Integer Sinusoidal Transforms Based On Lifting Factorization

Yonghong Zeng, Guoan Bi and Zhiping Lin
School of Electrical and Electronic Engineering
Nanyang Technological University, Singapore

Abstract—**A general method is proposed to factor discrete W transform (DWT) into lifting steps and additions. Then, based on the relationships among various types of discrete sinusoidal transforms, other types of transforms such as discrete Fourier transform (DFT) and discrete cosine transform (DCT) are factored into lifting steps and additions. After approximating the lifting matrices, we get various types of new integer discrete transforms such as IntDWT, IntDFT and IntDCT which are floating-point multiplication free. Transforms which map integer to integer are also proposed. Fast algorithms are given for the new transforms and their computational complexities are analyzed. Based on polynomial transform and index mapping, multi-dimensional integer transforms are presented with especially low computational complexity.**

Key words. Discrete sinusoidal transform, Integer Transform, Fast Algorithm, Mobile, Data Compression

I. Introduction

Discrete sinusoidal transforms have a wide range of applications such as data compression, feature extraction, multi-frame detection and filter bank [1], [7], [11], [12]. However, floating-point multiplications are inevitable for implementing such transforms, which prevents them from being widely used in areas such as mobile devices and lossless compression. In mobile devices, the power consumption used for computation, especially for floating-point multiplications, can not be neglected. Since there exists errors for quantizing the transforming coefficients, it is impossible to use them for lossless compression. Therefore it is not surprising that lossless coding schemes are hardly based on the discrete sinusoidal transforms. Generally speaking, integer transforms possess some features of their corresponding floating-point transforms such as the de-correlation property, while their computational cost is less. Since integer transforms require only integer arithmetic (additions and possibly multiplications), their implementation is greatly simplified. Therefore, they have been applied for image coding, filter bank and other areas [5], [8], [10], [14].

Lifting matrix is a major tool for constructing integer wavelet and integer discrete transforms [3], [5], [13], [14]. We give its definition and some simple properties in the following.

Definition 1: A lifting matrix is a matrix whose diagonal elements are 1's and only one non-diagonal element is nonzero [3], [13]. If the order of a lifting matrix is N, we use the notation $L_{i,j}(s)$ $(i \neq j)$ to denote the lifting matrix whose only nonzero element is at the ith row and the jth column $(i, j = 0, 1, \cdots, N-1)$ and whose non-diagonal nonzero element is s. A lifting step is multiplying a lifting matrix with a vector.

A distinguished feature of lifting matrix is that its inverse is still a lifting matrix with the same shape. In fact, we have

$$L_{i,j}^{-1}(s) = L_{i,j}(-s). \tag{1.1}$$

So, if a matrix can be factored into products of lifting matrices, its inverse is also products of lifting matrices. For a lifting step, floating-point multiplication is needed if s is an irrational number or even a rational number with unlimited digits. In this case, we should approximate s by another number. For easy realization, the number is desired to be of the form $\beta/2^\lambda$, where β and λ are integers.

Definition 2: The notation $\mathrm{RB}(s)$ is used to denote a number that is of the form $\beta/2^\lambda$ (dyadic rational number) and approximates to the real number s.

When s is approximated by $\mathrm{RB}(s)$, the matrix $L_{i,j}(s)$ is then approximated by $L_{i,j}(\mathrm{RB}(s))$ which is still invertible and whose inverse is $L_{i,j}(-\mathrm{RB}(s))$. The transform $y = L_{i,j}(s)x$ is then approximated by $\bar{y} = L_{i,j}(\mathrm{RB}(s))x$. Still we can reconstruct x from $\bar{y}$ by $x = L_{i,j}(-\mathrm{RB}(s))\bar{y}$. We can also approximate it by a non-linear transform. For example, it can be approximated by a transform defined as:

$$\hat{y}(i) = x(i) + \lfloor sx(j) \rfloor,\ \hat{y}(k) = x(k),\ k \neq i. \tag{1.2}$$

This transform is non-linear! Also it maps integer into integer! The non-linear transform is invertible and its inverse is as follows:

$$x(i) = \hat{y}(i) - \lfloor s\hat{y}(j) \rfloor,\ x(k) = \hat{y}(k),\ k \neq i. \tag{1.3}$$

Therefore, when a transform is factored into lifting steps, it is easy to approximate it by another transform which needs no floating-point multiplications or even by an integer to integer non-linear transform. Furthermore, the resultant transform is invertible and its inverse needs no floating-point multiplications as well.

