

# USE OF ADAPTIVE RESIZING IN 3-D DCT DOMAIN FOR VIDEO CODING

Jin Li<sup>1</sup>, Jarmo Takala<sup>1</sup>, Moncef Gabbouj<sup>1</sup> and Hexin Chen<sup>2</sup>

Department of Information Technology  
Tampere University of Technology, Tampere, Finland<sup>1</sup>  
School of Communication Engineering  
Jilin University, Changchun, China<sup>2</sup>

## ABSTRACT

This paper proposes an adaptive resizing algorithm in DCT domain for 3-D DCT based video codec. An  $8 \times 8 \times 8$  cube is resized to three modes along temporal dimension: a single  $8 \times 8$  block, a downsized  $8 \times 8 \times 4$  cube and two  $8 \times 8 \times 4$  cubes. The mode selection is based on the local motion activity and determined after 2-D DCT on each block. In addition, the proposed algorithm even simplifies the computational complexity for sequences with low motion activity. Experimental results show that the proposed algorithm can improve the coding efficiency for different types of video sequences. Best performance can be expected for those with low motion activity. Moreover, it outperforms other variable size of 3-D DCT schemes. Potential applications could be for portable digital devices with restrict battery lifetime and other areas with restrict real-time requirement.

**Index Terms** — 3-D DCT, video compression, resizing, computational complexity

## 1. INTRODUCTION

Most today's video coding standards such as H.263 and MPEG-1, 2, 4 rely on motion estimation/compensation to exploit the temporal correlations among inter frames. However, this requires a large number of computations and great computational power for hardware. Therefore, it is not appropriate for most portable devices like mobile phone and digital video cameras. Since for these types of applications, low complexity implementation and low power consumption are still the most critical issues.

An alternative approach is to use the three-dimensional discrete cosine transform (3-D DCT). A 3-D DCT based video codec extends 2-D DCT to the temporal dimension and remove the temporal correlations instead of utilizing motion estimation/compensation. Therefore, it is able to reach adequate compression efficiency with much less computational complexity.

One of the major problems in a 3-D DCT video codec is that if only utilizing fixed-length transform regardless the level of motion activity, the compression efficiency is usually inferior to most today's video standards. To solve this problem, adaptive 3-D DCT schemes are proposed in [1]-[4]. These techniques utilize variable size of 3-D DCT based on local motion activity. All the methods deal with sequences containing static background such as Miss American and Akiyo well, but the compression efficiency

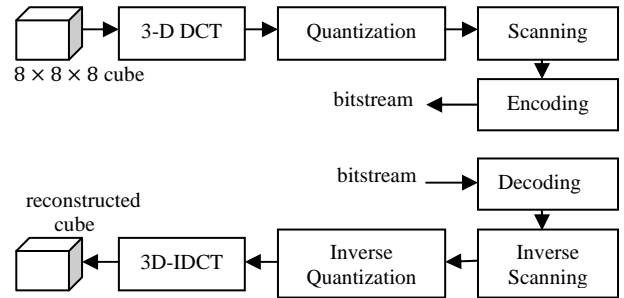


Fig. 1 Block diagram of typical 3-D DCT based video codec

maintains the same for those like Glasgow and Foreman. In addition, since the motion analysis is always performed in the spatial domain, the computations are increased. In 2005, a simplified 3-D DCT scheme was proposed in [5]. Compared to other approaches, it can simplify the computational complexity and increase the coding speed.

In what follows we describe an adaptive resizing algorithm in 3-D DCT domain. The  $8 \times 8 \times 8$  cubes are resized to three modes based on the level of local motion activity. Compared to other schemes, this technique tends to lose less energy and thus gives better compression performance. Moreover, it even reduces the computational cost compared to the baseline video codec.

The rest of this paper is organized as follows. In section 2 we briefly review the basics of 3-D DCT and the resizing algorithm. We describe the proposed algorithm in section 3. The experimental results are presented in section 4. Finally, we conclude the paper in section 5.

