LOW-COMPLEXITY ADAPTIVE BLOCK-SIZE TRANSFORM BASED ON EXTENDED TRANSFORMS

Honggang Qi[1], Wen Gao [1], Siwei Ma[2], Debin Zhao [1] and Xiangyang Ji[1]

[1]Institute of Computing Technology, Chinese Academy of Sciences
E-mail: {hgqi, wgao, dbzhao, xyji}@jdl.ac.cn
[2]Department of Electrical Engineering and Integrated Medias Center
University of Southern California
E-mail: swma@usc.edu

ABSTRACT

In this paper, a low-complexity 8x8/4x4 Adaptive Block-size Transform (ABT) scheme is tentatively proposed for the Chinese Audio and Video coding Standard (AVS)[1]. In the proposed ABT scheme, an integer 8x8 transform is derived from the integer 4x4 transform used in AVS according to a transform extension principle. The 8x8 transform not only has high energy compacted property but also can be merely implemented within several additions and shifts, and all intermediate results are limited with 16-bit. The 8x8 transform and 4x4 transform can be merged together and share the same scale matrix so that the hardware units and storage resources are efficiently saved for both encoder and decoder. The experimental results on numerous sequences show that the proposed ABT scheme can achieves significant performance improvement for AVS.

1. INTRODUCTION

The Discrete Cosine Transform (DCT) is an effective transform coding tool for image and video coding. In hybrid video coding, it is used to convert the correlated prediction residuals to seldom correlated frequency domain coefficients for more efficient data compression. Different coding standards employ different transforms. MPGE-2 adopts an 8x8 floating-point DCT, which is a real DCT with fine compression performance but high complexity. H.264/AVC adopts a 4x4 integer DCT [1], which has similar properties to real 4x4 DCT but relative low complexity. Since the inverse DCT in the encoder and the decoder are defined by exact integer operations, the problem of mismatches is avoided.

The different transform sizes possess different coding performances. Larger transform has a better energy compaction than smaller transform. However, the small transform prevents more ringing artifacts in coded pictures [2]. To exert the virtues of different transform sizes greatly, Adaptive Block-size Transform (ABT) scheme is employed in image and video coding standards. The scheme can improve the coding performance significantly. For example, the average luma PSNR gain in high-definition sequences is up to 0.5dB in H.264/AVC. In ABT scheme, the transforms in different sizes are adaptively chosen by encoder. Usually, RD decision is employed in encoder for choosing the better transform [2]. With RD decision, the optimal coding performance can be achieved in RD sense. Since two transform sizes are employed in ABT scheme, two zig-zag scans are needed for corresponding transforms. Moreover, the RD decision requires that the current block is coded twice with the two transforms. It is clear that the ABT scheme introduces high complexity to encoder. However, in view of its high coding performance, the increased complexity is tolerable in some applications. Thus, ABT scheme is adopted into Fidelity Range Extensions of H.264/AVC by Joint Video Team (JVT).

In this paper, the ABT scheme with extended transforms is proposed for Chinese Audio and Video coding Standard (AVS). In this scheme, the extended 8x8 transform is derived from the original 4x4 transform in AVS. The extended 8x8 transform not only improves the coding performance of ABT but also reduces the memory requirements and hardware units for both encoder and decoder. Since the computation and storage complexities of ABT mainly focus on the encoder and only some storage complexities are increased in the decoder, the proposed method can reduce the complexities of decoder efficiently. Thus, the method is meaningful to some applications where the decoders are paid more attention than encoder.

The rest of this paper is organized as follows: In the section 2, the concept of extended transforms is introduced, and an 8x8 transform which is an extension of 4x4 transform of AVS is designed. In section 3, the implementation of ABT in AVS is briefly specified. In

[1] The Chinese audio and video coding standard, an emerging standard targeting at higher coding efficiency and lower complexity.

section 4, the experimental results on test sequences with different coding characteristics are shown. Finally, the conclusions are made by us.

