An Adaptive Three-Dimensional DCT Compression Based on Motion Analysis

B. Furht, K. Gustafson, H. Huang, and O. Marques
Florida Atlantic University, Boca Raton, Florida, U.S.A.
{borko,ken2,hhesong,oge}@cse.fau.edu

ABSTRACT

In this paper, we propose an adaptive 3D-DCT compression technique, which dynamically determines an optimal size of the video cube based on the motion analysis. The technique consists of two steps: (a) it analyses the motion within a small (16x16) video cube of eight successive frames, and (b) it selects the size of the cube based on the motion analysis and applies the 3D-DCT algorithm on the selected video cube. The effectiveness of the proposed technique is illustrated by implementing it to a number of video sequences with low, medium, and high motion.

Keywords: video compression, three-dimensional DCT, adaptive block size, motion analysis.

1. INTRODUCTION

In recent years, several research efforts were made to apply 3D-DCT compression in order to achieve high compression ratios required for the transmission of video sequences over low bandwidth channels [1,2,3,4,5]. For such applications, MPEG compression techniques (MPEG-2 and MPEG-4), that are based on 2D-DCT and motion estimation, produce artifacts and blockness effects. Therefore, they are not well suited for transmission of videos at low bit rates.

In the 3D-DCT compression, we assume that pixels in a full-motion video are correlated in the spatial domain as well as in the temporal domain. Therefore, it is reasonable to expect that a similar concentration of energy will take place in both spatial dimensions and the time dimension. Then, the DCT can also be an appropriate transformation for representation of motion data. However, the performance of the traditional 3D-DCT will be degraded for video sequences with high motion. To solve the problem, Chan and Sui proposed a 3D-DCT technique with variable temporal length that is determined by the scene change detection [3]. Lee, Song, and Park developed another 3-D compression that combines 1-D temporal decomposition by the DCT and 2-D wavelet packet decomposition [4]. A fast implementation of the 3D-DCT transformation has been described in [8].

In this paper we propose an adaptive cube-size 3D-DCT technique that dynamically performs motion analysis and accordingly adapts the size of the video cube to be compressed. In this way, for video cubes with high motion the algorithm achieves better quality by compromising compression, while for video cubes with lower motion, it gives higher compression while still maintaining high quality.

2. ADAPTIVE 3D-DCT ALGORITHM

In the three-dimensional DCT compression algorithm, a video sequence is divided into video cubes of size $N_c \times N_r \times N_f$, where $N_c \times N_r$ is an image block of pixels, and N_f is the number of successive frames. The forward 3D-DCT transformation is then defined as:

$$F(u,v,ww) = C(u)C(v)C(w)\sum_{x=0}^{N_r-1}\sum_{y=0}^{N_c-1}\sum_{p=0}^{N_f-1} f(x,y,z)\times \frac{\cos(2x+1)up}{2N_r}\times\frac{\cos(2y+1)vp}{2N_c}\times\frac{\cos(2z+1)wp}{2N_f}$$

where

$$C(k) = \begin{Bmatrix} \frac{1}{\sqrt{2}} \;\; for\;\; k=0 \\ 1 \; otherwise \end{Bmatrix}$$

Originally, selected video cube has been of fixed size for all parts of a video sequence and its dimension has been 8x8xx8 [1,2,3].

2.1 Motion Analysis and the Determination of the Cube Size

We propose a 3D-DCT algorithm with a variable size of the video cube, which is dynamically determined based on the level of motion in each cube. Therefore, the motion analyzer is the first block in the encoding algorithm that partitions a video sequence of eight consecutive frames into video cubes. In our experiments, 16x16x8 video cubes are used for motion analysis. As an example, the partitioning of the security video is shown in Figure 1.

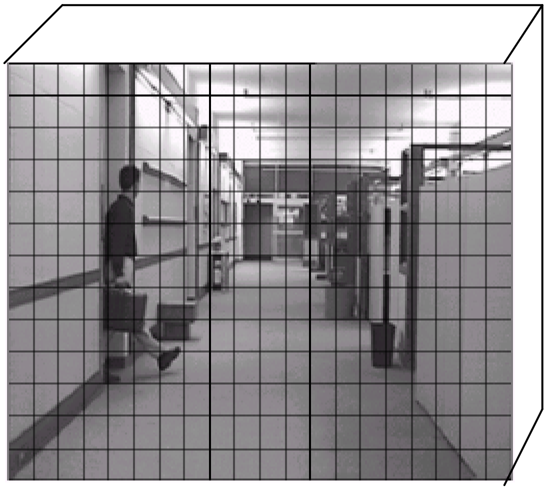

Figure 1. Partitioning a video sequence into 16x16x8 video blocks to perform motion analysis.

