FACTA UNIVERSITATIS (NIŠ)
SERIES: ELECTRONICS AND ENERGETICS VOL. 9, NO. 2 (1996), 173–184

SOME IMPROVEMENTS OF THE IMAGE SEQUENCE COMPRESSION

Dušan Ćurapov and Miodrag Popović

Abstract. In this paper, some improvements of the standard method for image sequence compression using Discrete cosine transform and motion compensation are described. The main improvements are realized by dividing the images into blocks of various size and by tuning the quantization matrices according to image activity. The realized signal to noise ratio was 37 to 39 dB at the bit rate of 0.3 bpp, what represents improvement of 2 to 4 dB over similar DCT–based methods.

1. Introduction

The recent advances in communication and computer systems have made possible the transmission of images at low bit rates, using standard telephone or ISDN channels. In order to transmit TV signal through such links, its spatial and temporal redundancies should be made significantly smaller. To reduce the spatial redundancy the Discrete cosine transform is usually used, because of its ability to compress the signal into a small number of DCT coefficients. Further, the DCT is a real transform, and fast algorithms for its computation exist. In order to realize compression in real time using modern VLSI integrated circuits for DCT computation, the image is usually divided into blocks of 8×8 pixels [7].

The simplest method for the removal of temporal redundancy is the coding of the difference between successive images from the sequence. The more elaborate methods extract some information about the motion and use it to compensate the motion before the difference and coding operations.

Manuscript received January 27, 1995.

A version of this paper was presented at the second Conference Telecommunications in Modern Satellite and Cables Services, TELSIKS'95, October 1995, Niš, Yugoslavia..

The authors are with Faculty of Electrical Engineering, University of Belgrade, Bulevar Revolucije 73, P.O. Box 816, 11001 Belgrade, Yugoslavia.

In order to facilitate the transmission and storage of video signals, several international standards are proposed, such as H.261 [1] and MPEG [3]. In these standards, the DCT is used to remove the spatial redundancy, and motion compensation is optionally used to remove the temporal redundancy. However, no specific motion compensation method is specified in these standards, only the position of motion vector in the data stream and its size are specified.

The simplified block diagrams of coder and decoder in both systems are very similar, and they are presented in Fig. 1.

