The Technique of Pre-Scaled Integer Transform

Ci-Xun Zhang, Jian Lou, Lu Yu, Jie Dong
Institute of Information and Communication Engineering
Zhejiang University
Hangzhou, China
cixunzhang@hotmail.com, yul@zju.edu.cn

Wai-Kuen Cham
Department of Electronic Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong, China
wkcham@ee.cuhk.edu.hk

***Abstract*—Integer Cosine Transform (ICT) is adopted by H.264/AVC for its bit-exact implementation and significant complexity reduction compared to Discrete Cosine Transform (DCT) with an impact in peak signal-to-noise ratio (PSNR) of less than 0.02dB. In this paper, a new technique, named Pre-Scaled Integer Transform (PIT), is proposed. With PIT, the implementation complexity is further reduced compared to conventional ICT, especially for low-end processors while all the merits of ICT are kept. Extensive experiments show that no obvious penalty in performance is observed but rather slightly gain in PSNR is obtained by using PIT when the integer transform matrix used meets certain requirements.**

I. INTRODUCTION

Integer cosine transform (ICT) was first introduced by W. K. Cham in 1989 [1] and is further developed in recent years. It has been proved that some ICTs have almost the same compression efficiency as the Discrete Cosine Transform (DCT) but much simpler implementation because only additions and shifts operations are needed [2]. Moreover, ICT can avoid inverse transform mismatch problems of DCT. Due to these advantages, the latest international video coding standard H.264/AVC adopted order-4 and 8 ICT [3] [4].

In this paper, a technique called Pre-Scaled Integer Transform (PIT) is proposed. The proposed technique can further reduce the implementation complexity of ICT while no penalty in performance is observed. This paper is organized as follows. In Section II, a brief review on conventional ICT scheme is given. Then, the technique of PIT is examined in Section III. The conclusion is given in Section IV.

II. CONVENTIONAL CODING SCHEME

A. Review on Discrete Cosine Transform

The 2-D Discrete cosine transform (DCT) for n×m block signals is defined as

$$F_{n\times m} = DCT_{n\times n} \times f_{n\times m} \times DCT^{T}_{m\times m}, \quad (1)$$

where $\mathbf{f_{n\times m}}$ and $\mathbf{F_{n\times m}}$ stand for the signals in the spatial and spectrum domain respectively while $\mathbf{DCT_{n\times n}}$ and $\mathbf{DCT_{m\times m}}$ are transform matrices.

B. Foundamentals of Integer Cosine Transform

ICT originates from the DCT and was derived using the principle of dyadic symmetry [1]. An ICT matrix can be represented as a matrix product of a diagonal normalization matrix and an integer transform matrix which contains only integers. In this paper, $\mathbf{ICT_{n\times n}}$ represents the n×n integer transform matrix. The counterpart of (1) for ICT is

$$F_{n\times m} = (ICT_{n\times n} \times f_{n\times m} \times ICT^{T}_{m\times m}) \otimes S_{n\times m}, \quad (2)$$

where

$$S_{n\times m} = S_{n\times 1} \times S_{1\times m}, \quad (3)$$

and $A_{n\times m} = B_{n\times m} \otimes C_{n\times m}$ means

$$A_{n\times m}(i,j) = B_{n\times m}(i,j) \times C_{n\times m}(i,j) \quad 0 \le i < n, 0 \le j < m. \quad (4)$$

$\mathbf{S_{n\times m}}$, as defined in (3), is called the scaling matrix which is computed from $\mathbf{S_{n\times 1}}$ and $\mathbf{S_{1\times m}}$ as follows,

$$S_{n\times 1}(i) = \frac{1}{\sqrt{\sum_{j=0}^{n} (ICT_{n\times n}(i,j))^2}} \qquad 0 \le i \le n, \quad (5)$$

$$S_{1\times m}(j) = \frac{1}{\sqrt{\sum_{i=0}^{m} (ICT^{T}_{m\times m}(i,j))^2}} \qquad 0 \le j \le m. \quad (6)$$

This research is sponsored by NSFC under contract No. 60333020 and No. 90207005.

