IEEE Int. Workshop VLSI Design & Video Tech.
Suzhou, China, May 28~30, 2005

# FAST DCT TO IT CONVERSION USING INTEGER APPROXIMATION FOR H.263 TO H.264 VIDEO TRANSCODING

*Kai-Tat Fung*, Wan-Chi Siu* and A.G. Constantinides+*

*Centre for Multimedia Signal Processing
Dept. of Electronic and Information Engg., The Hong Kong Polytechnic University, Hong Kong, China
and
+Department of Electrical Engineering
Imperial College of Science, Technology and Medicine, London, U.K.

## ABSTRACT

In this paper, a compressed domain video transcoder for H.263 to H.264 is proposed. The proposed video transcoder focuses on the conversion of DCT coefficients to Integer Transform(IT) coefficients. A set of operators is derived for converting the DCT coefficients to integer transform coefficients in the compressed domain. Using these operators, the proposed architecture is able to transcode the DCT coefficients to the integer transform coefficients directly. The integer approximation of the operators is proposed to avoid the floating point implementation for the proposed transcoder. Experimental results show that the proposed video transcoder can reduce substantially the amount of computation and provide transcoded videos with similar quality as compared with those obtained from the conventional cascaded video transcoder.

## I. INTRODUCTION

In [1], we have proposed a motion vector decomposition algorithm to speed up the motion re-estimation during the H.263 to H.264 video transcoding process[2-7]. The major concern is that the integer transform coefficients need to be re-computed[8]. In this paper, a new compressed domain video transcoder architecture is proposed to transcode the DCT coefficients to the integer transform coefficients in the compressed domain. Figure 1 shows a diagram of performing DCT coefficients to integer transform coefficients conversion for intra frame video transcoding. Figure 2 shows the conventional approach for transcoding the transformed coefficients in the pixel domain. The input is a block of 8x8 2D-DCT coefficients, A. An inverse DCT is applied to A to recover the 8x8 pixel block. The 8x8 pixel block is divided into four 4x4 blocks (pixel domain $data_i$ i=0,1,2,3). Each of the four blocks is passed to a corresponding Integer transform to generate four 4x4 blocks of transform coefficients. These four blocks of transform coefficients are combined to form a single 8x8 block. This process introduces high computational complexity as well as re-encoding errors. It is desired to perform the transcoding in the compressed domain to avoid the complete decoding and re-encoding process. Motivated by these problems, two sets of operators are derived to transcode the DCT coefficients to Integer transform coefficients. Since the operators contain floating point implementation, an integer approximation form of operators are derived for easy implementation.

The organization of this paper is as follows. Section II presents the architecture of the proposed compressed domain video transcoder, formulations of the proposed operators and their integer approximations. Experimental results are then given in Section III. Finally, some concluding remarks are provided in Section IV.

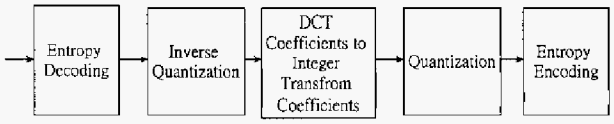


Figure 1. Intra block transcoding from H.263 to H.264/AVC

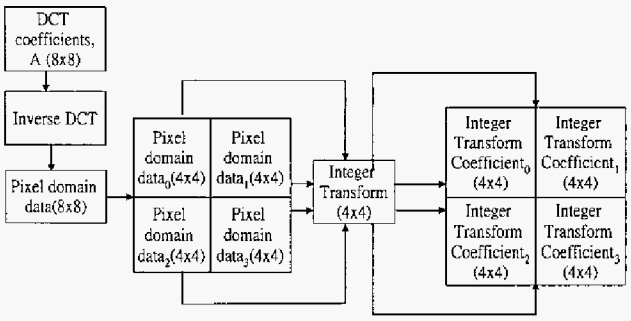


Figure 2. Conventional approach for transcoding the transformed coefficients in the pixel domain.

## II. COMPRESSED DOMAIN VIDEO TRANSCODER FOR H.263 TO H.264

The architecture of the proposed transcoder is as shown in Figure 3. The video transcoder performs an entropy decoding to extract the header information, coding mode, motion vectors and quantized DCT coefficients from the incoming bitstream. The motion vector information will be used to speed up the motion vector re-estimation process.