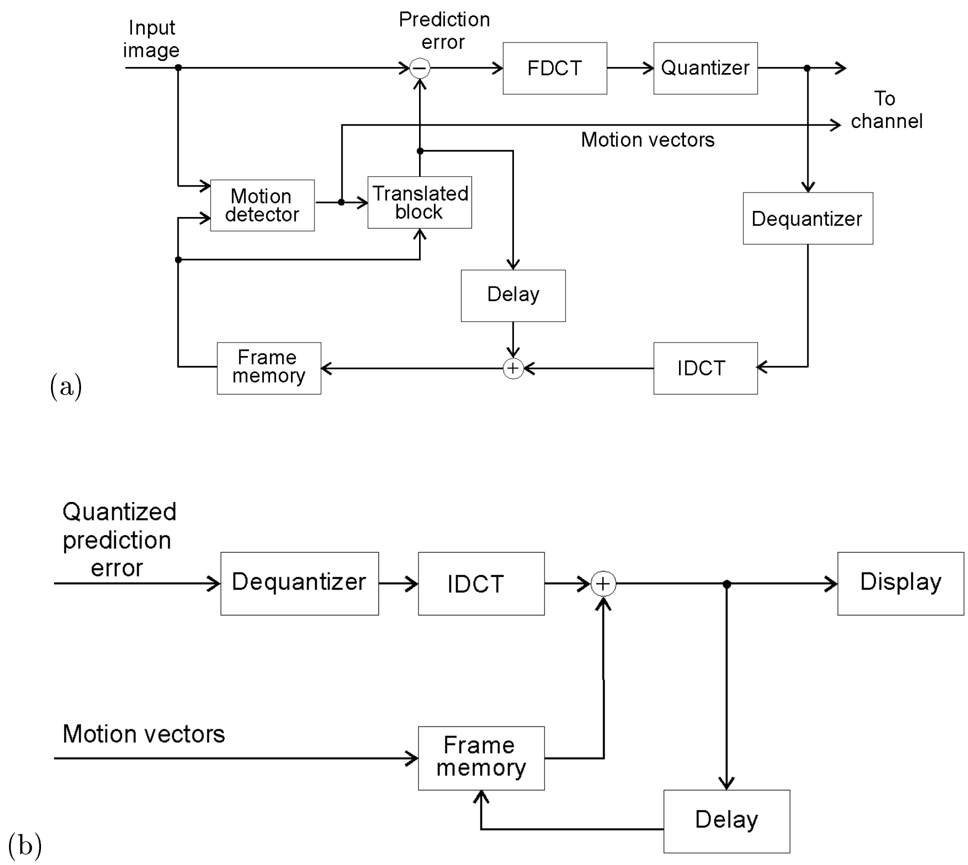

Figure 1. The block diagram of the motion compensated video signal transmission: (a) coder, (b) decoder.

In the last few years, several solutions for motion compensation in image sequences were proposed [6]. It has been shown that motion compensation can significantly improve visual quality of images at the fixed bit rate, or permit the transmission at lower bit rate at the fixed image quality. However, no method appears to be the best solution.

In this paper an experimental analysis is performed to establish what can be obtained if images are divided into blocks of variable sizes (instead of fixed size 8×8 pixels) in the process of transformation and coding. The results are evaluated using signal to noise ratio as the objective criterion, and also by a subjective evaluation. To facilitate the comparison with other results, the sequence known as *Miss America* was used in our experiments.

2. Motion compensation

The motion compensation is a very important part of any algorithm for the compression of image sequences. Every algorithm for the motion compensation consists of two parts: the *estimation* of the motion, and the *compensation* of the estimated motion. In order to realize efficient motion estimation and compensation, images are usually divided into small blocks, and the motion of the blocks is estimated giving a set of *motion vectors*. The quality of the motion estimation algorithm is dependent on its computational complexity and accuracy of the estimated motion vectors.

In the usual videoconferencing sequences, the motions of the blocks between two successive images are only few pixels. Because of that, the motion estimation of a block $n \times n$ from the current image, reduces to the determination of the best match of that block and blocks of the same size in the previous image which lay in the *search region* $(n + 2p) \times (n + 2p)$, where $\pm p$ is the maximal tolerable motion. The most accurate, but the least effective, motion estimation method is the block matching method based on the criterion of the maximal cross-correlation (or minimal average absolute error) between blocks from the two images. In both cases, it is necessary to examine $(2p + 1)^2$ possible matches, what represents very high computational complexity. Because of that, in the last few years several more effective methods for the motion estimation were proposed [6], that improve computational complexity up to ten times. In all these methods only the local optimum is found, but this solution is acceptable in this application.

One of the best algorithms for the motion estimation is the *three-step algorithm* [5]. In the first step of this algorithm, the mean absolute error (MAE) is calculated in nine points with coordinates $(i \pm p_1, j \pm p_1)$, $p_1 = 0$ or 3, based on formula [4]:

$$MAE(i,j) = \frac{1}{n^2} \sum_{u=1}^{n} \sum_{v=1}^{n} \mid f_k(u,v) - f_{k-1}(u+i, v+j) \mid \tag{1}$$

These points are denoted by 1 in Fig. 2. From these nine points, the point with minimum MAE is chosen, which represent the first approximation of the motion vector. In Fig. 1 this is the point $(i+3, j+3)$. In the second step, the MAE is calculated in new eight points (denoted by 2 in Fig. 2), using finer resolution $p_2 < p_1$, and the new approximation of the motion vector is determined. This is the point $(i+3, j+5)$ in Fig. 2. The second step can be repeated several times, every time with finer resolution $p_i < p_{i-1}$. In the last step, the resolution is $p_i = 1$. When the size of the search region is $p \leq 6$, only three steps are needed. The final position of the motion vector in Fig. 2 is $(i+2, j+6)$. This method is also suitable for hardware realizations in VLSI.