## 2. OVERVIEW OF 3-D DCT AND RESIZING ALGORITHM

### 2.1. Three-dimensional Discrete Cosine Transform

In a 3-D DCT based codec, a video sequence is divided into a number of  $M \times N \times L$  cubes, where  $M \times N$  is an image block of pixels, and  $L$  is the number of successive frames. The forward 3-D DCT is then defined as

$$F(u, v, w) = C(u, L)C(v, N)C(w, M) \sum_{x=0}^{L-1} \sum_{y=0}^{N-1} \sum_{z=0}^{M-1} f(x, y, z) \times \frac{\cos(2x+1)u\pi}{2L} \times \frac{\cos(2y+1)v\pi}{2N} \times \frac{\cos(2z+1)w\pi}{2M} \quad (1)$$

where

$$C(k, P) = \begin{cases} \sqrt{1/P} & k = 0 \\ \sqrt{2/P} & \text{otherwise} \end{cases}$$

The 3-D DCT can be computed by taking one-dimensional transform separately in each of the three dimensions. Although a number of transforms are required, the computational complexity – even without taking the motion estimation into account – is superior to 2-D DCT based video encoder. A typical 3-D DCT based codec is described in Fig. 1.

## 2.2. The Resizing Algorithm

The outline of resizing scheme [6]-[8] for 1-D signals is shown in Fig. 2. Let  $\mathbf{B}_1$  and  $\mathbf{B}_2$  denote the 4-point DCT of two consecutive 4-sample blocks  $\mathbf{b}_1$  and  $\mathbf{b}_2$ , and let  $\tilde{\mathbf{B}}$  be the resulting resized block. In principle, the resizing scheme can be viewed as follows.

- 1) Take 4-point DCT in block  $\mathbf{b}_1$ , and similarly for  $\mathbf{b}_2$ ;
- 2) Take 2-point inverse DCT of the two low-frequency coefficients in  $\mathbf{B}_1$ , and the same as  $\mathbf{B}_2$ ;
- 3) Combine the two 2-point blocks  $\tilde{\mathbf{b}}_1$  and  $\tilde{\mathbf{b}}_2$  into one block  $(\tilde{\mathbf{b}}_1, \tilde{\mathbf{b}}_2)$  and then take its 4-point DCT. The resulting block is the desired block  $\tilde{\mathbf{B}}$ .

The resizing method relies on two main principles. One, if the consecutive blocks are similar enough, the resizing operation does not produce significant high-frequency coefficients into the resulting block. Two, if the quantizer is going to remove the high-frequency of the blocks, the truncation of high-frequency information with downsizing operation is justified.

The resizing algorithm can be applied to the 3-D DCT video coding process to remove the temporal correlations more effectively. If consecutive DCT cubes contain only low-frequency contents, the algorithm can produce tighter information representation, thus obtain better compression performance.

## 3. PROPOSED ALGORITHM AND ARCHITECTURE

We proposed a 3-D DCT algorithm with variable size of cube. The mode selection is determined relying on the local motion activity within each cube. Totally, three modes are utilized to perform the transform in temporal direction. One, if the adjacent 2-D DCT coefficients in a cube are almost equal in the temporal dimension, which means that the cube contains no motion or very low motion, we apply 2-D DCT only to the first block instead of the whole cube.

Two, if the cube contains mild motion, we resize the  $8 \times 8 \times 8$  cube to a new  $8 \times 8 \times 4$  cube. Third, if the cube contains high motion activity, we take 3-D DCT for the two  $8 \times 8 \times 4$  cubes separately.