Then, quantization is applied to all transformed coefficients, i.e.

$$QF_{n\times m} = F_{n\times m} // Q_{n\times m}(QP), \quad (7)$$

where $\mathbf{Q_{n\times m}(QP)}$ is the quantization matrix with a quantization parameter QP. Here, $A_{n\times m} = B_{n\times m} // C_{n\times m}$ means

$$A_{n\times m}(i,j) = B_{n\times m}(i,j) / C_{n\times m}(i,j) \quad 0 \le i < n, 0 \le j < m. \quad (8)$$

Hence, the forward transform and quantization can be expressed as

$$QF_{n\times m} = (ICT_{n\times n} \times f_{n\times m} \times ICT^T_{m\times m}) \otimes S_{n\times m} // Q_{n\times m}(QP). \quad (9)$$

The inverse quantization and inverse transform can be written as

$$f'_{n\times m} = ICT^T_{n\times n} \times (QF_{n\times m} \otimes Q_{n\times m}(QP) \otimes S'_{n\times m}) \times ICT_{m\times m}, \quad (10)$$

where $\mathbf{f'_{n\times m}}$ is the reconstructed signal in the spatial domain and $\mathbf{S'_{n\times m}}$ is the inverse scaling matrix which can be derived from equation (3), (5) and (6) by replacing $\mathbf{ICT_{n\times n}}$ and $\mathbf{ICT^T_{m\times m}}$ with $\mathbf{ICT^T_{n\times n}}$ and $\mathbf{ICT_{m\times m}}$ in (5) and (6) respectively.

The whole process of forward transform and inverse transform can be represented as

$$f'_{n\times m} = ICT^T_{n\times n} \times (((ICT_{n\times n} \times f_{n\times m} \times ICT^T_{m\times m}) \otimes S_{n\times m} // Q_{n\times m}(QP)) \otimes Q_{n\times m}(QP) \otimes S'_{n\times m}) \times ICT_{m\times m}. \quad (11)$$

In (11), let

$$QS_{n\times m}(QP) = S_{n\times m} // Q_{n\times m}(QP), \quad (12)$$

$$IQS_{n\times m}(QP) = S'_{n\times m} \otimes Q_{n\times m}(QP), \quad (13)$$

substituting (12) and (13) into (11), we can get

$$f'_{n\times m} = ICT^T_{n\times n} \times (((ICT_{n\times n} \times f_{n\times m} \times ICT^T_{m\times m}) \otimes QS_{n\times m}(QP)) \otimes IQS_{n\times m}(QP)) \times ICT_{m\times m}. \quad (14)$$

Equation (14) means that the forward/inverse scaling and the quantization/de-quantization are combined. This is exactly what H.264/AVC does to reduce the computational complexity. Fig.1 is the block diagram for the conventional ICT coding scheme in H.264.

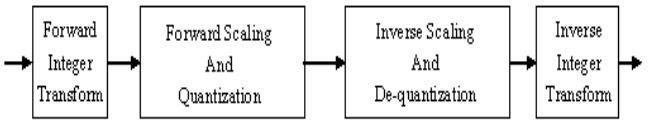

Figure 1. Block diagram of conventional ICT scheme in H.264

III. THE TECHNIQUE OF PRE-SCALED INTEGER TRANSFORM

A. Pre-Scaled Integer Transform

The main problem with the conventional ICT scheme shown in section II is that a memory of 6×4×4=96 bytes should be allocated to store the dequantization matrix $\mathbf{IQS_{n\times m}(QP)}$ in decoder if we fully expand it to avoid lookup operations in order to facilitate parallel processing and take advantage of the efficient multiply/accumulate architecture of many processors. (Otherwise, every transformed coefficient requires a 3-D lookup operation, which uses QP and the coordinates of the coefficient in the block, to search for the corresponding dequantization coefficient. In this situation, the computational burden will increase a lot.) In fact, this memory size can be reduced without increasing computational complexity if we change the order of the inverse scaling and as a result (11) can be rewritten as

$$f'_{n\times m} = ICT^T_{n\times n} \times (((ICT_{n\times n} \times f_{n\times m} \times ICT^T_{m\times m}) \otimes S_{n\times m} \otimes S'_{n\times m} // Q_{n\times m}(QP)) \otimes Q_{n\times m}(QP)) \times ICT_{m\times m}. \quad (15)$$

Let

$$QSS_{n\times m}(QP) = S_{n\times m} \otimes S'_{n\times m} // Q_{n\times m}(QP), \quad (16)$$

then (15) can be simplified as

$$f'_{n\times m} = ICT^T_{n\times n} \times (((ICT_{n\times n} \times f_{n\times m} \times ICT^T_{m\times m}) \otimes QSS_{n\times m}(QP)) \otimes Q_{n\times m}(QP)) \times ICT_{m\times m}. \quad (17)$$

Equation (17) means that the inverse scaling is moved to encoder side and combined with forward scaling and

quantization as one single process. This scheme is known as Pre-Scaled Integer Transform (PIT). The block diagram for the PIT coding scheme is shown below.

