs fluctuates from one frame to another. This is a characteristic of the proposed segmentation algorithm. If the high energy part is small, then the algorithm will delay its coding to later frames where the high energy region becomes large enough.

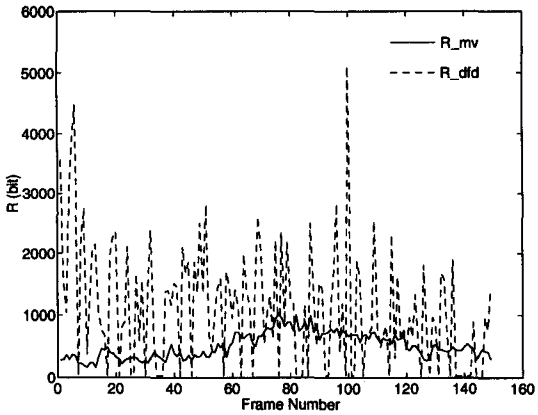


Figure 18: MA1: number of bits for DFDs and motion vectors as a function of frame number.

Fig. 19 shows the total number of bits for each frame. The intraframe requires more bits to code compared with coding the DFDs; however, intraframe coding is employed only once every few seconds.

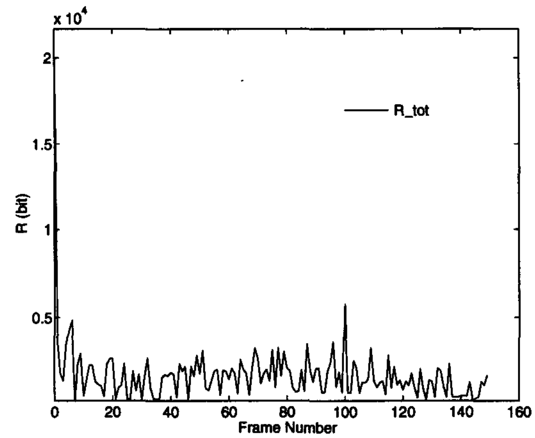


Figure 19: MA1: total number of bits for each frame.

Fig. 20 depicts the PSNR value as a function of frame number.

Similar results for simulations on MA2, MA3, and SM are reported in Figs. 21, 22, 23, 24, 25, 26, 27, 28, 29.

Table 6 summarizes the simulation results. *Miss America* with CIF resolution at 30 fps can be coded at around 60 kbits/sec, and *Miss America* with QCIF resolution can be coded at around 16 kbits/sec. *Sales Man* is much more difficult to code because of large amount of motion and complex uncovered background. These results are for coding the Y component only. If the coding procedure as described in this report is also applied on the U and V components, the bitrates reported here would increase by atmost 50 percent; with a vector coding approach, the bitrate increase is expected to be less than 50 percent. These results should be compared with the H.261 test model; presently

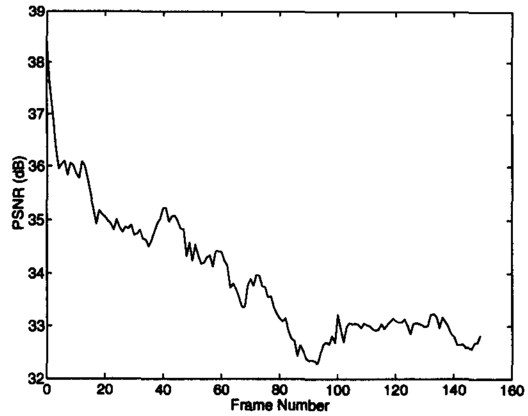


Figure 20: MA1: PSNR as a function of frame number.

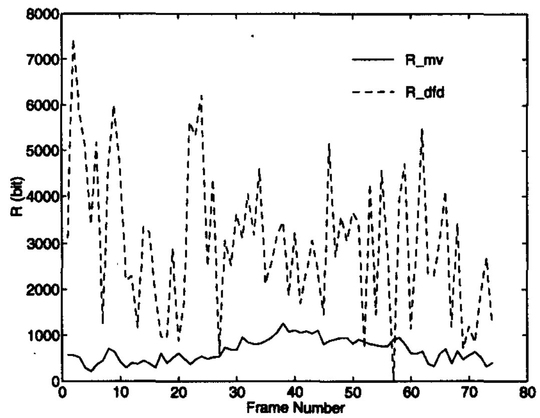


Figure 21: MA2: number of bits for DFDs and motion vectors as a function of frame number.