Then, for each video cube, the motion analyzer calculates the Normalized Pixel Difference (NPD) between the first and eighth frame as $\frac{1}{N}\sum_{i=1}^{N}\left|X(i)_1 - X(i)_8\right|$, where $X(i)_1$ are pixels from the first frame, $X(I)_8$ are pixels from the eighth frame, and N is the total number of pixels in a 16x16 block (N=256). Depending of the calculated NPD value, the motion in a block is determined as:

$$\begin{aligned}&\textit{No motion} \quad \textit{if} \quad NPD \le t_1\\&\textit{Low motion} \quad \textit{if} \quad t_1 \le NPD < t_2\\&\textit{High motion} \quad \textit{if} \quad NPD > t_2\end{aligned}$$

We conducted a number of experiments to determine values for thresholds t_1 and t_2, and we selected the values t_1=5 and t_2=25. Based on these thresholds, for the eight frames of the security video, the motion analyzer found that 353 blocks are with no motion, 17 blocks are of low motion, and 26 blocks of high motion.

In the case when the motion analyzer detected “no motion” in a video cube, the 3D-DCT is applied to 16x16x1 cube, which is in fact a 2D-DCT transform applied on the block in the first frame only. On the decoding side, pixels in the reconstructed block will be duplicated in the remaining seven frames.

2.2 Architecture of the Adaptive 3D-DCT Coder

After the motion analyzer determined the level of motion, the 3D-DCT transform is applied to various cube sizes according to the Table 1.

Table 1. Determining the Cube Size

Level of Motion	Cube Size
No	16x16x1
Low	16x16x8
High	8x8x8

When the motion analyzer detect “low motion”, the 3D-DCT is applied to entire 16x16x8 cube. However, if a region is estimated as “high motion” region, the cube is subdivided into 8x8x8 cubes, each one subjected to the 3D-DCT transformation. In this way, for high motion regions the algorithm will achieve better quality, however for low motion and no motion regions high compression ratios will be obtained.

After the pixels are transformed into the frequency domain coefficients, the algorithm proceeds with the quantization and Huffman encoding. Most energy is now concentrated in a few low-frequency coefficients, while the majority of high-frequency coefficients have zero or near-zero values, and need not to be encoded. The quantizer reduces the amplitude of the coefficients that contribute little or nothing to the quality of video, with the purpose of increasing the number of zero coefficients. The entropy encoder further compresses the data using a lossless variable length encoding Huffman algorithm. The architecture of the proposed adaptive 3D-DCT encoder using the motion analyzer, is shown in Figure 2.

The proposed algorithm could be extended by including additional levels of motion, for example no, low, medium, and high motion. In that case 8x8x8 3D-DCT can be applied to medium motion cubes, while high motion cubes will use 4x4x8 3D-DCT transform. This will further improve the trade-off between the quality and the compression ratios of the reconstructed video.

Finally, the algorithm can be further improved by implementing adaptive quantization tables. In this case, for high motion cubes, the quantization tables with smaller coefficients are used to produce high quality, while for low motion cubes the quantization tables contain larger coefficients. This part of the algorithm is shown in Figure 2 by using dashed line to indicate that the motion estimator impacts the quantization tables. This will further improve the quality of the reconstructed video for high motion scenes.

In the 3D-DCT decoder, all steps from the encoding process, except the motion analysis, are inverted and implemented in the reverse order.