Theoretically, any matrix with determinant 1 can be factored into products of some lifting matrices [3], [13]. However, for a given class of matrices, such as the transform matrices, it is still very difficult to find a general factorization of lifting matrices for them. In this paper, we will propose a unified method that can give the factorization of any discrete sinusoidal transform matrix with order $N = 2^t$.

II. The factorization of DWT-IV

Let $x(n)$ $(n = 0, 1, \cdots, N-1)$ be a real input sequence. We assume that $N = 2^t$ where $t > 0$. The scaled DWT-IV of $x(n)$ [16] is defined as follows:

$$X(k) = \sum_{n=0}^{N-1} x(n) \mathrm{cas} \frac{\pi(2n+1)(2k+1)}{2N},\ k = 0, 1, \cdots, N-1, \tag{2.1}$$

where function cas is defined by $\mathrm{cas}(u) = \cos(u) + \sin(u)$. Let W_N^{IV} be the transforming matrix of the DWT-IV, that is,

$$W_N^{IV} = (\mathrm{cas}\frac{\pi(2k+1)(2n+1)}{2N})_{k,n=0,1,\cdots,N-1}. \tag{2.2}$$

A. The factorization of DWT-IV

In order to factor the transforming matrix, we first derive a simple fast algorithm in the following. Let

$$H(k) = [X(2k)+X(2k+1)]/2,\ G(k) = [X(2k+1)-X(2k)]/2,$$

$$k = 0, 1, \cdots, N/2-1.$$

Based on trigonometric formulae, we have

$$\begin{aligned} H(k) &= \sum_{n=0}^{N-1} x(n)\cos\tfrac{\pi(2n+1)}{2N}\mathrm{cas}\tfrac{\pi(2n+1)(2k+1)}{N} \\ &= \sum_{n=0}^{N/2-1}[x(n)\cos\tfrac{\pi(2n+1)}{2N} + x(\tfrac{N}{2}+n)\sin\tfrac{\pi(2n+1)}{2N}] \\ &\cdot\mathrm{cas}\tfrac{\pi(2n+1)(2k+1)}{N}. \end{aligned}$$

$$\begin{aligned} G(k) &= \sum_{n=0}^{N-1} x(n)\sin\tfrac{\pi(2n+1)}{2N}\mathrm{cas}[-\tfrac{\pi(2n+1)(2k+1)}{N}] \\ &= \sum_{n=0}^{N/2-1}[x(n)\sin\tfrac{\pi(2n+1)}{2N} - x(\tfrac{N}{2}+n)\cos\tfrac{\pi(2n+1)}{2N}] \\ &\cdot\mathrm{cas}[-\tfrac{\pi(2n+1)(2k+1)}{N}]. \end{aligned}$$

Let

$$h(n) = x(n)\cos\frac{\pi(2n+1)}{2N} + x(\frac{N}{2}+n)\sin\frac{\pi(2n+1)}{2N}, \tag{2.3}$$

$$g(n) = -x(n)\sin\frac{\pi(2n+1)}{2N} + x(\frac{N}{2}+n)\cos\frac{\pi(2n+1)}{2N}. \tag{2.4}$$

Then $H(k)$ is the DWT-IV of $h(n)$ and $G(\frac{N}{2}-1-k)$ is the DWT-IV of $g(n)$. In fact, we have,

$$H(k) = \sum_{n=0}^{N/2-1} h(n)\mathrm{cas}\frac{\pi(2n+1)(2k+1)}{N}, \tag{2.5}$$

$$G(\frac{N}{2}-1-k) = \sum_{n=0}^{N/2-1} g(n)\mathrm{cas}\frac{\pi(2n+1)(2k+1)}{N}, \tag{2.6}$$

$$k = 0, 1, \cdots, N/2-1.$$

Therefore, a DWT-IV with length N is turned to 2 DWT-IV's with length $N/2$ in (2.5) and (2.6) at the cost of a pre-processing stage in (2.3)-(2.4) and a post-processing stage in the following:

$$X(2k) = H(k) - G(k),\ X(2k+1) = H(k) + G(k), \tag{2.7}$$

$$k = 0, 1, \cdots, N/2-1.$$

Based on the algorithm, we get a factorization of matrix W_N^{IV} in Lemma 1.