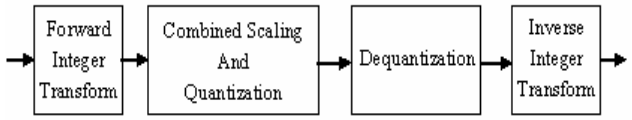

Figure 2. Block diagram of proposed PIT scheme

Compared with conventional ICT scheme in H.264, when PIT is used, implementation complexity can be reduced on decoder side. First, if the dequantization matrix is fully expanded, a memory of 6×4×4=96 bytes can be saved if only order-4 ICT is used and a memory of only 6 bytes (the QP period is 6 in H.264) is needed instead. Otherwise, if the dequantization matrix is not expanded, the computational complexity will be much lower because 3-D lookup operation is replaced by 1-D lookup operation, which uses QP only. In either case, on encoder side, since forward scaling, inverse scaling and quantization are combined as one single process, both the computational and storage complexity remains unchanged. At the time of the writing of this paper, the H.264/AVC standardization work was still in progress and 8×8 ICT has been adopted in Fidelity Range Extensions (FRExt) of H.264/AVC [4]. When 8×8 ICT is concerned, more complexity can be reduced with PIT because a memory of 6×8×8=384 bytes on decoder side can be replaced by a memory of only 6 bytes. Besides H.264, in some other video coding standards, e.g., AVS (AVS is abbreviation of Audio Video coding Standard, Chinese national coding standard aiming at applications of digital TV Broadcasting and high-definition DVD.), the scheme of periodic QP is not used. In this case, the saving of memory will be significantly larger. Only a memory of 64 bytes is needed instead of a memory of 64×8×8=4096 bytes. (The QP range is 0~63 and 8×8 ICT is employed in AVS. In order to save memory, the scaling and quantization/dequantization are separated in AVS. Even in this case, a memory of 8×8=64 bytes can be saved and at the same time the computational complexity is also reduced when PIT is used because no scaling is needed on decoder side any more.) From the analysis above, it is clear that the scheme of PIT is superior to the conventional ICT scheme for its significant complexity reduction, especially for low-end processors while no performance penalty is observed but rather a slightly gain in PSNR can be obtained which will be shown later in this paper.

B. Requirements of the integer transform matrix

It should be noticed that when the technique of PIT is used, the integer transform matrix itself should meet certain requirements. This is because when the inverse scaling is moved to the encoder side, an implicit frequency weighting matrix (i.e., $\mathbf{S'_{n\times m}}$) is applied to the transformed signals. In order not to change too much of the energy distribution of the signals in the spectrum domain, the coefficients of the frequency weighting matrix $\mathbf{S'_{n\times m}}$ should be closed to each other. From (3), (5), (6), it can be deduced that the norms of the column vectors of $\mathbf{ICT^T_{n\times n}}$ should be closed to each other. We define similarity factor η to represent the similarity of the norms of the column vectors of $\mathbf{ICT^T_{n\times n}}$.

$$\eta=\frac{\max\limits_{0\le i<n,0\le j<m}(S'_{n\times m}(i,j))-\min\limits_{0\le i<n,0\le j<m}(S'_{n\times m}(i,j))}{\max\limits_{0\le i<n,0\le j<m}(S'_{n\times m}(i,j))}. \quad (18)$$

C. Experimental results and analysis

In order to test the performance of PIT, extensive experiments have been done based on the JM7.6 platform. Five different inverse transform matrices are used in our experiments, namely, $\mathbf{T_1}$, $\mathbf{T_2}$, $\mathbf{T_3}$, $\mathbf{T_4}$, $\mathbf{T_5}$ (with the transform matrices $\mathbf{T_1^T}$, $\mathbf{T_2^T}$, $\mathbf{T_3^T}$, $\mathbf{T_4^T}$, $\mathbf{T_5^T}$ respectively, here the superscript T denotes transposition.):

$$T_1=\begin{pmatrix}1&2&1&1\\1&1&-1&-2\\1&-1&-1&2\\1&-2&1&-1\end{pmatrix}\quad T_2=\begin{pmatrix}2&3&2&1\\2&1&-2&-3\\2&-1&-2&3\\2&-3&2&-1\end{pmatrix}\quad T_3=\begin{pmatrix}4&5&4&2\\4&2&-4&-5\\4&-2&-4&5\\4&-5&4&-2\end{pmatrix}$$

$$T_4=\begin{pmatrix}7&9&7&4\\7&4&-7&-9\\7&-4&-7&9\\7&-9&7&-4\end{pmatrix}\quad T_5=\begin{pmatrix}13&17&13&7\\13&7&-13&-17\\13&-7&-13&17\\13&-17&13&-7\end{pmatrix}$$