In this paper, in order to obtain higher compression, several experiments using variable block size will be performed. It is known that blocks of greater size can be coded using smaller number of bits per pixel (bpp), but their use is not suitable if there is a large motion between images. That is the reason why in standard coding methods the block size of 8×8 pixels is used. In our experiment, in the first phase of the motion estimation, the block size of 16×16 pixels was used. Each block is characterized by two parameters: the motion vector and the MAE. The value of MAE determines the activity of the block, e.g. it represents the amount of motion in the block that can not be compensated. Using the MAE, all blocks are classified into three classes, so that class I contains the blocks with the highest activity. Then, in the second phase of the algorithm, four neighboring blocks are examined. If at least three blocks belong to class III and only one to class II, then these four blocks are merged into a larger block of size 32×32, with motion vector equal to the arithmetic mean of four motion vectors. If a block belongs to class I, it will be divided into four smaller blocks of size 8×8. The motion vectors of these new blocks should be determined again, but using the existing motion vector as a good initial guess to reduce the computation. The remaining blocks, which are not grouped nor divided, retain the initial size of 16×16 pixels. Using this procedure, only the blocks that contain large motion will be coded using block size of 8×8 pixels.

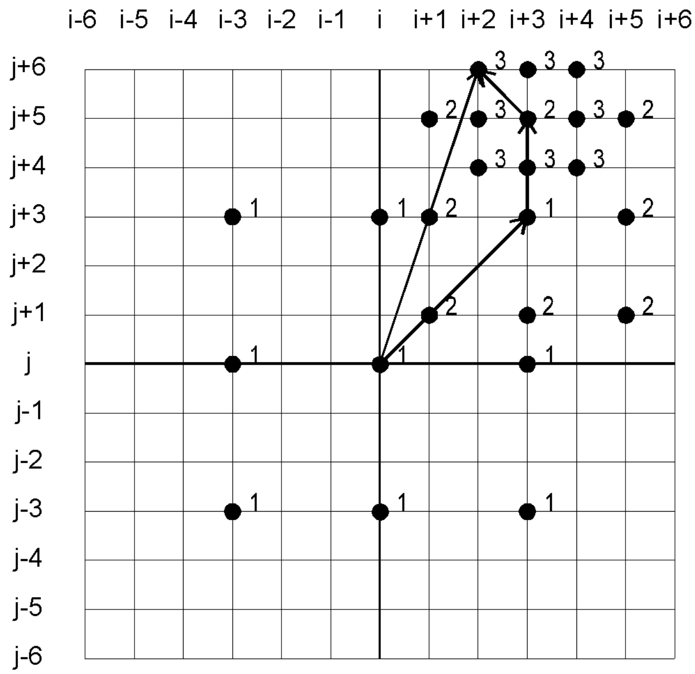

Figure 2. The three–step algorithm for motion estimation.

3. Discrete cosine transform

When motion vectors of all blocks of an image are determined, the differences between blocks from the current image and corresponding motion compensated blocks from the previous image are formed. The difference blocks are then transformed using 2–D Discrete cosine transform (DCT) using the expression:

$$F(u,v) = \frac{c_u c_v}{\sqrt{2n}} \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} f(x,y) \cos \frac{(2x+1)u\pi}{2n} \cos \frac{(2y+1)v\pi}{2n} \tag{2}$$

where:

$$c_u, c_v = \begin{cases} \frac{1}{\sqrt{2}} & u,v = 0 \\ 1 & u,v \neq 0 \end{cases} \tag{3}$$

and $n = 8, 16$ or 32.

