An Approach to Image Compression Using Three-Dimensional DCT

Xiuqi Li and Borko Furht
Department of Computer Science and Engineering,
Florida Atlantic University, Boca Raton, FL 33431
Email:xli@cse.fau.edu, borko@cse.fau.edu

Abstract

In this paper we propose a novel approach to image compression based on three-dimensional Discrete Cosine Transformation (DCT). The basic idea is to de-correlate similar pixel blocks through three-dimensional DCT transformation. A number of adjacent pixel blocks are grouped together to form a three-dimensional data cube. Each data cube is 3D DCT transformed, quantized, and Huffman encoded. Experimental results demonstrate the effectiveness of the new approach, specifically for medical and space exploration images.

1. Introduction

Three-dimensional discrete cosine transformation is typically used in video compression because video data are naturally three-dimensional. There have been a number of video compression algorithms that employ 3D DCT. These algorithms can be classified into three categories based on the DCT transformation: fixed-size 3D DCT, variable-size 3D DCT, and hybrid 2D/3D DCT.

In the XYZ video compression algorithm [1] [2], a video clip is divided into groups of 8 frames. Each group is further divided into data cubes of 8x8x8 pixels. Each data cube is then independently 3D-DCT transformed, quantized, and entropy encoded.

The authors in [3] [4] [5] proposed the variable-size data cube. The data cubes in [3] [4] can have variable temporal lengths, while the data cubes in [5] can have both variable temporal lengths and variable spatial sizes. Two scene change detectors are utilized in [3] to detect the temporal boundary of the two adjacent 3D data cubes. They are the Adaptive Block Filter (ABF) method and the Mean Absolute Difference (MAD) method. The algorithm proposed in [5] adaptively determines the optimal size of a video data cube based on the motion analysis. The cube sizes for the no-motion, low-motion, and high-motion scenarios are 16x16x1, 16x16x8 and 8x8x8 pixels respectively.

The 2D/3D hybrid algorithms were proposed in [6] [7]. The algorithm in [6] first estimates motion vectors (MV) based on the deformed frame difference (DFD) between two frames. If the DFD is less than a threshold value, the motion is considered as low-motion. The 2D blocks of 8x8 pixels in the adjacent frames pointed by low-motion MVs are put into one 3D data cube and then transformed using 3D-FDCT. The rest of the regions in each frame are then transformed using 2D-FDCT. In [7], the neighboring blocks that have similar motion vectors and are of the same type (boundary or non-boundary blocks) are merged into an arbitrary region. Regions are classified into motion compliance (MC) regions and motion failure (MF) regions. The temporal boundary between two adjacent cubes was detected using histograms. The MC regions are first coded using 1-D variable-point (2-, 3- or 4-) FDCT along the temporal dimension. Then the 2-D wavelet packet decomposition is applied to MC regions in the same frame. The MF regions are coded using the package contour coding method in [8].

Considering that small blocks of pixels are often correlated in an image, we propose to apply 3D-DCT transformation to image compression. To form the three-dimensional data required for 3D-DCT computation, eight two-dimensional pixel blocks are taken sequentially from an image and are assembled into a 3D cube.

The rest of the paper is organized as follows. The sequential 3D DCT image compression algorithm is presented in Section 2. We describe the architecture of the image codec, the formation of the three-dimensional data cube, and the three-dimensional DCT transformation. The experimental results are presented in Section 3. The conclusion and future work are discussed in Section 4.

2. Sequential 3D DCT Image Compression

This section describes the sequential 3D DCT image compression algorithm in detail. We first present the architecture of the sequential 3D DCT image encoder and decoder. Then, we describe how the three-dimensional data cube is constructed from the two-dimensional data blocks. Finally, we present the three-dimensional DCT transformation and related equations.