Lemma 1: The transforming matrix of DWT-IV can be factored recursively as follows:

$$W_1^{IV} = [1], \tag{2.8}$$

$$W_N^{IV} = P_N \begin{bmatrix} I_{N/2} & -I_{N/2} \\ I_{N/2} & I_{N/2} \end{bmatrix} \begin{bmatrix} I_{N/2} & 0 \\ 0 & \hat{I}_{N/2} \end{bmatrix} \cdot \begin{bmatrix} W_{N/2}^{IV} & 0 \\ 0 & W_{N/2}^{IV} \end{bmatrix} P'_N D_N P_N, \tag{2.9}$$

where $I_{N/2}$ is the identity matrix of order $N/2$, $\hat{I}_{N/2}$ is the matrix by reversing the rows of $I_{N/2}$, D_N is a block diagonal matrix defined by

$$D_N = \mathrm{diag}(T(0),\ T(1),\ \cdots,\ T(N/2-1)), \tag{2.10}$$

where $T(n)$ $(n = 0, 1, \cdots, N/2-1)$ are rotation matrices with order 2 defined by

$$T(n) = \begin{bmatrix} \cos\frac{\pi(2n+1)}{2N} & \sin\frac{\pi(2n+1)}{2N} \\ -\sin\frac{\pi(2n+1)}{2N} & \cos\frac{\pi(2n+1)}{2N} \end{bmatrix}, \tag{2.11}$$

P_N is permutation matrices defined by

$$P_N = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ & & \cdots & & & \cdots & & \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}, \tag{2.12}$$

P'_N is the transposition of P_N, and $W_{N/2}^{IV}$ is the transforming matrix of the scaled DWT-IV with length $N/2$.

Floating-point multiplications are needed for the algorithm when the matrix D_N is multiplied by a vector. In order to avoid floating-point multiplications, we want to turn this matrix into products of lifting matrices and then approximate the elements of the lifting matrices by numbers which are of the form $\beta/2^\lambda$, where β and λ are integers. D_N can be easily turned into products of lifting matrices. In fact, each block in D_N can be factored as:

$$T(n) = \begin{bmatrix} 1 & \tan\frac{\alpha_n}{2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\sin\alpha_n & 1 \end{bmatrix} \begin{bmatrix} 1 & \tan\frac{\alpha_n}{2} \\ 0 & 1 \end{bmatrix}, \tag{2.13}$$

where $\alpha_n = \frac{\pi(2n+1)}{2N}$. Therefore, D_N is factored into $\frac{3N}{2}$ lifting matrices. If the same method is used to factor the matrix $W_{N/2}^{IV}$ recursively until the order is 2, we get the complete factorization of W_N^{IV}.

B. Integer DWT-IV and fast algorithm

By approximating every non-zero non-diagonal element s of the lifting matrices in (2.13) by $\mathrm{RB}(s)$ defined in Definition 2, we get an approximation matrix $\bar{T}(n)$ for $T(n)$ given in the following:

$$\bar{T}(n) = \begin{bmatrix} 1 & \mathrm{RB}(\tan\frac{\alpha_n}{2}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\mathrm{RB}(\sin\alpha_n) & 1 \end{bmatrix} \times \begin{bmatrix} 1 & \mathrm{RB}(\tan\frac{\alpha_n}{2}) \\ 0 & 1 \end{bmatrix}. \tag{2.14}$$

Then D_N is approximated by $\bar{D}_N$ given by

$$\bar{D}_N = \mathrm{diag}(\bar{T}(0),\ \bar{T}(1),\ \cdots,\ \bar{T}(N/2-1)). \tag{2.15}$$

Approximating D_N by $\bar{D}_N$, we finally get an approximation matrix $\bar{C}_N^{IV}$ for C_N^{IV} which defines a new transform without floating-point multiplications. As in [5], [14], we call it type-IV Integer Discrete W Transform (IntDWT-IV). However, this does not means that the transform IntDWT-IV maps integer to integer. An integer to integer transform will be introduced in the next subsection.