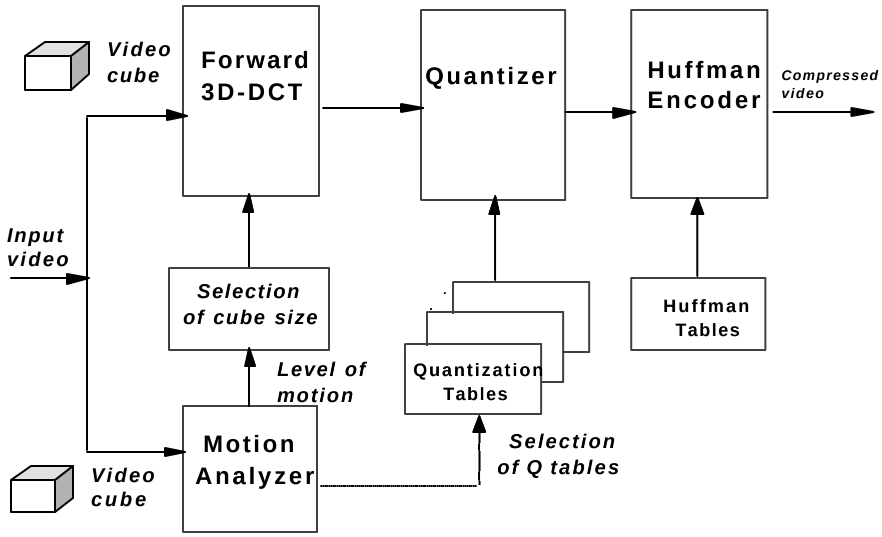

Figure 2. The architecture of the adaptive 3D-DCT encoder.

3. EXPERIMENTAL RESULTS

We applied the adaptive 3D-DCT algorithm to a number of video sequences with various level of motion. We measured the compression ratio, and calculated bits/pixel in the compressed video and the Peak Signal to Noise Ratio (PSNR). The PSNR is defined as:

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

where MSE is the mean square error given as:

$$MSE = \frac{1}{MNP}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\sum_{p=0}^{P-1}(X_{m,n,p} - \hat{X}_{m,n,p})^2$$

where MxN defines the frame size, and P is the number of frames in the sequence, $X_{m,n,p}$ *and* $\hat{X}_{m,n,p}$ are the original and reconstructed pixels in the m^{th} row, n^{th} column, and p^{th} frame, respectively.

We applied the algorithm for a variety of quantization tables (QT) that were created using the following formula:

$$Q(i,j) = 1 + [(1 + i + j) \times quality]$$

where Q(i,j) are quantization coefficients and *quality* specifies the quality factor, and its recommended range is 1 to 25. Quality =1 gives the best quality.

The results for the eight frames of *Security* video in the format YUV (4:2:0) are presented in Table 2, and Figures 3 and 4.

Table 2. Adaptive 3D-DCT applied to *Security* sequence, YUV (4:2:0)

Quality in QT	**Cr**	**Bits/pixel**	**PSNR**
1	46	0.174	37.19
3	88	0.091	34.62
5	120	0.067	32.78
10	190	0.042	30.15
20	408	0.020	27.45

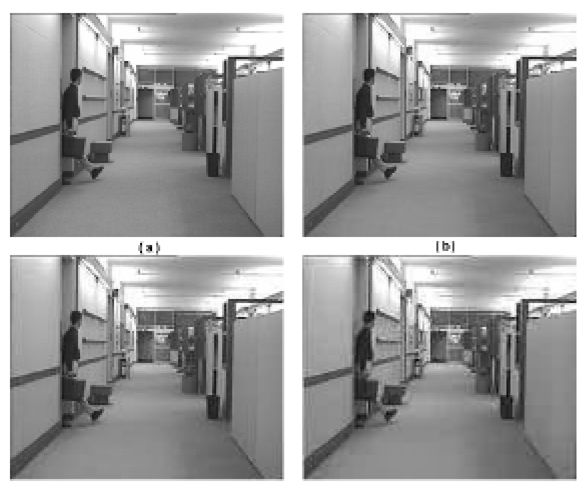

Figure 3. Security video - first frame: (a) original, (b) Q=5, (c) Q=10, and (d) Q=20

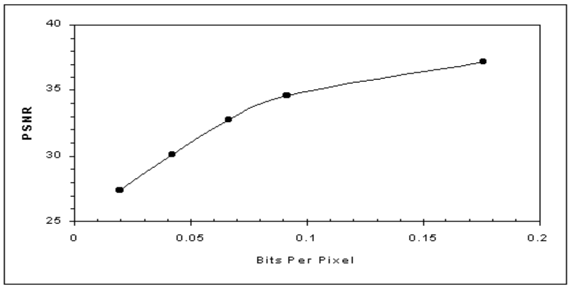

Figure 4. PSNR as a function of bits/pixel for *Security* video

As it can be seen from the experimental results, the proposed adaptive 3D-DCT algorithm performs very well for low-motion video sequences by providing very compression (or low bits/pixels) while maintaining a very good quality of video. For Q=20, the obtained has been Cr=408, however the video begin showing artifacts.