The values of DCT coefficients $F(u,v)$ are very different. It has been found by a detailed examination that the values of all DCT coefficients have

Laplacian probability density with mean value equal to zero. Therefore, the expression for Laplacian probability density

$$p(x) = \frac{\alpha}{2} e^{-\alpha|x-\mu|} \tag{4}$$

with $\mu = 0$, can be used to represent the probability density of the DCT coefficient of difference blocks. Also, the variance of DCT coefficients is given by:

$$\sigma^2 = \frac{2}{\alpha} \tag{5}$$

4. Design of the quantizers

Based on the variances of the DCT coefficients, the bit allocation matrices which show the allocation of the available bits to the quantizers can be determined. In the available literature, two types of bit allocation algorithms are described. In the first type, the fixed bit allocation pattern is used, which is determined from the image statistics. In the second type, the DCT coefficients are divided by the elements of the predetermined quantization matrix and subsequently quantized and coded. In this case, some overhead information about the position and size of DCT coefficients must be transmitted. The first type is considered simpler but less effective.

In this paper, an integer fixed bit allocation algorithm is used, which is described in more detail in [4], [2]. In this algorithm, B represents the desired average bit rate, and $N = n^2$ is the number of DCT coefficients in a block.

1. Initialization of the algorithm:

$$n_k^0 = 0,\ 0 \leq k \leq N-1,\ j = 1 \tag{6}$$

2. Calculate

$$n_k^j = n_k^{j-1} + \delta(k-i) \tag{7}$$

where i is the value of the index k for which the expression

$$\Delta_k^j = \sigma_k^2 \left[d(n_k^{j-1}) - d(n_k^{j-1}+1) \right] \tag{8}$$

reaches its maximum. The Δ_k^j represents the reduction in distortion if the j-th bit is allocated to the k-th DCT coefficient, and $d(m)$ represents the mean square distortion of the m-bit quantizer for the unity variance input.

3. The process of bit allocation is finished if the inequality

$$\sum_{k=0}^{N-1} n_k^j \geq NB \tag{9}$$

is satisfied. If it is not satisfied, put $j \to j+1$, and repeat the second step of the algorithm.

The second step in this algorithm need some additional comments. In the most common case of Lloyd–Max optimum quantizer, the mean square distortion function $d(m)$ has the form:

$$d(m) = a2^{-bm} \tag{10}$$

where the parameters a and b depend on the type of the distribution and the range of variable m. Their values can be determined from Table 4.4 in Ref. [4]. In this case eqn. (8) reduces to:

$$\Delta_k^j = (1 - 2^{-b})\sigma_k^2 d(n_k^{j-1}) \tag{11}$$

that is much easier to compute.

Another problem may occur in step 2 when two or more indices k give equal Δ_k^j. In that case an additional bit is allocated to each quantizer.

Using this algorithm, and starting from the prescribed bit rates for block size 8×8, B_8, 16×16, B_{16}, and 32×32, B_{32}, the quantization matrices can be formed. But, it has been experimentally found that for block size 8×8 the distribution of DCT coefficients can be very different. Hence, in this case, it is useful to propose several quantization matrices. The decision, what matrix should be used, is made based on the value of the AC activity measure, defined as:

$$ACT = \frac{1}{63}\left[\left(\sum_{k=0}^{7}\sum_{l=0}^{7} F^2(k,l)\right) - F^2(0,0)\right]^{1/2} \tag{12}$$

In this work five types of 8×8 quantization matrices are used, shown in Fig. 3a. The quantization matrices for block sizes 16×16 and 32×32 are shown in Fig. 3b and Fig. 3c. Taking into account the frequency of the appearance of various blocks, and its bit rates $B_8 = 0.5$ bpp, $B_{16} = 0.25$ bpp, and $B_{32} = 0.125$ bpp, the average bit rate is about 0.3 bpp.