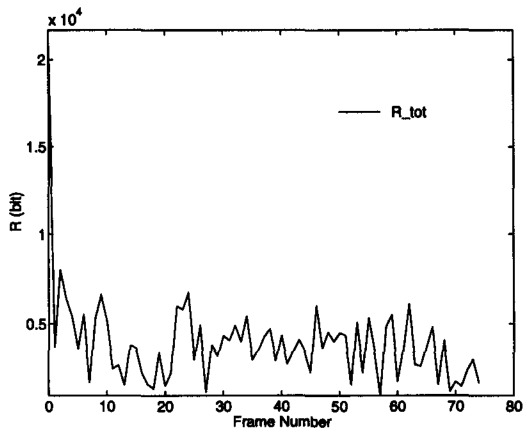


Figure 22: MA2: total number of bits for each frame.

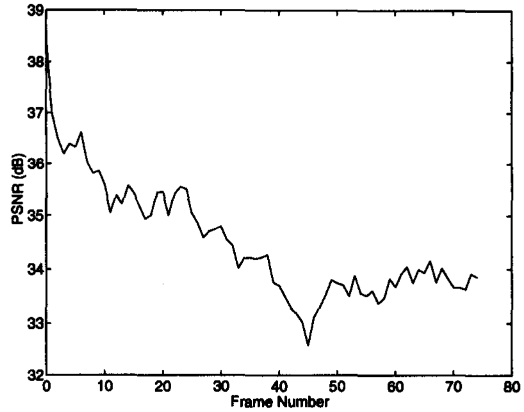


Figure 23: MA2: PSNR as a function of frame number.

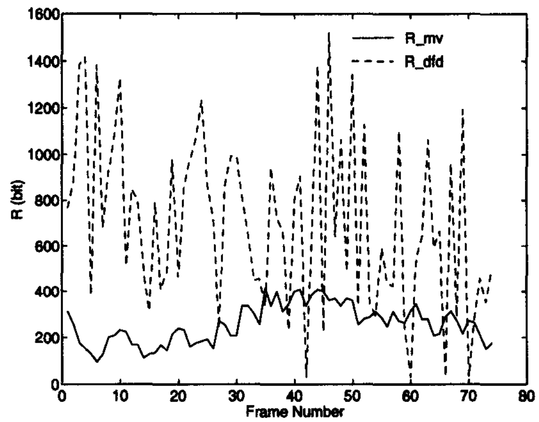


Figure 24: MA3: number of bits for DFDs and motion vectors as a function of frame number.

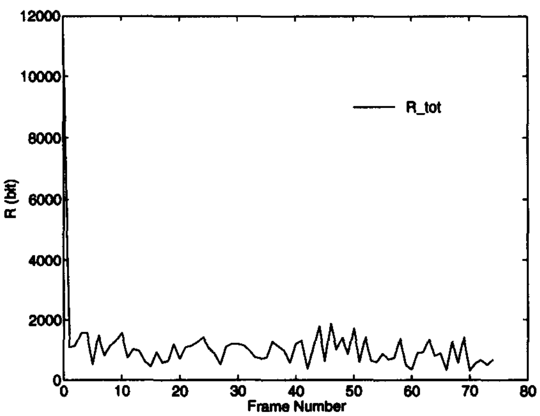


Figure 25: MA3: total number of bits for each frame.

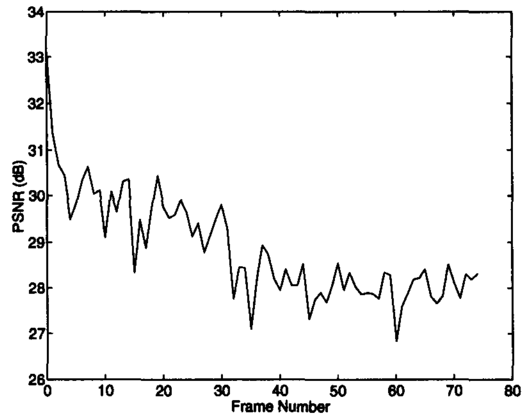


Figure 26: MA3: PSNR as a function of frame number.

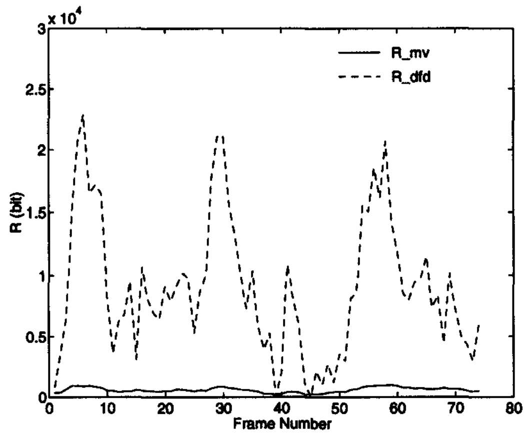


Figure 27: SM: number of bits for DFDs and motion vectors as a function of frame number.