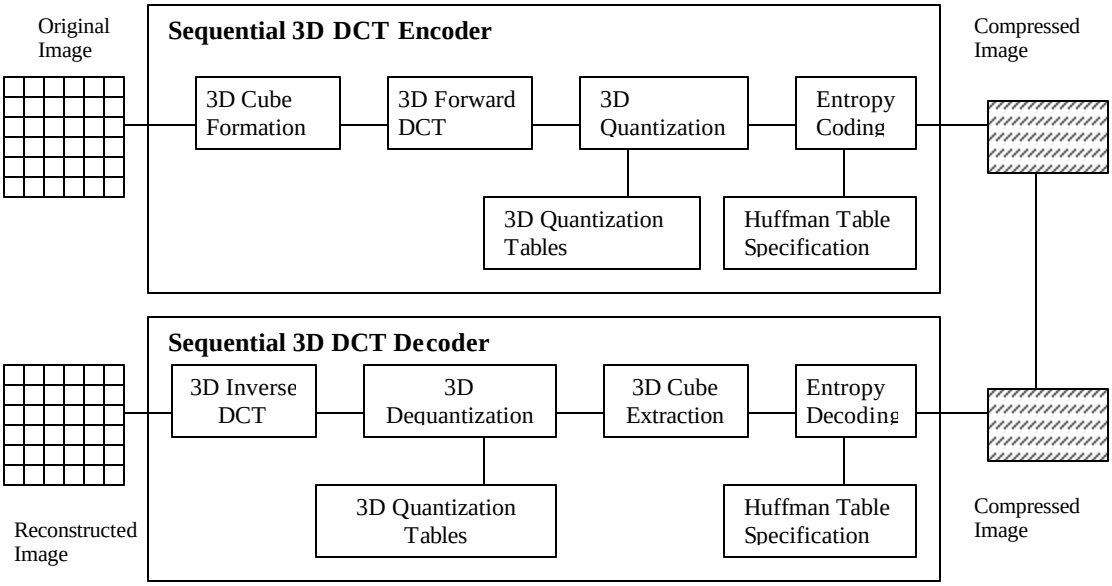

Figure 1. The block diagram of the sequential 3-D DCT codec.

1.1 The Architecture of the Sequential 3D DCT Image Codec

The block diagram of the sequential 3D DCT image codec is shown in Figure 1. The image in RGB format is first converted to YCbCr format to separate the luminance from the chrominance information. After the conversion, each color component is partitioned into blocks of 8x8 pixels. The intensity values of pixels in each block are normalized to [-128, +127]. Each block is then processed in the following order.

1) 3D Data Cube Formation

Starting with the top-left corner of the image, every eight adjacent two-dimensional pixel blocks are taken to construct a three-dimensional data cube. The details of the 3D Cube formation are explained in Section 2.2.

2) 3D FDCT Transformation

The three-dimensional forward discrete cosine transformation is performed on each three-dimensional data cube. The purpose is to reduce the redundancy of similar pixel blocks. After the 3D DCT transformation, only a small number of low-frequency coefficients are significant. Most high-frequency coefficients are near zeros.

3) Quantization

All DCT coefficients in the same data cube are quantized using the following formula:

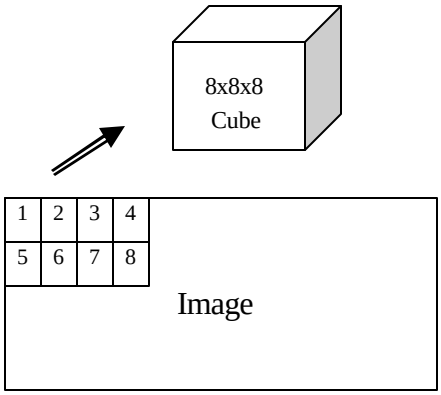

Figure 2. Forming a sequential cube of 8x8x8 pixels.

$$F_q(u,v,w) = round\left(\frac{F(u,v,w)}{Q(u,v,w)}\right) \quad (1)$$

where u, v, and w are the spatial indices; $F(u,v,w)$ refers to the coefficient value before the quantization; $Q(u,v,w)$ denotes the element in the quantization table; $F_q(u,v,w)$ represents the quantized coefficient.

4) Zigzag and Entropy Coding

Each 2-D block in a 3D cube is zigzagged into a vector. The difference between the DC values in the adjacent 3D cubes is computed and Huffman encoded. The run-length of zero AC coefficients and the non-zero AC coefficients in each 2-D block in the same data cube are also Huffman encoded.

