A Fast 4 × 4 Forward Discrete Tchebichef Transform Algorithm

Kiyoyuki Nakagaki and Ramakrishnan Mukundan

***Abstract*—The discrete Tchebichef transform (DTT) is a transform method based on discrete orthogonal Tchebichef polynomials, which have applications recently found in image analysis and compression. This letter introduces a new fast 4 × 4 forward DTT algorithm. The new algorithm requires only 32 multiplications and 66 additions, while the best-known method using two properties of the DTT requires 64 multiplications and 96 additions. The proposed method could be used as the base case for recursive computation of transform coefficients. Experimental results showing performance improvement over existing techniques are presented.**

***Index Terms*—Algorithms, discrete cosine transforms, discrete Tchebichef transform, image coding, signal processing.**

I. Introduction

IMAGE transform methods using orthogonal kernel functions are commonly used in image compression. One of the most widely used image transform methods is the discrete cosine transform (DCT) [5], used in JPEG image compression standard [11]. Due to its popularity, there have been many fast algorithms proposed for the DCT. Many of them are based on the recursion on the size of input data N [3], [10], [12], where the problem of computing the DCT coefficients of size N is recursively reduced to the problems of computing the DCT coefficients of size $N/2$. Some fast algorithms are developed specifically for two-dimensional DCT [1], [2], [7]. In addition, [6] proposes a 4 × 4 DCT algorithm to serve the base case for the recursive methods.

The discrete Tchebichef transform (DTT) is another transform method using Tchebichef polynomials [4], [9], which has as good energy compaction properties as the DCT and works better for a certain class of images [8]. Due to its high energy compaction property, the DTT has been used in image processing applications such as image compression and image feature extraction.

The DTT has the additional advantage of requiring the evaluation of only algebraic expressions, whereas certain implementations of DCT require lookup tables for computing trigonometric functions. However, the algebraic form of the DTT does not permit a recursive reduction of polynomial order as in the case of the DCT. Therefore, not many fast algorithms for the DTT have so far appeared in the literature. In this letter, a new fast 4 × 4 forward discrete Tchebichef transform is proposed. The proposed algorithm will be useful for the computation of base case for a recursive evaluation of transform coefficients in DTT-based image compression algorithms.

The definition of the DTT is given in Section II. Two of its properties are listed in Section III. Section IV gives a description of the proposed algorithm. Comparative analysis and the conclusion are given in Sections V and VI, respectively.

II. Discrete Tchebichef Transform

Given a set of input values (image intensity values for image compression) of size N × N, the forward discrete Tchebichef transform of order $p+q$ is defined as

$$T_{pq} = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} t_p(x)t_q(y)f(x,y) \qquad p,q = 0\ldots N-1 \tag{1}$$

where $t_p(x)$ is the orthonormal version of Tchebichef polynomials given by the following recursive relation:

$$t_p(x) = (\alpha_1 x + \alpha_2)t_{p-1}x + \alpha_3 t_{p-2}(x) \tag{2}$$

$$t_0(x) = \frac{1}{\sqrt{N}} \tag{3}$$

$$t_1(x) = (2x+1-N)\sqrt{\frac{3}{N(N^2-1)}} \tag{4}$$

where

$$\alpha_1 = \frac{2}{p}\sqrt{\frac{4p^2-1}{N^2-p^2}}$$

$$\alpha_2 = \frac{1-N}{p}\sqrt{\frac{4p^2-1}{N^2-p^2}}$$

$$\alpha_3 = \frac{1-p}{p}\sqrt{\frac{2p+1}{2p-3}}\sqrt{\frac{N^2-(p-1)^2}{N^2-p^2}}.$$

Both DCT and DTT satisfy the properties of separability and even symmetry as outlined in Section III.

III. Properties of the DTT

A. Separability

The definition of DTT can be written in separable form as

$$T_{pq} = \sum_{x=0}^{N-1} t_p(x) \sum_{y=0}^{N-1} t_q(y)f(x,y) \tag{5}$$

Manuscript received October 27, 2006; revised February 16, 2007. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Benoit Champagne.

The authors are with the Department of Computer Science, University of Canterbury, Christchurch, New Zealand (e-mail: kna23@student.canterbury.ac.nz; mukund@cosc.canterbury.ac.nz).