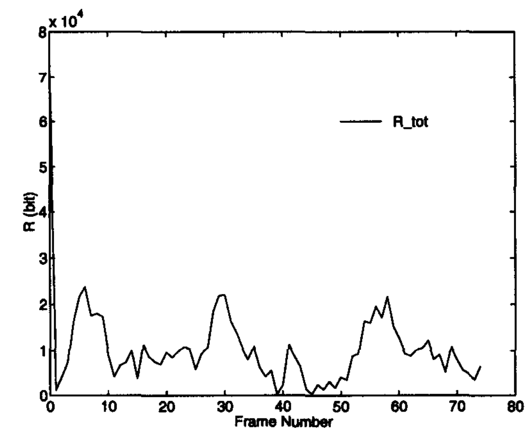


Figure 28: SM: total number of bits for each frame.

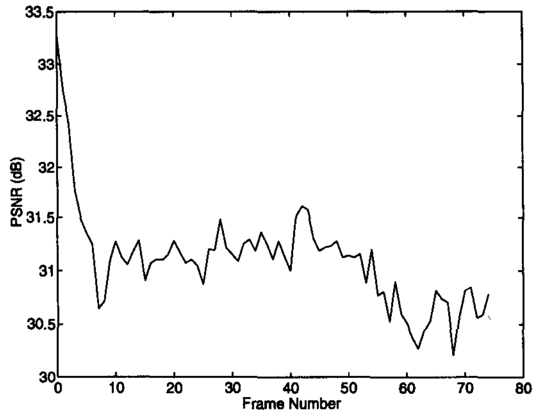


Figure 29: SM: PSNR as a function of frame number.

sequence	MA1	MA2	MA3	SM
N	150	75	75	75
resolution	CIF	CIF	QCIF	CIF
frame rate (f/s)	30	15	15	15
R_b(bit)	21640	21640	11216	74008
R_{dfd}(bit)	165016	223080	51992	665968
R_{mv}(bit)	76600	49248	19120	44296
R_{tot}(bit)	263256	293968	82328	784272
bitrate(kb/s)	52.7	58.8	16.5	156.9

Table 6: Simulation results on various test sequences

the H.261 codec software we posess does not operate at bitrates below 256 kbits/sec.

The final judgment of the quality of a codec is of course the visual quality of the reconstructed sequences. A video tape has been generated to demonstrate the results; this tape includes the original sequence, the reconstructed sequence, motion vectors, and segmentation of DFDs.

6 Conclusions

A low bitrate codec has been developed. It uses wavelet transform for intraframe coding. A segmentation based method is used for interframe coding. This method does not use a DCT unlike the DCT based approaches adopted in the H.261, MPEG1 and MPEG2 video compression standards. A pre-processing scheme is applied to the input video sequence so as to yield an uniform motion-vector field. The motion-vectors are further processed so as to reduce their contribution to the aggregate bitrate. An arithmetic coder is used to code the segmented DFDs and the motion vectors. During the reconstruction phase in the decoder, edge preserving post-processing is used to remove some of the coding artifacts. Fairly good visual quality can be obtained at 64 kbits/sec for CIF resolution sequence and at 16 kbits/sec for QCIF resolution sequence.

A coding system is a sophisticated system with many parameters involved. Optimizing these parameters at the system level is not an easy task. Further research will be directed at this avenue.

The current coding system is a variable rate one. Efforts should be made to develop a good buffer control mechanism and an adaptive quantization strategy so that constant output bit bitrate can be obtained with maximally uniform visual quality. After completion of the optimized coder based on the approach proposed in this report, further work needs to be done in rigorously comparing the performance of this optimized coder with schemes that might be variants of H.261.

Acknowledgment

The research reported in this document was completed during the first author's summer training phase at Hewlett-Packard Laboratories. The authors are grateful to Dr. Daniel Pitt at HP Labs and Dr. Murat Kunt at EPFL for offering this unique opportunity. Special thanks go to Sam Liu, for his outstanding perspectives on low bitrate coding. The results would not have been possible without Konstantinos Konstantinides support on computer environment.

The developed coding system involves several software modules, some of which are obtained by courtesy of the following persons: Olivier Egger for arithmetic coder, Fédéric Dufaux for motion compensation, Sam Liu for interpolation.

References

[1] M. Gilge, T. Engelhardt, and R. Mehlan. Coding of arbitrarily shaped image segments based on a generalized orthogonal transform. *Signal Processing: Image Communication*, 1:153–180, 1989.

[2] W. Li and F.X. Mateo. Segmentation based coding of motion compensated prediction error images. In *ICASSP'93*, volume V, pages 357–360, 1993.

[3] W. Li and M. Kunt. Morphological segmentation applied to displaced frame difference coding. *Special Issue of Signal Processing on Mathematical Morphology and its Applications on Signal Processing, Accepted for publication.*

[4] I.H. Witten, R.M. Neal, and J.G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30:520–540, June 1987.

[5] Mark Nelson. *The Data Compression Book.*

[6] W. Li and O. Egger. Improved subband coding of images using unequal length pr filters. In *Proceedings of Gretsi'93*, September 1993.