(a) The original image

(b) The reconstructed image using 3D DCT (PSNR=41.76, BPP=0.67)

(c) The reconstructed image using JPEG (PSNR=41.58, BPP=0.68)

Figure 3. The experimental results for the image "sonogram."

The decoding procedure is the reverse of the encoding procedure. The compressed image is first Huffman decoded. All 2-D blocks of quantized coefficients in the same data cube are identified and de-quantized. The 3D inverse DCT is then applied to each data cube. At the last stage, the intensity values are shifted back to [0,255].

2.2 3D Data Cube Formation

As illustrated in Figure 2, each 3D data cube is formed using a 2x4 adjacent blocks of 8x8 pixels in the following manner. An 8x8 block is considered as a unit. To form the first data cube, we take the first four blocks in the first row of blocks and number them 1, 2, 3 and 4 respectively. Then, we take the first four blocks in the second row and number them 5, 6, 7, and 8 respectively. These eight blocks are then assembled into a three-dimensional data cube of 8x8x8 pixels. The order in which a block is placed in the 3D data cube is based on its block number.

The second data cube is constructed by taking the next four pixel blocks in the first row of blocks and then the next four pixel blocks in the second row of blocks. Blocks are numbered and ordered in the same way as in the formation of the first data cube. The rest of the image is processed in a similar way.

2.3 3D DCT Transformation

The 3D Forward Discrete Cosine Transformation used in the proposed 3D DCT image coder is based on the following formulas:

$$F(u,v,w) = \frac{C(u)C(v)C(w)}{8} \sum_{x=0}^{7}\sum_{y=0}^{7}\sum_{z=0}^{7} f(x,y,z)\cos\_ prod \qquad (2)$$

where *cos_prod* and *C(u)*, *C(v)*, and *C(w)* are defined as:

$$\cos\_ prod = \cos\frac{(2x+1)up}{16}\cos\frac{(2y+1)vp}{16}\cos\frac{(2z+1)wp}{16} \qquad (3)$$

$$C(i) = \frac{1}{\sqrt{2}}, \quad if \ \mathrm{i} = 0; \quad C(i) = 1, \quad if \ \mathrm{i} > 0; \quad i \in \{u,v,w\} \qquad (4)$$

In the equations (2) – (4), *x,y,z* are pixel indices in the time domain, *u,v,w* are coefficient indices in the frequency domain, *f(x,y,z)* denotes a normalized pixel intensity value, and *F(u,v,w)* is a DCT coefficient value.

The proposed 3D DCT image decoder employs the following 3D Inverse Discrete Cosine Transformation:

$$f(x,y,z) = \frac{1}{8}\left[\sum_{u=0}^{7}\sum_{v=0}^{7}\sum_{w=0}^{7} c(u)c(v)c(w)F(u,v,w)\cos\_ prod\right] \qquad (5)$$

where *cos_prod* is defined as in the formula (3), *C(u)*, *C(v)*, and *C(w)* take the same values as in the formula (4), and *f(x,y,z)* and *F(u,v,w)* have the same meanings as in the formula (2).

3. Experimental Results

The proposed sequential 3D DCT image compression algorithm is benchmarked against the baseline JPEG compression algorithm. Different categories of images are collected for performance evaluation purpose. These categories include medical, texture, art, sculpture, and space exploration images.

The compression efficiency is measured using the compression ratio. The quality of an uncompressed image is measured using the Peak Signal to Noise Ratio (PSNR). The PSNR is computed based on the Mean Square Error (MSE). The formulas (6) and (7) define the PSNR and MSE, respectively:

$$PSNR = 10\log 10\left(\frac{255^2}{MSE}\right) \qquad (6)$$

$$MSE = \frac{1}{MN} \sum_{x=0}^{M} \sum_{y=0}^{N} (I(x,y) - \hat{I}(x,y))^2 \qquad (7)$$

In formulas (6) and (7), M and N refer to the number of pixels in a row and a column respectively, *I(x,y)* signifies the original intensity value of a pixel at spatial location (x,y), and $\hat{I}(x,y)$ denotes the intensity value of the pixel at the same spatial location in the uncompressed image.