0-7803-9005-9/05/$20.00 ©2005 IEEE

For intra mode video transcoding, the DCT coefficients will be transcoded into the integer transform coefficients directly in the compressed domain using the proposed operators. For low processing power devices, integer approximation form of the operator will be activated for easy implementation.

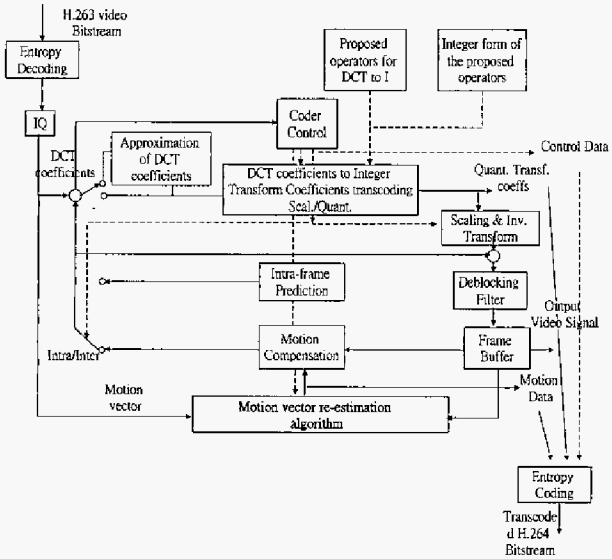


Figure 3. The proposed compressed domain video transcoder for H.263 to H.264

In Figure 2, we have shown that high computational complexity is required for transcoding the DCT coefficients to integer transform coefficients in the previous section. In this section, we construct some operators and reuse the incoming DCT coefficients such that we can transcode the integer transform coefficients in the compressed domain to reduce the computational complexity and re-encoding error. Let us recall the definition of 2D-DCT of an 8x8 block, A, as shown below,

$$DCT(A) = \hat{A} = TAT^t \qquad (1)$$

where T is an 8x8 DCT matrix with entries t(i,j) given by

$$t(i,j) = \frac{1}{2}k(i)\cos\frac{(2j+1)i\pi}{16}$$

i represents the row index, j represents the column index and

$$k(i) = \begin{cases} \frac{1}{\sqrt{2}}, & i = 0 \\ 1, & otherwise. \end{cases}$$

The Integer Transform(IT) of a 4x4 block, B, can be defined as

$$I(B) = IBI^t \otimes (\text{scaling factor1}) \qquad (2)$$

where I is a 4x4 IT matrix given by

$$I = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

$$scaling\ \ factor1 = \begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix} = \begin{bmatrix} 0.25 & 0.158 & 0.25 & 0.158 \\ 0.158 & 0.1 & 0.158 & 0.1 \\ 0.25 & 0.158 & 0.25 & 0.158 \\ 0.158 & 0.1 & 0.158 & 0.1 \end{bmatrix}$$

for a=0.5, $b = \sqrt{\frac{2}{5}}$ and $\otimes$ represents the term by term multiplication.

Similarly, the inverse transform is given by:

$$B' = J[I(B) \otimes \text{scaling factor2}]J^t \qquad (3)$$

where J is a 4x4 inverse IT matrix given by

$$J = \begin{bmatrix} 1 & 1 & 1 & 0.5 \\ 1 & 0.5 & -1 & -1 \\ 1 & -0.5 & -1 & 1 \\ 1 & -1 & 1 & -0.5 \end{bmatrix}$$

and

$$scaling\ \ factor2 = \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix} = \begin{bmatrix} 0.25 & 0.316 & 0.25 & 0.316 \\ 0.316 & 0.4 & 0.316 & 0.4 \\ 0.25 & 0.316 & 0.25 & 0.316 \\ 0.316 & 0.4 & 0.316 & 0.4 \end{bmatrix}$$

In the following, we will derive a set of equations to transform the DCT coefficients, DCT(A), into the integer transform coefficients directly in the compressed domain. A similar formulation can be obtained for transcoding the integer transform coefficients to DCT coefficients.