All the implementations are in 16-bit. And the five transform matrices chosen in our experiment allow both the efficient shift/add implementation and a separable matrix-multiply implementation which can take advantage of the efficient multiply/accumulate architecture of many processors that produces identical results. In fact, $\mathbf{T_1^T}$ is used in H.264 now and $\mathbf{T_5^T}$ is once proposed in H.264. $\mathbf{T_2^T}$, $\mathbf{T_3^T}$, $\mathbf{T_4^T}$ are proposed in AVS-M. (M is for Mobility, AVS-M aims at applications of video communications on mobile devices.) They have different similarity factors and were chosen to demonstrate how similarity factor affects the performance when PIT is used. The experimental results are given in Table I in the form of average PSNR gains with equal bit rates using the method proposed in [5]. The entropy coding method is CABAC. Four QP points of 28, 32, 36, 40 are used for H.264 (anchor), $\mathbf{T_3}$, $\mathbf{T_4}$, $\mathbf{T_5}$ while four QP points of 27, 31, 35, 39 for $\mathbf{T_1}$, $\mathbf{T_2}$ in order to match the bit rates of H.264. The similarity factors of the matrices are also calculated and given. Finally, to further illustrate the coding efficiency of different transform matrices, two graphs, Fig. 3 and Fig. 4 are presented at the end of this section to show rate distortion curves for two sequences mobile and news.

The experimental results show that the similarity factor of the selected inverse integer transform matrix should be kept not larger than 0.2 without noticeable loss in performance. In most cases, the smaller the similarity factor is, the better the performance is. The only exception is that $\mathbf{T_4}$ gives more gain in PSNR than $\mathbf{T_5}$ although $\eta_{T_4}>\eta_{T_5}$. This is because the errors caused by the right shifts when

using $\mathbf{T_5}$ is larger than that when using $\mathbf{T_4}$. It may be also inferred from these experiment results that $\mathbf{T_4}$ yields the best performance compared to others for its small coefficient similarity factor and small errors caused by right shifts.

PIT has been adopted in AVS because of its excellent performance and significant complexity reduction with the order-8 matrix

$$T = \begin{pmatrix} 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 10 & 9 & 6 & 2 & -2 & -6 & -9 & -10 \\ 10 & 4 & -4 & -10 & -10 & -4 & 4 & 10 \\ 9 & -2 & -10 & -6 & 6 & 10 & 2 & -9 \\ 8 & -8 & -8 & 8 & 8 & -8 & -8 & 8 \\ 6 & -10 & 2 & 9 & -9 & -2 & 10 & -6 \\ 4 & -10 & 10 & -4 & -4 & 10 & -10 & 4 \\ 2 & -6 & 9 & -10 & 10 & -9 & 6 & -2 \end{pmatrix}.$$

IV. CONCLUSION

In this paper, a new technique, named Pre-Scaled Integer Transform (PIT), is proposed to further reduce the implementation complexity compared to conventional Integer Cosine Transform (ICT) scheme in H.264, especially for low-end processors while all the merits of ICT are kept. With PIT, the larger the transform size (e.g., 8×8) is, the more significant reduction of storage and computational complexity can be achieved. Extensive experiments show that no noticeable penalty in performance is introduced but rather slightly gain in PSNR is obtained with PIT when the integer transform matrix used meets certain requirements.