Definition 3: Assume that $N = 2^t$. The transforming matrix of a type-IV Integer Discrete W Transform (IntDWT-IV) $\bar{W}_N^{IV}$ is defined recursively by :

$$\bar{W}_1^{IV} = [1], \tag{2.16}$$

$$\bar{W}_N^{IV} = P_N \begin{bmatrix} I_{N/2} & -I_{N/2} \\ I_{N/2} & -I_{N/2} \end{bmatrix} \begin{bmatrix} I_{N/2} & 0 \\ 0 & \hat{I}_{N/2} \end{bmatrix} \times \begin{bmatrix} \bar{W}_{N/2}^{IV} & 0 \\ 0 & \bar{W}_{N/2}^{IV} \end{bmatrix} P_N' \bar{D}_N P_N, \tag{2.17}$$

where the notations used are the same as those in Lemma 1.

The transform is not unique. Actually, any choice of function RB determines a transform. Based on the definition, we get a fast algorithm for the IntDWT-IV as follows.

Algorithm 1: **Fast algorithm for IntDWT-IV**

Step 1. Compute

$$\begin{aligned} &\bar{g}(n) = -f_n[x(n) + e_n x(\tfrac{N}{2}+n)] + x(\tfrac{N}{2}+n), \\ &\bar{h}(n) = x(n) + e_n x(\tfrac{N}{2}+n) + e_n \bar{g}(n), \\ &n = 0, 1, \cdots, N/2-1, \end{aligned}$$

where $e_n = \mathrm{RB}(\tan\frac{\alpha_n}{2})$ and $f_n = \mathrm{RB}(\sin\alpha_n)$.

Step 2. Compute the IntDWT-IV's with length $N/2$ of sequence $\bar{h}(n)$, $\bar{g}(n)$ and let the outputs be $\bar{H}(k)$ and $\bar{G}(k)$ respectively. If $N/2 > 1$, the $N/2$-point IntDWT-IV is decomposed into smaller ones until the length becomes 1. The 1-point IntDWT-IV can be got directly.

Step 3. Compute

$$X(2k) = \bar{H}(k) - \bar{G}(\frac{N}{2} - 1 - k),$$

$$X(2k+1) = \bar{H}(k) + \bar{G}(\frac{N}{2} - 1 - k), \; k = 0, 1, \cdots, \frac{N}{2} - 1.$$

Now we consider the computational complexity of Algorithm 1. Step 1 needs $\frac{3N}{2}$ lifting steps. Step 3 needs N additions. Let $LW^{IV}(N)$ and $AW^{IV}(N)$ represent the number of lifting steps and additions for computing an IntDWT-IV with length N. Then, we have

$$LW^{IV}(N) = \frac{3}{2}N\log_2 N, \; AW^{IV}(N) = N\log_2 N. \tag{2.18}$$

The matrix $\bar{W}_N^{IV}$ is invertible and its inverse matrix can also be factored as products of lifting matrices and simple matrices whose elements are 0, ±1 or $\pm1/2$. Therefore, we can get a fast algorithm for inverse IntDWT-IV (**reconstruction algorithm for IntDWT-IV**) directly. The number of operations for reconstruction is the same as that for the forward transform. It should be noted that the matrix $\bar{W}_N^{IV}$ is not orthogonal in general. So, we can not use $(\bar{W}_N^{IV})'$ for reconstruction.

C. Integer to integer transform

From last subsection we know that the matrix of DWT-IV is factored into products of lifting matrices and some simple matrices whose elements are either ±1 or 0. In section I, we have shown in (1.2) and (1.3) that a lifting step can be approximated by a non-linear integer to integer transform. So, if all the lifting steps in the computations of DWT-IV are approximated by integer to integer transforms, we finally get an integer to integer transform that approximates to the DWT-IV. We use the notation II-DWT-IV for the transform. Based on Lemma 1, a fast algorithm is given for II-DWT-IV in the following.

Algorithm 2: **Fast algorithm for II-DWT-IV**

Step 1. Compute

$$\begin{aligned} &h_1(n) = x(n) + \lfloor x(\tfrac{N}{2}+n)\tan(\alpha_n/2)\rfloor, \\ &g_1(n) = x(\tfrac{N}{2}+n), \\ &h_2(n) = h_1(n), \\ &g_2(n) = -\lfloor h_1(n)\sin(\alpha_n)\rfloor + g_1(n), \\ &h(n) = h_2(n) + \lfloor g_2(n)\tan(\alpha_n/2)\rfloor, \\ &g(n) = g_2(n), \\ &n = 0, 1, \cdots, N/2-1. \end{aligned}$$

Step 2. Compute the II-DWT-IV's with length $N/2$ of sequence $h(n)$, $g(n)$ and let the outputs be $H(k)$ and $G(k)$ respectively. If $N/2 > 1$, the $N/2$-point II-DWT-IV is decomposed into smaller ones until the length becomes 1. The 1-point II-DWT-IV can be got directly.