In the proposed method, the mode decision and the resizing operation are operated in the process of temporal transform, thus the 2-D DCT can be first performed for each block as

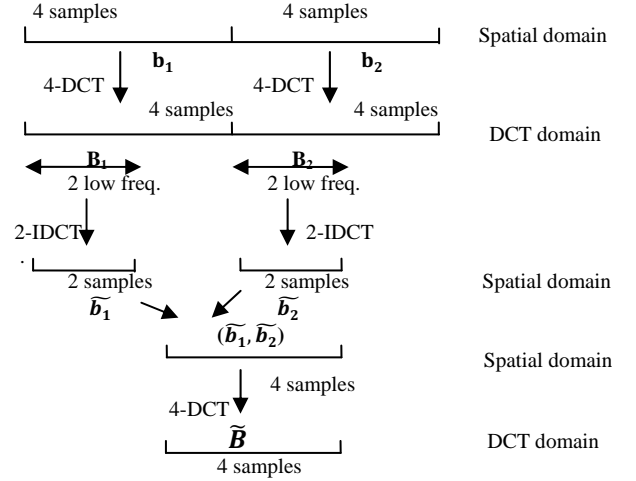


Fig.2 Resizing in DCT domain

$$F(u, v, t) = C(u, 8)C(v, 8) \sum_{x=0}^7 \sum_{y=0}^7 p(x, y, t) \quad 0 \leq t \leq 7 \quad (2)$$

$$\frac{\cos(2x+1)u\pi}{16} \times \frac{\cos(2y+1)v\pi}{16}$$

where  $p(x, y, t)$  is the pixel value at position  $(x, y, t)$  in the cube and  $F(u, v, t)$  is the transformed coefficients.

### 3.1 Mode 1: 2-D DCT

The selection between 3-D DCT and the approximation algorithm 2-D DCT is made by computing Normalized Pixel Difference (NPD) of DC coefficient and a few AC coefficients and by comparing NPD with the threshold  $T_1$ . The NPD is calculated between the first and the rest seven time-dependent  $8 \times 8$  blocks within the cube as

$$NPD(t) = \frac{1}{4} \sum_{u=0}^1 \sum_{v=0}^1 |F(u, v, t) - F(u, v, 0)| \quad \forall t \in \{1, 2, \dots, 7\} \quad (3)$$

If even one of the seven resulting values exceeds the fixed threshold, which means that the cube is not constant enough to be approximated, the 3-D DCT is selected. Otherwise, we apply 2-D DCT only to the first block of the cube. In the decoding side, pixels in the reconstructed block will be duplicated in the remaining seven blocks.

### 3.2 Mode 2: Resizing of 3-D DCT

Mode 1 is only suitable for the cubes with low motion activity. For other cubes containing mild motion activity, the approximation algorithm 2-D DCT does not work well anymore, because simple duplication will significantly decrease the reconstructed video quality. For these types of cubes, using of the resizing algorithm can remove the temporal correlations more effectively and obtain more compact coefficients presentation. The resizing of 3-D DCT includes the following steps.

- 1) After the 2-D DCT for each block done, we continue the transform along temporal dimension on the first four blocks and the last four blocks.

$$F_1(u, v, w) = C(w, 4) \sum_{t=0}^3 F(w, v, t) \frac{\cos(2t+1)w\pi}{8} \quad (4)$$

$$F_2(u, v, w) = C(w, 4) \sum_{t=4}^7 F(w, v, t) \frac{\cos(2t+1)w\pi}{8} \quad (5)$$

- 2) The two  $8 \times 8 \times 4$  cubes are then quantized by quantization parameter  $Q_{uvw}$

$$F_i^Q(u, v, w) = (F_i(u, v, w)) / Q_{uvw} \quad (6)$$

$$\forall u, v \in \{0, 1, \dots, 7\}, 0 \leq w \leq 3 \text{ and } i = 1, 2$$

- 3) As aforementioned, two conditions have to be satisfied to ensure that the resizing algorithm really improve the compression efficiency. This is done by comparing NPD of the two transformed cubes and checking the high-frequency information.

$$NPD = \frac{1}{8} \sum_{u=0}^7 \sum_{v=0}^7 \sum_{w=0}^3 |F_1^Q(u, v, w) - F_2^Q(u, v, w)| \quad (7)$$

$$F_i^Q(u, v, w) = 0 \quad \forall u, v \in \{0, 1, \dots, 7\}, \quad (8)$$

$$i = 1, 2, w = 2, 3$$

If both NPD is smaller than the threshold  $T_2$  and (8) is fulfilled, we apply the resizing algorithm.