2. THE EXTENDED TRANSFORMS OF ABT

Extended transforms means that the *N*x*N* transform as a part involves in the 2*N*x2*N* transform [4]. Taking 4x4 and 8x8 transform as an example, if the 4x4 transform is expressed with (1), its extended 8x8 transform should be expressed as (2).

$$D4 = \begin{bmatrix} c_0 & c_0 & c_0 & c_0 \\ c_2 & c_6 & -c_6 & -c_2 \\ c_4 & -c_4 & -c_4 & c_4 \\ c_6 & -c_2 & c_2 & -c_6 \end{bmatrix} \quad (1)$$

$$D8 = \begin{bmatrix} c_0 & c_0 & c_0 & c_0 & c_0 & c_0 & c_0 & c_0 \\ c_1 & c_3 & c_5 & c_7 & -c_7 & -c_5 & -c_3 & -c_1 \\ c_2 & c_6 & -c_6 & -c_2 & -c_2 & -c_6 & c_6 & c_2 \\ c_3 & -c_7 & -c_1 & -c_5 & c_5 & c_1 & c_7 & -c_3 \\ c_4 & -c_4 & -c_4 & c_4 & c_4 & -c_4 & -c_4 & c_4 \\ c_5 & -c_1 & c_7 & c_3 & -c_3 & -c_7 & c_1 & -c_5 \\ c_6 & -c_2 & c_2 & -c_6 & -c_6 & c_2 & -c_2 & c_6 \\ c_7 & -c_5 & c_3 & -c_1 & c_1 & -c_3 & c_5 & -c_7 \end{bmatrix} \quad (2)$$

It can be found from (1) and (2) that the even basis functions of 8x8 transform *D8* inherit all basis functions of 4x4 transform *D4*. i.e. $D8_{2i,j} = D4_{i,j}$, $(0 \leqslant i, j < 4)$. The relation can be more clearly expressed from their butterfly flow graph in Fig.1, where the two 4x4 transforms $D4^{II}$ and $D4^{IV}$ mean the II-type and IV-type DCT respectively [3]. If $D4^{II}$ is identical to *D4*, it is said that *D8* is extended from *D4*.

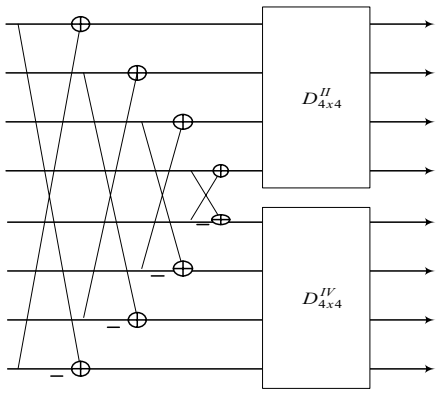

Fig.1. Fast DCT butterfly signal flow graph.

2.1. 4x4 transform

A low-complexity integer 4x4 transform is employed in AVS-M (a version of AVS orient to mobile applications), The transform is expressed as follows:

$$C4 = \begin{bmatrix} 2 & 2 & 2 & 2 \\ 3 & 1 & -1 & -3 \\ 2 & -2 & -2 & 2 \\ 1 & -3 & 3 & -1 \end{bmatrix} \quad (3)$$

The transform has a good energy compaction and low complexity. It can be implemented with merely several addition and shift operations. In practice, the *C4* is just a core matrix of DCT. The norms of its some rows are unequal. A scale matrix is needed for normalizing the core matrix. As the output of the transform and scale of the input *X*, *Y* is expressed as

$$Y = (C4 \times X \times C4') \otimes S4 \quad (4)$$

where the notation ⊗ denotes element-by-element multiplication, and *S4* means the 4x4 scale matrix. The scale matrix *S4* can be calculated by

$$S4_{x,y} = 1 \Big/ \sqrt{(\sum_{j=0}^{3} C4_{x,j}^2) \times (\sum_{i=0}^{3} C4_{i,y}'^2)} \quad (5)$$