Step 3. Compute

$$X(2k) = H(k) - G(\frac{N}{2} - 1 - k),$$

$$X(2k+1) = H(k) + G(\frac{N}{2} - 1 - k).$$

II-DWT-IV is invertible and its inverse is also an integer to integer transform. By inverting the steps of Algorithm 2, we can get the reconstruction algorithm for II-DWT-IV easily.

III. Factorization of other types of DWT

Using the relationships among discrete W transforms which are stated in [6], [17], we can factor the DWT-I (DHT), DWT-II and DWT-III into lifting steps and additions. Based on the factorization, new integer DWT-I, DWT-II and DWT-III are then obtained. Table I gives the number of operations for IntDWT and DWT, where the number of operations for DWT is based on the best known algorithms for DWT in [6], [17].

Using the relationships among discrete sinusoidal transforms [2], [6], we can also get other types of integer sinusoidal transforms such as integer discrete Fourier transform (IntDFT) and integer discrete cosine transform (IntDCT). Table II gives the comparison of number of operations for IntDCT and DCT, where the number of operations for DCT is based on the best known algorithms [2], [6].

IV. Multi-dimensional integer transform and fast algorithm

In general, an MD transform can be generated by simply using the tensor products of the corresponding 1D trans-

TABLE I

Number of operations for IntDWT and DWT

transform	lifting steps	additions
IntDWT-I	$\frac{3}{2}N\log_2 N - 6N$ $+3\log_2 N + 6$	$N\log_2 N - 4N$ $+4\log_2 N + 8$
IntDWT-II	$\frac{3}{2}N\log_2 N - 3N$ $+3$	$N\log_2 N - 2N$ $+4$
IntDWT-III	$\frac{3}{2}N\log_2 N - 3N$ $+3$	$N\log_2 N - 2N$ $+4$
IntDWT-IV	$\frac{3}{2}N\log_2 N$	$N\log_2 N$
	multiplications	additions
DWT-I	$\frac{1}{2}N\log_2 N - \frac{3}{2}N$ $+2$	$\frac{3}{2}N\log_2 N - \frac{5}{2}N$ $+6$
DWT-II	$\frac{1}{2}N\log_2 N - \frac{1}{2}N$	$\frac{3}{2}N\log_2 N - \frac{3}{2}N$
DWT-III	$\frac{1}{2}N\log_2 N - \frac{1}{2}N$	$\frac{3}{2}N\log_2 N - \frac{3}{2}N$
DWT-IV	$\frac{1}{2}N\log_2 N + \frac{1}{2}N$	$\frac{3}{2}N\log_2 N - \frac{1}{2}N$

TABLE II

Number of operations for IntDCT

transform	lifting steps	additions
IntDCT-II	$\frac{3}{2}N\log_2 N - 3N + 6$	$N\log_2 N - 3N + 4$
IntDCT-III	$\frac{3}{2}N\log_2 N - 3N + 6$	$N\log_2 N - 3N + 4$
IntDCT-IV	$\frac{3}{2}N\log_2 N$	$N\log_2 N - N$
	multiplications	additions
DCT-II	$\frac{1}{2}N\log_2 N$	$\frac{3}{2}N\log_2 N - N + 1$
DCT-III	$\frac{1}{2}N\log_2 N$	$\frac{3}{2}N\log_2 N - N + 1$
DCT-IV	$\frac{1}{2}N\log_2 N + N$	$\frac{3}{2}N\log_2 N$

form, that is, we process the MD input array by implementing the one-dimensional transform along its dimensions consecutively, which is also known as the row-column method. This leads to an MD transform with separable kernel. However, sometimes non-separable transform is more useful. We can propose non-separable MD integer transforms by combining the 1D integer transforms and the polynomial transform [18], [19]. The proposed MD transform not only has a non-separable kernel, but also needs a much smaller number of operations than that of the row-column MD transform. As an example, Table III gives the comparisons between the proposed integer rD-DWT-II and the row-column integer rD-DWT-II of size $N \times N \times \cdots \times N$. The lifting steps for the proposed method is only $1/r$ times that of the row-column method. The number of additions is also decreased greatly.