- 4) Since the high-frequency coefficients are truncated to zeros, we only take the inverse DCT in temporal dimension for the first two blocks of each cube and then combine them into a new  $8 \times 8 \times 4$  cube.

$$\begin{bmatrix} IF_i(u, v, 0) \\ IF_i(u, v, 1) \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \times \begin{bmatrix} F_i(u, v, 0) \\ F_i(u, v, 1) \end{bmatrix} \quad i = 1, 2 \quad (9)$$

The new combined cube is  $IF_{new}$ , it satisfies

$$IF_{new}(u, v, t) = \begin{cases} IF_1(u, v, t) & \text{if } t = 0, 1 \\ IF_2(u, v, t-2) & \text{if } t = 2, 3 \end{cases} \quad (10)$$

- 5) Take the transform in temporal dimension for  $IF_{new}$  as (4), we resize a  $8 \times 8 \times 8$  cube into a new  $8 \times 8 \times 4$  transformed cube.

### 3.3 Mode 3: $8 \times 8 \times 4$ 3-D DCT

If mode 1 and 2 are not fulfilled, it indicates that the cube contains high local motion activity. In this case, smaller cube size in temporal dimension is usually better since the temporal information is not highly correlated. Therefore, if one of the two conditions in (7) and (8) is not fulfilled, we directly encode the two  $8 \times 8 \times 4$  cubes  $F_1^Q$  and  $F_2^Q$  instead of the  $8 \times 8 \times 8$  cube.

The proposed algorithm to encode an  $8 \times 8 \times 8$  cube can be summarized as

- 1) Take 2-D DCT for each block of the cube ;
- 2) Calculate NPD among blocks. If  $NPD \leq T_1$ , encode as mode 1;

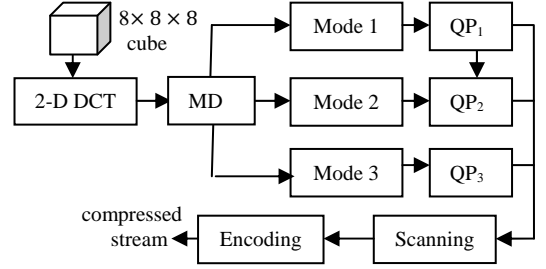


Fig. 3 Architecture of proposed 3-D DCT based encoder. MD denotes mode decision and  $QP_i$  is the quantization parameter of Mode  $i$ .

- 3) If  $NPD > T_1$ , take 1-D DCT along temporal direction for each consecutive four blocks.
- 4) Calculate NPD of the two cubes. If  $NPD \leq T_2$  and (8) is fulfilled, encode as mode 2.
- 5) Otherwise, encode as mode 3.

Fig. 3 gives the architecture of the proposed encoder. Moreover, two extra bits are encoded for each cube to indicate the mode decision in the encoder. In the decoder, all steps from the encoding process, except the mode analysis, are implemented in the reverse order.

## 4. EXPERIMENTAL RESULTS

The proposed 3-D DCT algorithm was tested against the baseline 3-D DCT codec and the reference codec in [1]. Video sequences with various motion activities were encoded and decoded. The Peak Signal to Noise Ratio (PSNR) versus compression ratio (CR) curves were plotted based on the obtained results.

The quantization parameter (QP) for AC coefficients is uniform, and they satisfy the following relationship for different mode

$$2QP_1 = QP_2 = QP_3$$

where  $QP_i$  denotes the QP of mode  $i$ .