Table 3 shows the results for a very high-motion video – Football sequence. The video consists of 56 frames, and the motion analyzer had detected 1,116 video cubes with high motion, 875 with low motion, and 781 with no motion.

Table 3. Adaptive 3D-DCT applied the *Football* sequence, YUV (4:2:0)

Quality in QT	Cr	Bits/pixel	NRMSE
5	54	0.148	0.011
10	96	0.083	0.013
20	164	0.049	0.016
(5,10,10)	72	0.120	0.012

In Table 3, we also reported results (last row) when we applied adaptive quantization. In this experiment, we used different quantization tables for no/low motion and high motion cubes. Quality=5 was applied for the high motion cubes, and quality=10 for low/no motion cubes. The obtained results show that the quality of the video has been improved compromising the compression ratio.

Table 4. Non-adaptive 3D-DCT applied the *Football* sequence, YUV (4:2:0). All cubes are 8x8x8.

Quality	Cr	Bits/pixel	NRMSE
5	44	0.182	0.010
10	76	0.105	0.012
20	121	0.067	0.015

We also applied the original, non-adaptive 3D-DCT compression on the football sequence assuming that all cubes are of the same size, 8x8x8. The obtained results, shown in Table 4, indicate that the adaptive 3D-DCT is superior because it gives higher compression ratios while maintaining similar quality.

4. CONCLUSIONS

We presented an innovative adaptive 3D-DCT technique for video compression at low bit rates. For a low motion video, expected compression ratios that can be obtained are in the range 1:300 to 1:400 (or about 0.02 bits/pixel), while still maintaining a relatively good quality of video. Potential applications could be for streaming video over the Internet, wireless video, videoconferencing, and videophone. On the other hand, the algorithm dynamically adapts to high-motion video by achieving a high quality and still maintaining relatively high compression ratio in the range of 80-150. Therefore, the algorithm is also well suited for applications such as digital TV, HDTV, and video on demand.

Future work will focus on experimenting with the motion analyzer by extending the number of motion levels and including smaller video cubes for high motion regions (for example, 4x4x8 or 4x4x4). In addition, we intend to experiment with various adaptive quantizations that will further improve the quality of the video. Finally, our objective is the built software encoder and decoder for the proposed algorithm, that will require an implementation of a fast 3D-DCT algorithm, such as one proposed in [6].

REFERENCES

1. R. Westwater and B. Furht, “The XYZ Algorithm for Real-Time Compression of Full-Motion Video,” Real-Time Imaging Journal, Vol. 2, 1996. pp. 19-34.
2. M. Servais and G. De Jager, “Video Compression using the Three-Dimensional Discrete Cosine Transform (3D-DCT),” Proc. of the IEEE Communications and Signal Processing South African Symposium, COMSIG’97, 1997, pp. 27-32.
3. Y-L. Chan and W-C. Siu, “Variable Temporal-Length 3-D Discrete Cosine Transform Coding,” IEEE Transactions on Image Processing, Vol. 6, No. 5, May 1997, pp. 758-763.
4. G.H. Lee, J.H. Song, and R-H. Park, “Three-Dimensional DCT/WT Compression Using Motion Vector Segmentation for Low Bit-Rate Video Coding,” Proc. Of the IEEE International Conference on Image Processing, Vol. 3, 1997, pp. 456-459.
5. G.P. Abousleman, M.W. Marcellin, and B.R. Hunt, “Compression of Hyperspectral Imagery using the 3-D DCT and Hybrid DPCM/DCT,” IEEE Transactions of Geoscience and Remote Sensing, Vol. 33, No. 1, January 1995, pp. 26-34.
6. O. Alshibami and S. Boussakta, “Fast Algorithm for the 3D DCT,” Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 3, 2001, pp. 1945-1948.