TABLE I. PERFORMANCE OF DIFFERENT INTEGER TRANSFORM MATRICES USING PIT COMPARED WITH H.264/AVC

PSNR Gain (dB)	T_1	T_2	T_3	T_4	T_5
bus_cif_30fps	-0.417	-0.056	-0.050	0.013	-0.014
bus_cif_15fps	-0.455	-0.091	-0.073	-0.003	-0.031
bus_qcif_15fps	-0.478	-0.044	-0.054	-0.010	-0.020
bus_qcif_5fps	-0.626	-0.099	-0.098	-0.002	-0.014
football_cif_30fps	-0.296	-0.073	-0.063	-0.004	-0.031
football_cif_15fps	-0.330	-0.104	-0.071	-0.005	-0.027
football_qcif_15fps	-0.358	-0.083	-0.063	-0.016	-0.027
football_qcif_5fps	-0.384	-0.081	-0.048	0.015	0.002
foreman_cif_30fps	-0.361	-0.048	-0.019	0.020	0.002
foreman_cif_15fps	-0.362	-0.040	-0.005	0.029	-0.003
foreman_qcif_15fps	-0.267	-0.043	-0.035	0.043	-0.014
foreman_qcif_5fps	-0.292	-0.071	-0.045	0.010	0.015
mobile_cif_30fps	-0.403	-0.064	-0.070	0.008	-0.001
mobile_cif_15fps	-0.392	-0.070	-0.074	0.013	0.017
mobile_qcif_15fps	-0.388	-0.036	-0.068	0.029	0.019
mobile_qcif_5fps	-0.383	-0.049	-0.085	-0.001	0.013
news_cif_30fps	-0.670	-0.069	-0.014	0.045	0.027
news_cif_15fps	-0.578	-0.046	-0.007	0.052	0.031
news_qcif_15fps	-0.674	-0.102	-0.056	0.004	-0.028
news_qcif_5fps	-0.675	-0.138	-0.034	0.021	-0.015
paris_cif_30fps	-0.725	-0.061	-0.024	-0.004	0.007
paris_cif_15fps	-0.717	-0.083	-0.017	0.008	0.020
paris_qcif_15fps	-0.595	-0.043	-0.029	0.006	0.034
paris_qcif_5fps	-0.607	-0.010	-0.000	0.043	0.050
tempete_cif_30fps	-0.374	-0.025	-0.038	0.014	0.008
tempete_cif_15fps	-0.332	-0.010	-0.042	0.014	0.008
tempete_qcif_15fps	-0.413	-0.058	-0.037	0.002	-0.003
tempete_qcif_5fps	-0.434	-0.074	-0.058	-0.007	-0.004
average PSNR gain	-0.464	-0.063	-0.046	0.012	0.001
similarity factor(η)	0.6000	0.2000	0.0938	0.0051	0

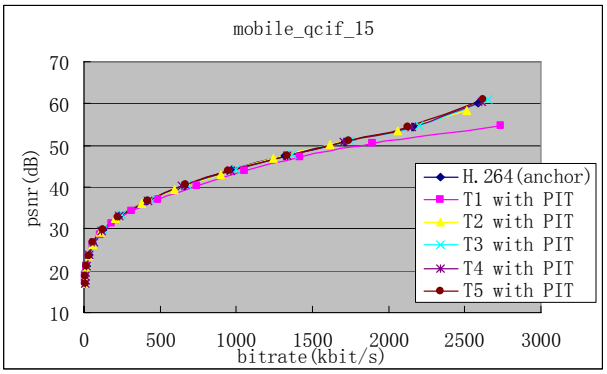

Figure 3. Rate-distortion curves of different integer transform matrices using the technique of PIT for mobile sequence (QCIF, 15fps)

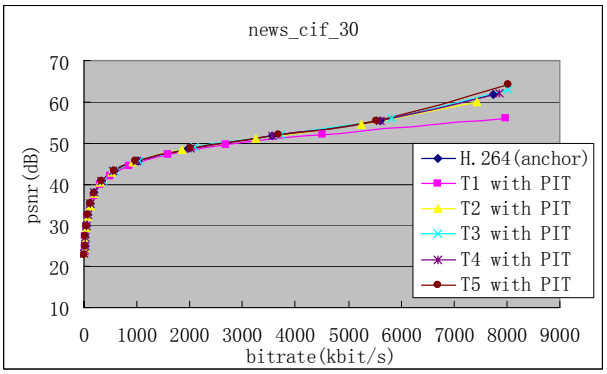

Figure 4. Rate-distortion curves of different integer transform matrices using the technique of PIT for news sequence (CIF, 30fps)

REFERENCES

[1] Cham, W.K., "Development of integer cosine transforms by the principle of dyadic symmetry," Proc. IEE, I, 136, (4), pp. 276-282, 1989.

[2] H. Malvar, A. Hallapuro, M. Karczwicz, and L. Kerofsky, "Low-complexity transform and quantization in H.264/AVC," IEEE Trans. circuits Syst. Video Technol., vol. 13, pp. 598-603, July 2003.

[3] T. Wiegand, G. J. Sullivan, G. BjØntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE Trans. Circuits Syst. Video Technol., vol. 13, pp. 560-576, July 2003.

[4] Gary J. Sullivan, Pankaj N. Topiwala, and Ajay Luthra, "The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions," Proc. SPIE Int. Soc. Opt. Eng. 5558, Aug 2004.

[5] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," ITU-T SG16 Doc. VCEG-M33, 2001.