2.2. 8x8 transform design

According to the requirement of performance, the integer 8x8 transform should preserve the real DCT compaction property. Its basis functions of transform matrix should approximate the real transform's. Meanwhile, these basis functions should not be so large that the transform can be implemented within narrow bit range. Moreover, the integer 8x8 transform should be an extension of the integer 4x4 transform *C4* so that the 8x8 transform can be shared by the 4x4/8x8 transforms. Thus, the integer 8x8 transform should meet the following three design rules:

1) The basis functions of each row of the designed 8x8 transform should correlate closely with corresponding floating-point basis functions of real 8x8 transform where the correlation is measured by the cosine of the included angle between them;
2) The basis functions of the designed 8x8 transform matrix should be represented by several simple integers;
3) The designed 8x8 transform must be the extension of 4x4 transform of AVS i.e. meeting the relation: $C8_{2x,y} = C4_{x,y}$, $0 \leqslant x, y < 4$.

According to the above rules, an integer 8x8 DCT is expressed as follows:

$$C8 = \begin{bmatrix} 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \\ 6 & 6 & 3 & 2 & -2 & -3 & -6 & -6 \\ 6 & 2 & -2 & -6 & -6 & -2 & 2 & 6 \\ 6 & -2 & -6 & -3 & 3 & 6 & 2 & -6 \\ 4 & -4 & -4 & 4 & 4 & -4 & -4 & 4 \\ 3 & -6 & 2 & 6 & -6 & -2 & 6 & -3 \\ 2 & -6 & 6 & -2 & -2 & 6 & -6 & 2 \\ 2 & -3 & 6 & -6 & 6 & -6 & 3 & -2 \end{bmatrix} / 2 \qquad (6)$$

where *C8* is an integer orthogonal matrix. Its even basis functions and those of integer 4x4 transform matrix *C4* meet the relation, $C8_{2x,y}=C4_{x,y}$. Thus, *C8* is an extended transform of *C4*. The odd basis functions of *C4* consists of integers (6, 6, 3, 2), which are so simple that dynamic range is only increased by 5.1 bits ($\log_2 34$). The cosine of the included angle between the integer array (6, 6, 3, 2) and the float array of real DCT (0.4904, 0.4157, 0.2778, 0.0975) is 0.9834 very near to the upper limit 1. Thus, it has the similar compacting property to the real DCT. Similarly to the normalization of *C4*, a scale matrix *S8* for normalization of *C8* is needed. The scale matrix *S*8 is expressed as

$$S8_{x,y} = 1/\sqrt{(\sum_{j=0}^{7} C8_{x,j}^2)\times(\sum_{i=0}^{7} C8'^2_{i,y})} \qquad (7)$$

since $C8_{2x,y}=C4_{x,y}$, $0 \leqslant x, y<4$, thus

$$\begin{aligned} S4_{x,y} &= 1/\sqrt{(\sum_{j=0}^{3} C4_{x,j}^2)\times(\sum_{i=0}^{3} C4'^2_{i,y})} \\ &= 1/\sqrt{(\sum_{j=0}^{7} C8_{2x,j}^2/2)\times(\sum_{i=0}^{7} C8'^2_{i,2y}/2)} = S8_{2x,2y}/2 \end{aligned} \qquad (8)$$

The division by 2 in above formula can is implemented with just one-bit right-shift operation in hardware. The relation is useful for us to merge the 4x4 scale matrix into the 8x8 scale matrix. The normalizations of 4x4 and 8x8 transforms can be done with only an 8x8 scale matrix. The above analysis on forward transforms can be also analogically done on inverse transforms.

3. THE IMPLEMENTATION OF ABT SCHEME

The ABT scheme is implemented into AVS-M reference software. Since there is only intra 4x4 prediction mode in AVS-M and the 4x4 transform usually has better performance than 8x8 transform in this mode, the proposed scheme is just applied to the luma inter-prediction mode. Moreover, numorous experimental results show that ABT applying to chroma signal does not bring the significant coding gains. Thus, from the view point of complexity， ABT scheme is not used in chroma signals. Unlike the conventional schemes where all 8x8, 8x4, 4x8 and 4x4 transforms are used, no more transforms are used in proposed scheme except for 8x8 and 4x4 transforms. Because the 8x4 and 4x8 transforms just obtain small PSNR gains while introducing large coding loads to the codec.