In the experiments, the threshold  $T_1$  and  $T_2$  were fixed to a constant value of 8. The quantization parameter for DC coefficients was 10 for the three modes. Experimental results show that the proposed algorithm can give better compression performance for different type of sequences. Best improvement can be expected for sequences with low motion activity. For others containing high motion activity the proposed algorithm can gain 0.5-1.2dB for luminance components and 0.3-0.8dB for chrominance components. Fig. 4 shows the luminance PSNR versus CR curves of Akiyo and Glasgow. According to the obtained results, the proposed scheme shows notable improvements over the baseline codec. Moreover, it gives better compression result than the reference codec.

Table I shows the utilization ratio of the three modes in the proposed video codec. Since Mode 1 is determined irrelevant to QP, the utilization ratio is a constant value for a specific sequence. Whereas, the decision of Mode 2 and Mode 3 is partly dependent on QP, the utilization changes with different QP. Fig. 5 shows the utilization of the three

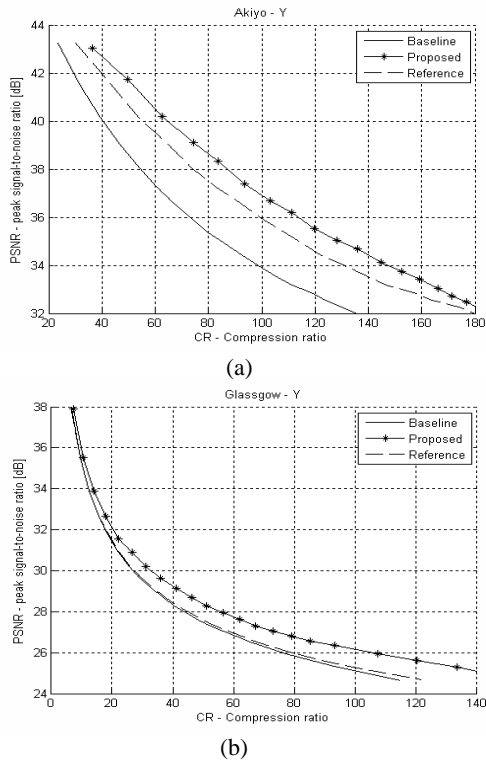


Fig. 4 The luminance PSNR curves are plotted versus the compression ratio for (a) Akiyo and (b) Glasgow based on three different codec: the baseline 3-D DCT based codec, the proposed 3-D DCT based codec and the reference codec in [1].

modes on frame 100 of Akiyo sequence and frame 145 of Glasgow sequence and how it changes with different QP.

Moreover, the proposed algorithm can even reduce the computations for sequences with low motion activity. Take Akiyo for example, the required computations for 3-D DCT in the proposed codec are less than 84.9% of the baseline codec. For Glasgow, the increase does not exceed 18.4% at the worst case.

The selection of the thresholds  $T_1$  and  $T_2$  is empirically determined. In the experiments, we also tested different thresholds of 0, 4, 8, 12, 20 and finally the threshold of 8 usually gives best PSNR versus CR results.

## 5. CONCLUSION

An adaptive resizing algorithm in DCT domain is proposed for 3-D DCT based video process. This proposed 3-D DCT coding adaptively resize the temporal length of the cube depending on the local motion activity. A series of experiments show that the proposed resizing algorithm can give better compression results than the baseline codec and the reference codec.

Although the compression efficiency is still lower than H.264, the encoding process of 3-D DCT is much faster. This makes the 3-D DCT video codec especially suitable for such devices with restrict computational power. Potential applications of the proposed 3-D DCT approach could be for portable digital devices like mobile phone and digital video cameras. Moreover, since the proposed algorithm requires less computations, it is also suited for applications with restrict real-time requirement.