V. Conclusion

Methods are proposed to factor the discrete sinusoidal transforms into lifting steps and additions. By approximating the lifting matrices, new integer discrete transforms are obtained which are floating-point multiplication free. Also integer to integer transforms are available. Fast algorithms are given for the new transforms and their computational complexities are analyzed. The proposed transforms can be used in mobile computing, lossless image coding, multiplierless filter bank and other related fields.

TABLE III

The comparison of the number of operations for rD integer DWT-II

method	lifting steps and additions
proposed	$\frac{3}{2}N^r\log_2 N - 3N^r + 3N^{r-1}$, $(r+1)N^r\log_2 N - \frac{3}{2}N^r + N^{r-1}$
row-column	$\frac{3}{2}rN^r\log_2 N - 3rN^r + 3rN^{r-1}$, $2rN^r\log_2 N - \frac{3}{2}rN^r + rN^{r-1}$

References

[1] G.P. Apousleman, M.W. Marcellin and B.R. Hunt, Compression of hyperspectral imagery using the 3-D DCT and hybrid DPCM/DCT, IEEE Trans. Geoscience and Remote Sensing, vol. 33, no. 1, pp. 26-34, 1995.

[2] V. Britanak and K.R. Rao, The fast generalized discrete Fourier transforms: a unified approach to the discrete sinusoidal transforms computation, Signal Processing, vol. 79, no. 2, pp.135-140, Dec. 1999.

[3] R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, Wavelet transforms that map integers to integers, Appl. Comput. Harmon. Anal., vol. 5, no. 3, pp. 332-369, 1998.

[4] W.K. Cham and P.C. Yip, Integer sinusoidal transforms for iamge processing, International J. Electronics, Vol. 70, no. 6, pp.1015-1030, 1991.

[5] Y.-J. Chen, Integer discrete cosine transform (IntDCT), Invited paper, The second Intern. Conf. Inform. Comm. and Sig. Proc., Singapore, Dec. 1999

[6] Z. R. Jiang, Y. Zeng and P. N. Yu, Fast algorithms, National University of Defense Technology Press, Changsha, P.R.China, 1994. (in Chinese)

[7] H.S. Malvar, Signal processing with lapped transform, Norwood, MA: Artech House, 1991.

[8] N. Memon, X. Wu and B.L. Yeo, Improved techniques for Lossless image compression with reversible integer wavelet transforms, IEEE Int. Conf. Image Processing, 1998, vol.3, pp.891-895.

[9] S.C. Pei and J.J. Ding, Integer discrete Fourier transform and its extension to integer trigomatric transforms, Proc. IEEE International Symposium on Circuits and Systems, May 28-31, 2000, Geneva, Switzerland, vol. V, PP. 513-516.

[10] W. Philps, Lossless DWT for combined lossy/lossless image coding, IEEE Int. Conf. Image Processing, 1998, vol.3, pp.871-875.

[11] K.R. Rao and P. Yip, Discrete cosine transform: algorithms, advantages and applications, Academic Press, New York, 1990.

[12] Y.L. Siu and W.C. Siu, Variable temporal-length 3-D discrete cosine transform coding, IEEE Trans. Image Processing, Vol. 6, no. 5, pp. 758-763, 1997.

[13] W. Sweldens, The lifting scheme: a construction of second generation wavelets, SIAM J. Math. Anal., vol.29, no.2, pp. 511-546, 1998.

[14] T.D. Tran, Fast multiplierless approximation of the DWT, http://thanglong.ece.jhu.edu/Tran/Pub/intDWT.ps.gz

[15] S. Vankataraman, V.R. Kanchan, K.R. Rao and M. Mohanty, Discrete transforms via the Walsh-Hadamard transform, Signal Processing, vol. 14, no. 4, pp. 371-382, June 1988.

[16] Z. Wang and B.R. Hunt, The discrete W transform, Appl. Math. Comput., vol. 16, pp.19-48, 1985.

[17] Z. Wang, The fast W transform–algorithms and programs, Science in China (series A), vol. 32, 1989, pp. 338-350.

[18] Y. Zeng, G. Bi and A.R. Leyman, New polynomial transform algorithms for multidimensional DCT, IEEE Trans. Signal Processing, vol. 48, no. 10, Oct. 2000.

[19] Y. Zeng and X. Li, Multidimensional polynomial transform algorithms for Multidimensional discrete W transform, IEEE Trans. Signal Processing, vol. 47, no. 7, pp. 2050-2053, 1999.