RD optimizer is used to choose the transform size with the better coding performance for current coded block. Since the RD optimizer just works in the encoder, a one-bit flag is added into bitstream to signal the decoder which transform is used to coding the current block. For further reducing the complexity, the 4x4 and 8x8 zig-zag scans, entropy codings and deblocking tools are also merged with some optimization algorithms. For example, the 8x8 quantized transform coefficients are split into four 4x4 coefficients such that the original 4x4 zig-zag scan tool can be applied to the four 4x4 coefficients and the 4x4 entropy coding tool is still workable in the four 4x4 coefficients.

4. EXPERIMENTAL RESULTS

In the proposed ABT scheme, only the 8x8 forward/inverse transform units are implemented in codec. The 4x4 forward/inverse transform operations can be done by the 4x4 forward/inverse transform units involved in the 8x8 forward/inverse transform units. Thus, the 4x4 forward/inverse transform units of both encoder and decoder are saved. Like H.264, the scales matrix of core transform matrix is usually merged into the quantization [5] in AVS for reducing the computational complexity and memory cost. The merging process is derived from:

$$\begin{aligned} Y &= (C \times X \times C') \otimes E \,//\, Q(qp) \\ &= (C \times X \times C') \otimes (E \,//\, Q(qp)) = (C \times X \times C') \otimes \hat{Q}(qp) \end{aligned} \qquad (9)$$

where the notation // denotes element-by-element division and *Q* means the quantization step, which is a vector function of the quantization parameter *qp*. In formula (9), the calculation of scale matrix and quantization vector *E//Q*(*qp*) is merged into a scale and quantization (or inverse quantization) matrix $\hat{Q}(qp)$. Since the scale matrixes of the 4x4/8x8 extended transforms have such relation as (8), only the $\hat{Q}(qp)$ of 8x8 transform is stored and that of 4x4 transform is saved. Since there are no 4x8 and 8x4 transforms, their $\hat{Q}(qp)$ are be saved too. As a result, the memory of (4x4+8x4+4x8)x8x16x2 = 20480 bits are saved (the quantization step period is 8 in AVS

and the data type of scale and quantization matrixes is 16-bit).

The ABT scheme is implemented into reference software AVS-M R0. The comparisons are conducted between codec with and without ABT. Numerous sequences in QCIF and CIF formats are tested. All sequences are coded with a fixed Quantization Parameter (QP) in 20, 26, 32 and 38. The RD optimization and loopfilter are enabled, two frames are referenced and 2D-VLC is used in entropy coding. There is no B-picture in AVS-M for the consideration of complexity. Thus, the ABT scheme is applied to the P-pictures only. For fully demonstrating ABT's performances, only the first picture structure is set to I-picture and others are P-pictures. 200 frames of each sequence are coded. The results of comparison are listed in Table 1. The average luma PSNR gains and average bitrate savings of proposed ABT in various sequences are calculated using the method proposed in [6]. The average luma PSNR gains are from 0.19 to 0.69dB and the average bitrate savings are from 3.59% to 12.09%. The results imply that with the increase of picture sizes, the more PSNR gains are achieved. The RD curves of sequence *coastguard* in CIF and QCIF formats are also shown in Fig. 2. It is observed that the significant performance improvements, especially at high bitrates, are achieved in sequence *coastguard* using the proposed ABT scheme.

4. CONCLUSIONS

In this paper, an ABT scheme based on 8x8/4x4 extended transforms for AVS is proposed. The 8x8 transform in the proposed scheme is extended from the 4x4 transform of AVS-M, which has similar decorrelation ability with the real DCT while low complexity. The advantages of extended transforms used in ABT are that not only some transform units are saved in hardware but also smaller memory is required. Compared with general ABT schemes, the proposed ABT scheme based on extended transforms has the lower hardware complexities and maintain the high performance.