Table I Utilization Ratio of Proposed Modes

QP	Akiyo			Foreman			Glasgow		
	M1 (%)	M2 (%)	M3 (%)	M1 (%)	M2 (%)	M3 (%)	M1 (%)	M2 (%)	M3 (%)
8	66.1	23.7	10.2	2.1	29.6	68.3	2.4	13.1	84.5
16	66.1	28.9	5.0	2.1	49.9	48.0	2.4	23.8	73.8
24	66.1	31.0	2.9	2.1	61.4	36.5	2.4	32.0	65.6
32	66.1	32.3	1.6	2.1	69.3	28.6	2.4	38.8	58.8
40	66.1	33.0	0.9	2.1	75.2	22.7	2.4	44.6	53.0
50	66.1	33.4	0.5	2.1	80.3	17.6	2.4	51.1	46.5
66	66.1	33.7	0.2	2.1	85.3	12.6	2.4	60.1	37.5

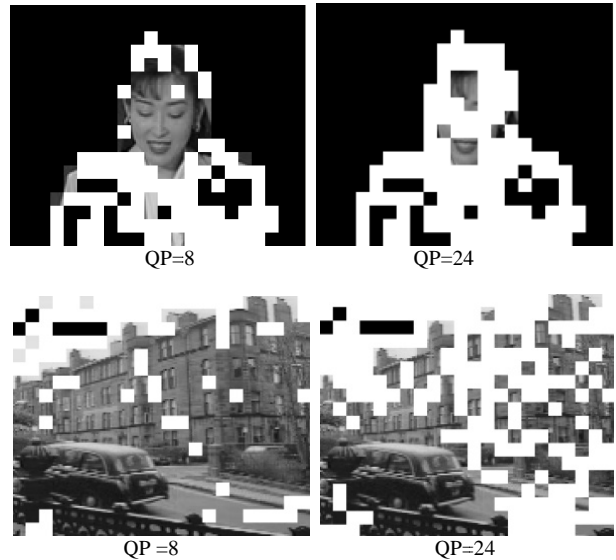


Fig. 5 Utilization of proposed modes on frame 100 of Akiyo and frame 145 of Glasgow, Black: Mode 1, White: Mode 2 and Other area: Mode 3

## 6. ACKNOWLEDGEMENT

This work is supported partly by Chinese Science & Technology Ministry under Grant 2005DFA10300 and by Academy of Finland under Grant 117065.

## 7. REFERENCES

- [1] N.P. Sgouros, S.S. Athineos, P.E. Mardaki and *etc.*, "Use of an Adaptive 3D-DCT Scheme for Coding Multiview Stereo Images," *IEEE Proceedings of ISSPIT*, pp.180-185, 2005
- [2] 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, pp. 758-763, 1997
- [3] B. Furht, K. Gustafson, H. Huang, and O. Marques, "An Adaptive Three-Dimensional DCT Compression Based on Motion Analysis," *Proceedings of ACM*, pp. 765-768, 2003.
- [4] S.C. Tai, Y.G. Wu and C.W. Lin, "An Adaptive 3-D Discrete Cosine Transform Coder for Medical Image Compression," *IEEE Transactions on Information Technology in Biomedicine*, Vol.4, pp. 259-264, 2000
- [5] J.J. Koivusaari, and J.H. Takala, "Simplified Three-Dimensional Discrete Cosine Transform Based Video Codec," *SPIE Proceedings of Multimedia on Mobile Devices*, pp. 11-21, 2005.
- [6] R. Dugad and N. Ahuja, "A Fast Scheme for Image Size Change in The Compressed Domain," *IEEE Transactions on Circuits and Systems for Video Technology* 11, pp.461-474, 2001
- [7] J.J. Koivusaari, J.H. Takala, and M. Gabbouj, "Image Coding Using Adaptive Resizing in The Block-DCT Domain," *SPIE Proceedings of Multimedia on Mobile Devices II*, pp. 1-9, 2006.
- [8] Y.S. Park and H.W. Park, "Arbitrary-Ratio Image Resizing Using Fast DCT of Composite Length for DCT-based Transcoder," *IEEE Transactions on Image Processing* 15, pp.494-500, 2006