Applying the inverse DCT to the DCT coefficients, DCT(A), we have pixel domain data,

$$data = T^t[DCT(A)]T \qquad (4)$$

In order to satisfy the H.264 format, let us divide this reconstructed result into four groups. These four groups of data in the pixel domain can be extracted by using the proposed operators $S_{i1}$ and $S_{i2}$, i.e., the pixel domain data becomes,

$$data_i = S_{i1}T^t[DCT(A)]TS_{i2}$$

for i=0,1,2,3 (5)

where $s_{01} = s_{11} = s_{02}^T = s_{22}^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$,

$s_{21} = s_{31} = s_{12}^T = s_{32}^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

and i represents the index of the 4 groups of the integer transform coefficients.

Note that A is with a size of 8x8 and the integer transform coefficients is 4x4. Therefore, one set of DCT coefficients will give four sets of the integer transform coefficients.
Then we can apply the integer transform to the pixel domain data, $data_i$. We have

Integer transform coefficients $_i$ =I(pixel domain $data_i$) $I^t \otimes$ (scaling factor 1) (6)

Combining equations (4-6), we have

Integer transform coefficients $_i$=I($S_{i1}T^tDCT(A)TS_{i2}$)$I^t$ $\otimes$ (scaling factor 1) (7)

Let us define the pre-computed operators, $P_{1i}$=I*$S_{i1}$*$T^t$ and $P_{2i}$=T*$S_{i2}$*$I^t$,
Therefore, we have

Integer transform coefficients $_i$= $P_{1i}$*[DCT(A)]* $P_{2i}$ $\otimes$ (scaling factor 1) (8)

where

$P_{10}=P_{11}=\begin{bmatrix} 1.4142 & 1.2815 & 0 & -0.4500 & 0 & 0.3007 & 0 & -0.2549 \\ 0 & 0.9236 & 2.2304 & 1.7799 & 0 & -0.8638 & -0.1585 & 0.4824 \\ 0 & -0.1056 & 0 & 0.7259 & 1.4142 & 1.0864 & 0 & -0.5308 \\ 0 & 0.1169 & 0.1585 & -0.0922 & 0 & 1.0379 & 2.2304 & 1.9750 \end{bmatrix}$,

$P_{12}=P_{13}=\begin{bmatrix} 1.4142 & -1.2815 & 0 & 0.4500 & 0 & -0.3007 & 0 & 0.2549 \\ 0 & 0.9236 & -2.2304 & 1.7799 & 0 & -0.8638 & 0.1585 & 0.4824 \\ 0 & 0.1056 & 0 & -0.7259 & 1.4142 & -1.0864 & 0 & 0.5308 \\ 0 & 0.1169 & -0.1585 & -0.0922 & 0 & 1.0379 & -2.2304 & 1.9750 \end{bmatrix}$,

$P_{20}=P_{22}=\begin{bmatrix} 1.4142 & 0 & 0 & 0 \\ 1.2815 & 0.9236 & -0.1056 & 0.1169 \\ 0 & 2.2304 & 0 & 0.1585 \\ -0.4500 & 1.7799 & 0.7259 & -0.0922 \\ 0 & 0 & 1.4142 & 0 \\ 0.3007 & -0.8638 & 1.0864 & 1.0379 \\ 0 & -0.1585 & 0 & 2.2304 \\ -0.2549 & 0.4824 & -0.5308 & 1.9750 \end{bmatrix}$

and $P_{21}=P_{23}=\begin{bmatrix} 1.4142 & 0 & 0 & 0 \\ -1.2815 & 0.9236 & 0.1056 & 0.1169 \\ 0 & -2.2304 & 0 & -0.1585 \\ 0.4500 & 1.7799 & -0.7259 & -0.0922 \\ 0 & 0 & 1.4142 & 0 \\ -0.3007 & -0.8638 & -1.0864 & 1.0379 \\ 0 & 0.1585 & 0 & -2.2304 \\ 0.2549 & 0.4824 & 0.5308 & 1.9750 \end{bmatrix}$

Note that $P_{1i}$ and $P_{2i}$ can be pre-computed. Therefore, low computational complexity can be achieved since the transformation is done in the compressed domain. In other words, no complete decoding and re-encoding process are required to obtain the integer transform coefficients from DCT coefficients.