5. ACKNOWLEDGMENTS

This work is partially supported by the key project of National Natural Science Foundation of China under grant No. 60333020 and Natural Science Foundation of Beijing under grant No. 4041003.

6. REFERENCES

[1] Wiegand, T.; Sullivan, G.J.; Bjntegaard, G.; Luthra, A. "Overview of the H.264/AVC video coding standard," IEEE Trans. on CSVT. Special Issue on the H.264/AVC video coding standard, July, 2003.

[2] Wien, M., "Variable block-size transforms for H.264/AVC, " Special Issue on the H.264/AVC video coding standard," July, 2003.

[3] W. Chen, C. H. Smith and S. C. Fralick, "A Fast Computational Algorithm for the Discrete Cosine Transform", IEEE Transactions on Communications, vol. com-25, No. 9, pp. 1004-1009, September 1977.

[4] Jie Dong, Jian Lou, etc, "A new approach to compatible adaptive block-size transforms" VCIP2005, Beijing, Jan, 2005.

[5] Lousi Kerofsky, Shawmin Lei, "Reduced bit-depth quantization", ITU-T SG16 DOC. VCEG-N20, Aug, 2001.

[6] G. Bjontegaard, "Calculation of average PSNR differences between RD-Curves", Doc. VCEG-M33, Mar. 2001.

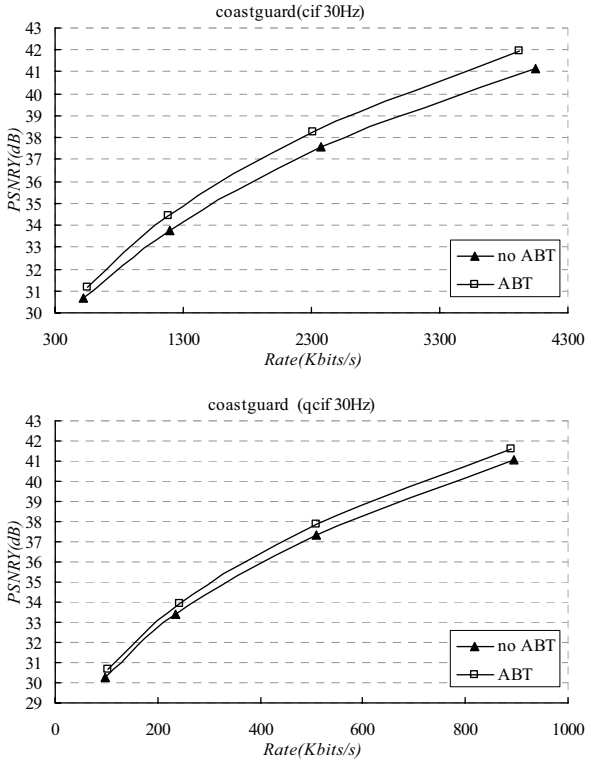

Fig.2. The RD curves for typical test sequence *coastguard* in QCIF and CIF formats.

Table 1. Comparisons of sequences coded with and without ABT scheme.

Size / Sequance	CIF(30Hz)		QCIF(30Hz)		QCIF(15Hz)	
	PSNRY gains	Bitrate savings	PSNRY gains	Bitrate savings	PSNRY gains	Bitrate savings
tempete	*0.28dB*	*5.15%*	*0.36dB*	*5.87%*	*0.35dB*	*5.98%*
mobile	*0.40dB*	*6.48%*	*0.32dB*	*5.14%*	*0.27dB*	*4.30%*
foreman	*0.23dB*	*5.11%*	*0.22dB*	*4.59%*	*0.20dB*	*4.36%*
football	*0.41dB*	*6.47%*	*0.27dB*	*3.98%*	*0.22dB*	*3.16%*
coastguard	*0.69dB*	*12.09%*	*0.44dB*	*8.45%*	*0.44dB*	*8.50%*
bus	*0.49dB*	*8.13%*	*0.33dB*	*5.03%*	*0.27dB*	*4.32%*
silent	*0.19dB*	*3.77%*	*0.22dB*	*3.59%*	*0.20dB*	*3.33%*