The experimental results for a medical image sonogram.bmp are shown in Figures 3 and 4. Figure 3(a) shows the original image, while Figure 3(b) is the reconstructed image using the proposed algorithm when the PSNR is 41.76 dB and the BPP is 0.67. Figure 3(c) is the reconstructed image using JPEG when the PSNR is 41.58 dB and the BPP is 0.68. The graphs showing the PSNR as a function of bits/pixel are shown in Figure 4.

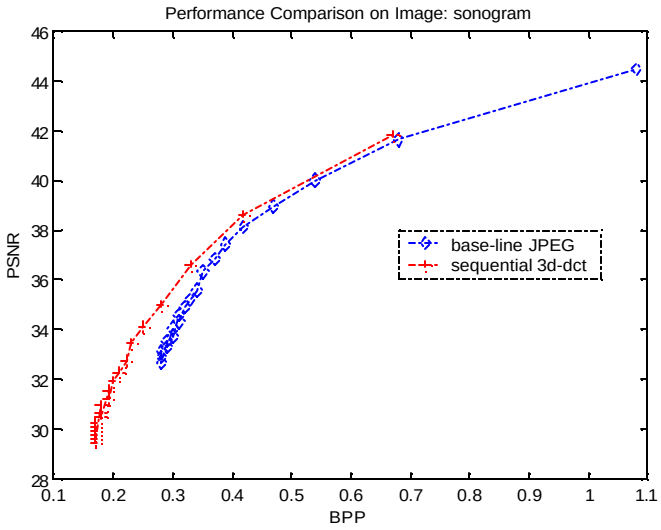

Figure 4. The PSNR as a function of bits/pixels.

The experimental results demonstrate that the proposed approach is slightly better than the JPEG for the specific medical image.

4. Conclusions and Future Work

In this paper, we proposed an innovative image compression algorithm that utilizes three-dimensional discrete cosine transformation. The algorithm first divides an image into 8x8 blocks. Then eight adjacent blocks are taken sequentially and repeatedly to form three-dimensional data cubes, which are required by the 3D DCT transformation. Next, a three-dimensional discrete cosine transformation is performed on each data cube. The DCT coefficients in the same 3D data cube are then quantized and Huffman encoded. The experimental results have shown that the new algorithm is better than JPEG for some classes of images.

The sequential 3D DCT image coder, described in this paper, uses fixed-size data cubes. It does not group all pixel blocks that are similar to each other. We are currently experimenting with a similar coder that uses variable-length three-dimensional cubes. The adjacent 8x8 blocks are first compared to each other, and, if there is a sufficient pixel similarity between them, they are grouped into the same 3D cube. In this case, each cube is of a variable length and consists of blocks which are similar to each other. The blocks may not be ordered in a sequential way. The preliminary results show that the variable-length 3D DCT image coder shows better performance than the sequential coder for specific classes of images.

5. References

[1] B. Furht, "Video Presentation and Compression," in Handbook of Multimedia Computing, CRC Press, Boca Raton, FL, 1999.

[2] R. Westwater and B. Furht "The XYZ Algorithm for Real-Time Compression of Full-Motion Video," Journal of Real-Time Imaging, Vol. 2, No. 1, February 1996, pp. 19-34.

[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] Y.-L. Chan and W.-C. Siu, "Efficient Interframe Transform Coding Using Temporal Context," Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS 96), 1996, pp. 786-789.

[5] B. Furht, K. Gustafson, H. Hesong, and O. Marques, "An Adaptive Three-Dimensional DCT Compression Based on Motion Analysis," Proceedings of the ACM Symposium on Applied Computing, 2003.

[6] I. Kiyohisa, T. Yoshida, and I.Y. Nishihara, "2D/3D Hybrid Video Coding Based on Motion Compensation," Proceedings of the International Conference on Image Processing, 1999.

[7] 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," Proceedings of the International Conference on Image Processing, 1997, pp. 456-459.

[8] J.J. Chae, S.B. Chae, W.Y. Choi, and R.-H. Park, "Effective Contour Coding Techniques Using 2x2 Blocks," Proceedings of the International Symposium on Information and its Applications, 1990.