In the above formulation, we have derived a set of operators to transcode the DCT coefficients to integer transform coefficients. The major concern is that floating point operations are more expensive to implement than integer operations in low processing power devices. Motivated by this, it is beneficial to make an approximation form of the derived operators in the previous subsection. It is obvious that we can multiply each of the proposed operators by a large integer number such that the proposed operators become integers. The resulting coefficients should be scaled down by the same factor. If the factor is a power of 2, the downscaling of the results can be done by pure shifting. Furthermore, the shifting operations can also be absorbed in the quantization process during the transcoding; i.e. no extra operations are required in this approximation process. Therefore, it is good to choose a larger integer with power of 2 to achieve good accuracy. In general, 32 bit arithmetic is the capability of most signal processors. For DCT to integer transform conversion, the DCT coefficients have a dynamic range of 4096(=256x16). Hence a wordlength of 12 bits is required to represent it. The maximum sum of absolute values of our proposed operators is 6.44. So, the maximum dynamic range gain for the 2D operator is 41.47(=$6.44^2$) which requires 6 bits to represent it. The scaling factor for which we can select is the square root of $2^{32-6-12}=2^7$, i.e.128. Hence, the approximation form of the proposed operators becomes:

$P_{10}{}^I=P_{11}{}^I=\begin{bmatrix} 181 & 164 & 0 & -58 & 0 & 38 & 0 & 33 \\ 0 & 118 & 285 & 228 & 0 & -111 & -20 & 62 \\ 0 & -14 & 0 & 93 & 181 & 139 & 0 & -68 \\ 0 & 15 & 20 & -12 & 0 & 133 & 285 & 253 \end{bmatrix}$,

$P_{12}{}^I=P_{13}{}^I=\begin{bmatrix} 181 & -164 & 0 & 58 & 0 & -38 & 0 & 33 \\ 0 & 118 & -285 & 228 & 0 & -111 & 20 & 62 \\ 0 & 14 & 0 & -93 & 181 & -139 & 0 & 68 \\ 0 & 15 & -20 & -12 & 0 & 133 & -285 & 253 \end{bmatrix}$,

$P_{20}{}^I=P_{22}{}^I=\begin{bmatrix} 181 & 0 & 0 & 0 \\ 164 & 118 & -14 & 15 \\ 0 & 285 & 0 & 20 \\ -58 & 228 & 93 & -12 \\ 0 & 0 & 181 & 0 \\ 38 & -111 & 139 & 133 \\ 0 & -20 & 0 & 285 \\ -33 & 62 & -68 & 253 \end{bmatrix}$

and $P_{21}{}^I=P_{23}{}^I=\begin{bmatrix} 181 & 0 & 0 & 0 \\ -164 & 118 & 14 & 15 \\ 0 & -285 & 0 & -20 \\ 58 & 228 & -93 & -12 \\ 0 & 0 & 181 & 0 \\ -38 & -111 & -139 & 133 \\ 0 & 20 & 0 & -285 \\ 33 & 62 & 68 & 253 \end{bmatrix}$

## III. EXPERIMENTAL RESULTS

Extensive simulations have been done with reference to a cascaded pixel-domain transcoder. These experiments have been performed to evaluate the overall efficiency of the proposed compressed domain video transcoding for H.263 to H.264. The cascaded transcoder was implemented using an H.263 video codec and an H.264 video codec. In the front encoder, the first frame of a sample sequence was encoded as an intraframe (I-frame), and the remaining frames were encoded as interframes (P-frames). Results of our experimental work show that the proposed video transcoders outperform the conventional cascaded video transcoder in terms of computational complexity as shown in Table 1. Quality degradation is used to indicate the case using the approximation of DCT coefficients[1] and the integer approximation of operators in the proposed video transcoder as compared with the conventional cascaded video transcoder. The speed up ratio is defined as the time for transcoding integer transform coefficients using the proposed transcoder with the approximation of DCT coefficients as compared with the time for conventional cascaded video transcoder. Theoretically, the speed up ratio for transcoding the DCT coefficients to Integer transform coefficients is about 2.8 times. Since our proposed operators have many zero values, 3.15 times speed up ratio can be achieved when these redundant operations are eliminated. If the approximation of DCT coefficients is used, the proposed video transcoding architecture can achieve a speed up of about 3.6 to 8.4 times for transcoding the DCT coefficients to integer transform coefficients. The results are more significant for sequences with low motion activities using the approximation form of the DCT coefficients for fast computation since only a few DCT coefficients are needed to be transcoded.