5. Computer simulation

In order to assess the performance of the proposed modifications using the objective and subjective criterions, some simulations were performed on

2	1	1	1	1	0	0	0
1	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	0	0	1	0

3	1	1	1	1	0	0	0
2	1	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	1	0	0	0	0	1	0
1	1	0	0	0	0	0	0
1	1	1	1	1	1	2	1

3	2	2	1	1	0	0	1
2	2	1	1	0	0	0	0
2	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	1	0	0	0	0	1	0
1	0	0	0	0	0	0	0
1	1	0	0	0	0	1	1

3	2	2	1	1	1	0	0
3	2	1	1	1	0	0	0
2	2	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0

4	3	2	2	1	1	0	0
3	2	2	1	1	0	0	0
2	2	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(a)

2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	2	3	2	0

(b)

2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

(c)

Figure 3. The quantization matrices for the sequence *Miss America*: (a) 8×8 ($B_8 = 0.5$ bpp), (b) 16×16 ($B_{16} = 0.25$ bpp), (c) 32×32 ($B_{32} = 0.125$ bpp).

a computer system consisting of a PC 486/DX2 computer, the video subsystem Imaging Technology Series 151, and a high quality video display. All simulation programs that simulate the operations in coder and decoder were written in Microsoft C language. However, there are some differences between the simulations and a real transmission system. The DCT and its inverse were calculated using floating point instead of fixed point arithmetic. Also, the entropy coding and error correcting coding were omitted from simulations, since the main goal was to determine the influence of the block size on the compression due to quantization of DCT coefficients. Since other experiments, using fixed size blocks, were conducted using the same experimental setup, the obtained results can be used for comparison with other methods. The system was tested using several known videoconferencing sequences.

In order to evaluate the quality of the reconstructed sequence, the peak signal to noise ratio ($PSNR$), defined as:

$$PSNR = 10 \log_{10} \frac{255^2}{\frac{1}{N_{TOT}} \sum_{i=1}^{N_{TOT}} (f_i - \hat{f}_i)^2} \tag{13}$$

was used as the objective criterion. In this expression, N_{TOT} represents the total number of pixels in an image, f_i represent the value of a pixel from original image, and $\hat{f}_i$ represent the value of the same pixel from the reconstructed image. In Fig. 4 are shown the results of simulations for the first 30 frames from the sequence *Miss America*, at the bit rate $B = 0.3$ bpp. To facilitate the simulation, the first frame in the reconstructed sequence was taken from the original sequence.

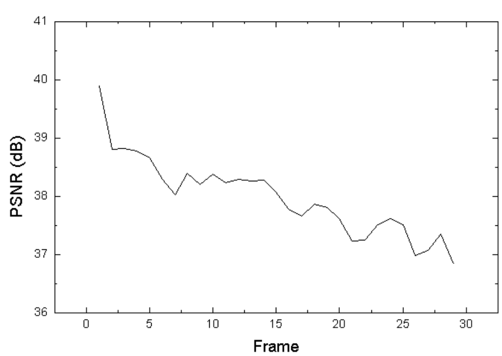

Figure 4. The peak signal–to–noise ratio for the first 30 frames from the sequence *Miss America* at $B = 0.3$ bpp.

It is easily seen in Fig. 4 that the realized *PSNR* varies between 37 dB and 39 dB. This result compares very well with the results known from the literature. For example, in our previous simulations of DCT-based methods using fixed size blocks of 8×8 pixels [2], the realized *PSNR* was smaller for 2–4 dB.

(a)

(b)