## IV. CONCLUSION

In this paper, a new compressed domain video transcoder for H.263 to H.264 is proposed. We have derived two sets of operators to transcode the DCT coefficients into integer transform coefficients in the compressed domain. The integer approximation of the operators is proposed to avoid the floating point implementation for the proposed transcoder. The proposed architecture can combine with any existing fast searching algorithm for multiple reference frames and variable block sizes estimation to speed up the transcoding process. Results of our experimental work show that the proposed architecture produces similar pictures quality with low computational complexity as compared with the conventional video transcoder.

## V. ACKNOWLEDGMENTS

This work is supported by the Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, Hong Kong Polytechnic University and the Research Grant Council of the Hong Kong SAR Government (PolyU 5234/03E). K.T. Fung acknowledges the research studentships provided by the same University.

Table 1. Performance of the proposed transcoder, where the frame rate of the incoming bitstream was 30 frames/s. H.263 was used as the front encoder for encoding "Salesman", "Miss_America", "Hall", "Tennis", "Football" and "Flower".

| Sequences | Input bitrate | Quality degradation | Speed up ratio |
|---|---|---|---|
| Salesman (352x288) | 512k | 0.05 | 8.2 |
| | 256k | 0.06 | 8.4 |
| Miss_ America (352x288) | 512k | 0.09 | 7.9 |
| | 256k | 0.1 | 8.0 |
| Hall (352x288) | 512k | 0.14 | 6.9 |
| | 256k | 0.17 | 7.1 |
| Tennis (352x240) | 3M | 0.22 | 6.3 |
| | 1.5M | 0.27 | 6.4 |
| Flower (352x240) | 3M | 0.29 | 4.4 |
| | 1.5M | 0.32 | 4.5 |
| Football (352x240) | 3M | 0.33 | 3.6 |
| | 1.5M | 0.37 | 3.7 |

## VI. REFERENCES

[1] Kai-Tat Fung and Wan-Chi Siu, "Low-Complexity H.263 to H.264 video transcoding using motion vector decomposition," paper accepted, to be published in the 2005 IEEE International Symposium on Circuits and Systems (ISCAS 2005).

[2] Video Coding for Low Bitrate Communication, ITU-T Recommendation H.263, May 1997.

[3] Jeongnam Youn, Ming-Ting Sun and Chia-Wen Lin, "Motion vector refinement for high-performance transcoding," IEEE Transactions on Multimedia, Vol.1, pp.30 - 40, March 1999

[4] G. Keeman, R. Hellinghuizen, F. Hoeksema and G. Heideman, "Transcoding of MPEG bitstreams," Signal Processing: Image Communication, vol. 8, pp. 481-500, Sept. 1996.

[5] Shanableh, T.; Ghanbari, M., "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats," IEEE Transactions on Multimedia, vol.2, pp.101-110, June 2000.

[6] Bo Shen, Ishwar K. Sethi and Bhaskaran Vasudev, "Adaptive motion-vector resampling for compressed video downscaling," IEEE Transactions on circuit and systems for video technology, vol.9, no.6, September 1999.

[7] H. Sun, W. Kwok and J.W. Zdepski, "Architectures for MPEG compressed bitstream scaling," IEEE Trans. Circuits Syst. Video Technol., vol. 6, pp. 191-199, Apr. 1996.

[8] Wiegand,T., Sullivan, G.J., Bjntegaard, G. and Luthra, A., " Overview of the H.264/AVC video coding standard" IEEE Transactions on Circuits and Systems for Video Technology, Vol.13, No.7, pp.560- 576, July 2003.