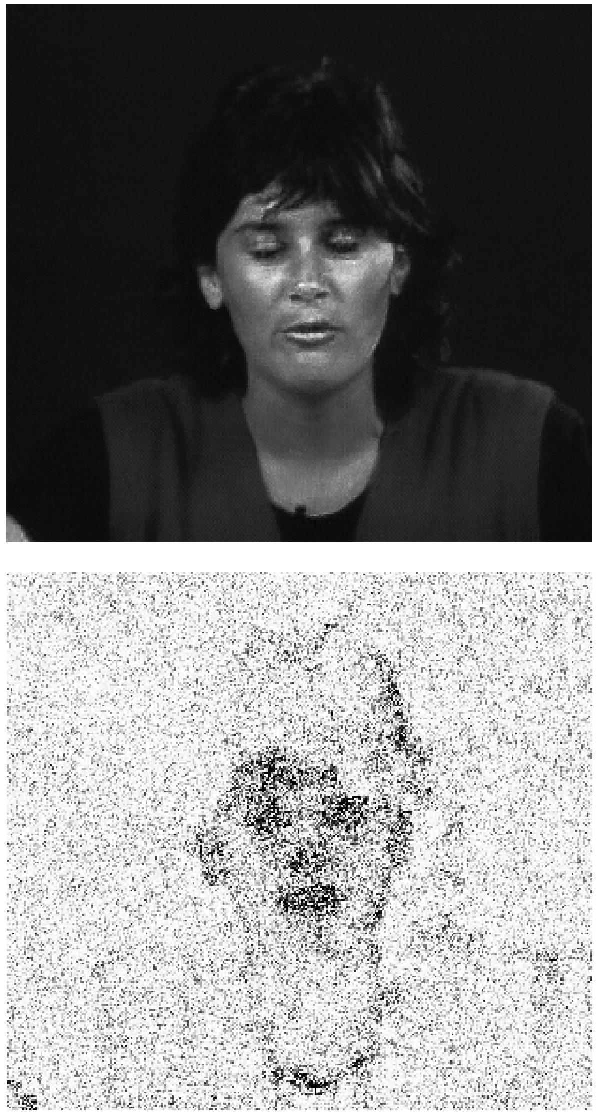

Figure 5. (a) The reconstructed 16th frame from the sequence *Miss America*,
(b) The negative of the difference between original
and reconstructed 16-th frame multiplied by 20.

The subjective evaluation of the quality of the reconstructed sequence was performed by observing the images on a high quality large video display. Only at the bit rates smaller than 0.5 bpp some small distortions can be noticed in regions having high motion (lips, eyes, neck, and part of the hair) as can be seen in Fig. 5b. At the higher bit rates such distortions can not be noticed.

6. Conclusion

In this paper a new system for the compression of image sequences, suitable for videoconferencing systems, is described. The main improvements over standard systems lays in the dividing the images into blocks of variable sizes (8×8, 16×16 or 32×32 pixels), and coding of blocks according to the statistical properties of their DCT coefficients. It has been found that DCT coefficients of differences between motion compensated blocks from current and previous images have Laplace distribution. Based on this fact, a set of quantization matrices and corresponding optimal quantizers were designed. The second improvement is in the use of several quantization matrices for the smallest blocks of 8×8 pixels, according to their activity. The proposed modifications were tested using first 30 frames from the videoconferencing sequence *Miss America*, and it has been found that the peak signal to noise ratio (PSNR) varies between 37 dB and 39 dB, at the bit rate of 0.3 bpp. This represents an improvement of 2 dB to 4 dB over existing DCT-based methods using fixed size blocks. The subjective quality of the image sequence was evaluated by viewing the sequence on a high quality display. Some small distortions can be noticed in regions having high motion only at the bit rates smaller than 0.5 bpp.

REFERENCES

1. CCITT Recommendation H.261: *Video Codec for Audiovisual Services at* $p \times 64$ *Kbit/s*. Dec. 1990.
2. D. Ćurapov and M. Popović: *Use of motion compensation in the efficient coding of image sequences.* YU Info Symp., Brezovica, Yugoslavia, April 1995.
3. ISO/IEC-11172: *Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to* 1.5 *Mbit/s*. Oct. 1992.
4. A. K. Jain: *Fundamentals of Digital Image Processing.* Prentice Hall, Englewood Cliffs, 1989.
5. T. Koga, K. Iinuma, A. Hirano, Y. Iijima and T. Ishiguro: *Motion–compensated interframe coding for video conferencing.* Proc. Nat. Telecomm. Conf. NTC'81, New Orleans, Dec. 1981, pp. G5.3.1-G5.3.5.
6. H. G. Musmann, P. Pirsch and H. J. Grallert: *Advances in Picture Coding.* Proc. IEEE, vol. 73, No. 4, April 1985, pp. 523-548.

7. K. R. Rao and P. Yip: *Discrete Cosine Transform – Algorithms, Advantages, Applications.* Academic Press, London, 1990.