Digital Object Identifier 10.1109/LSP.2007.898331

and therefore evaluated using two one-dimensional transforms as follows:

$$g_q(x) = \sum_{y=0}^{N-1} t_q(y) f(x,y) \tag{6}$$

$$T_{pq} = \sum_{x=0}^{N-1} t_p(x) g_q(x). \tag{7}$$

B. Even Symmetry

It is shown in [9] that Tchebichef polynomials satisfy the property

$$t_p(N-1-x) = (-1)^p t_p(x), \quad \mathrm{p} = 0, 1, \ldots, \mathrm{N}-1. \tag{8}$$

This allows us to reduce the number of multiplications by redefining DTT as

$$T_{pq} = \sum_{x=0}^{\frac{N}{2}-1} t_p(x) \left(g_q(x) + (-1)^p g_q(N-1-x)\right) \tag{9}$$

where

$$g_q(x) = \sum_{y=0}^{\frac{N}{2}-1} t_q(y) \left(f(x,y) + (-1)^q f(x, N-1-y)\right). \tag{10}$$

The above two properties are commonly used in transform coding methods to get a substantial reduction in the number of arithmetic operations. It is well known that the order of complexity can be reduced from $O(N^4)$ to $O(N^3)$ using the separability property alone. In the following section, we attempt to use specifically the transform properties for $N = 4$ to further reduce the amount of computation.

IV. Fast 4 × 4 Forward DTT Algorithm

Substituting $N = 4$ and each $p \in \{0,1,2,3\}$ into (2), the orthonormal Tchebichef polynomials $t_p(x)$ is rewritten as

$$\begin{aligned}
t_0(x) &= \frac{1}{2} \\
t_1(x) &= \frac{(2x-3)}{\sqrt{20}} \\
t_2(x) &= \frac{1}{2}x^2 - \frac{3}{2}x + \frac{1}{2} \\
t_3(x) &= \frac{\sqrt{5}}{3}x^3 - \frac{3\sqrt{5}}{2}x^2 + \frac{47\sqrt{5}}{30}x - \frac{\sqrt{5}}{10}.
\end{aligned}$$

The polynomials for $x = 0 \ldots 3$ are evaluated as in Table I.

The table suggests that $t_p(x)$ for $N = 4$ has three distinct values, regardless of sign and implies the following relationships:

$$\begin{aligned}
t_0(0) &= t_0(1) = t_0(2) = t_0(3) \\
&= t_2(0) = -t_2(1) = -t_2(2) = t_2(3) = \frac{1}{2} \\
t_1(0) &= -t_1(3) = -t_3(1) = t_3(2) = -\frac{3\sqrt{5}}{10} \\
t_1(1) &= -t_1(2) = t_3(0) = -t_3(3) = -\frac{\sqrt{5}}{10}.
\end{aligned}$$

TABLE I
Evaluation of the Tchebichef Polynomials for $N = 4$ at $x = 0, 1, 2, 3$

	$x=0$	$x=1$	$x=2$	$x=3$
$t_0(x)$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
$t_1(x)$	$-\frac{3\sqrt{5}}{10}$	$-\frac{\sqrt{5}}{10}$	$\frac{\sqrt{5}}{10}$	$\frac{3\sqrt{5}}{10}$
$t_2(x)$	$\frac{1}{2}$	$-\frac{1}{2}$	$-\frac{1}{2}$	$\frac{1}{2}$
$t_3(x)$	$-\frac{\sqrt{5}}{10}$	$\frac{3\sqrt{5}}{10}$	$-\frac{3\sqrt{5}}{10}$	$\frac{\sqrt{5}}{10}$

From the above, it is obvious that $t_1(0) = 3t_1(1)$ and $t_1(0)^2 + t_1(1)^2 = 0.5$.

Since $t_p(x)$ has three distinct values, there are only the following six values for the product $t_p(x)t_q(y)$ of the definition of the DTT:

$$\begin{aligned}
A &= t_0(0)t_0(0) \quad D = t_1(0)t_1(0) \\
B &= t_0(0)t_1(0) \quad E = t_1(0)t_1(1) \\
C &= t_0(0)t_1(1) \quad F = t_1(1)t_1(1).
\end{aligned} \tag{11}$$

Factorizing the definition of the forward DTT for $\mathrm{N} = 4$

$$T_{pq} = \sum_{x=0}^{3}\sum_{y=0}^{3} t_p(x)t_q(y)f(x,y) \tag{12}$$

with respect to A, B, C, D, E, and F gives the following expressions for the transform coefficients:

$$\begin{aligned}
T_{00} = {} & A\,(f(0,0) + f(0,1) + f(0,2) + f(0,3) \\
& + f(1,0) + f(1,1) + f(1,2) + f(1,3) \\
& + f(2,0) + f(2,1) + f(2,2) + f(2,3) \\
& + f(3,0) + f(3,1) + f(3,2) + f(3,3)) \\
T_{01} = {} & B\,(f(0,0) - f(0,3) + f(3,0) - f(3,3) \\
& + f(1,0) - f(1,3) + f(2,0) - f(2,3)) \\
& + C\,(f(0,1) - f(0,2) + f(3,1) - f(3,2) \\
& + f(1,1) - f(1,2) + f(2,1) - f(2,2)) \\
T_{02} = {} & A\,(f(0,0) - f(0,1) - f(0,2) + f(0,3) \\
& + f(1,0) - f(1,1) - f(1,2) + f(1,3) \\
& + f(2,0) - f(2,1) - f(2,2) + f(2,3) \\
& + f(3,0) - f(3,1) - f(3,2) + f(3,3)) \\
T_{03} = {} & B\,(f(0,1) - f(0,2) + f(1,1) - f(1,2) \\
& + f(2,1) - f(2,2) + f(3,1) - f(3,2)) \\
& + C\,(-f(0,0) + f(0,3) - f(1,0) + f(1,3) \\
& - f(2,0) + f(2,3) - f(3,0) + f(3,3))
\end{aligned}$$

$$\begin{aligned}
T_{10} = {} & B\,(-f(0,0) - f(0,1) - f(0,2) - f(0,3) \\
& + f(2,0) + f(2,1) + f(2,2) + f(2,3)) \\
& + C\,(-f(1,0) - f(1,1) - f(1,2) - f(1,3) \\
& + f(3,0) + f(3,1) + f(3,2) + f(3,3)) \\
T_{11} = {} & D\,(f(0,0) - f(0,3) - f(3,0) + f(3,3))
\end{aligned}$$

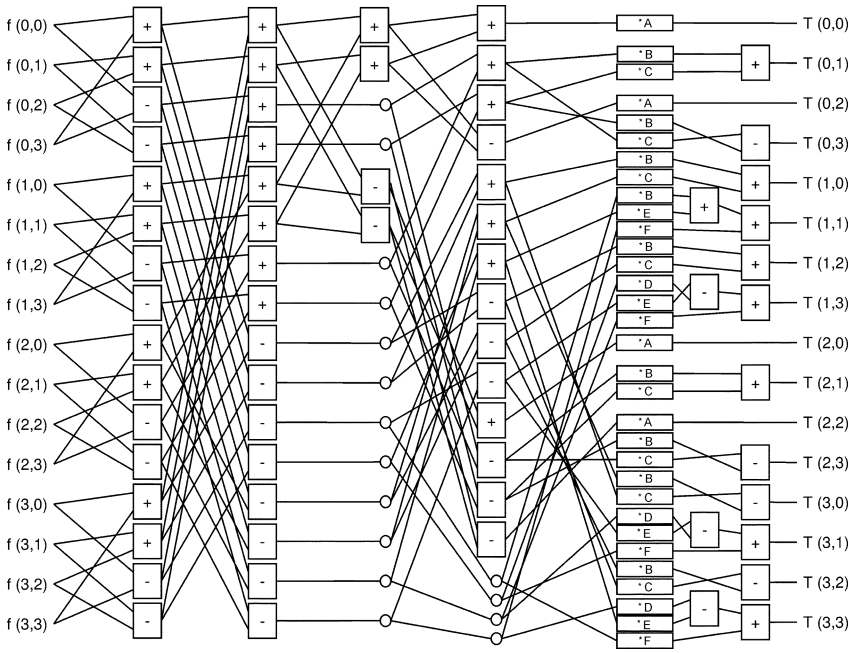

Fig. 1. Signal flow graph of the new 4 × 4 DTT algorithm.

$$
\begin{aligned}
&+E\,(f(0,1)-f(0,2)+f(1,0)-f(1,3)\\
&\quad -f(2,0)+f(2,3)-f(3,1)+f(3,2))\\
&+F\,(f(1,1)-f(1,2)-f(2,1)+f(2,2))\\
T_{12}=&B\,(-f(0,0)+f(0,1)+f(0,2)-f(0,3)\\
&\quad +f(3,0)-f(3,1)-f(3,2)+f(3,3))\\
&+E\,(-f(1,0)+f(1,1)+f(1,2)-f(1,3)\\
&\quad +f(2,0)-f(2,1)-f(2,2)+f(2,3))\\
T_{13}=&D\,(-f(0,1)+f(0,2)+f(3,1)-f(3,2))\\
&+E\,(f(0,0)-f(0,3)-f(1,1)+f(1,2)\\
&\quad +f(2,1)-f(2,2)-f(3,0)+f(3,3))\\
&+F\,(f(1,0)-f(1,3)-f(2,0)+f(2,3))
\end{aligned}
$$

$$
\begin{aligned}
T_{20}=&A\,(f(0,0)+f(0,1)+f(0,2)+f(0,3)\\
&\quad -f(1,0)-f(1,1)-f(1,2)-f(1,3)\\
&\quad -f(2,0)-f(2,1)-f(2,2)-f(2,3)\\
&\quad +f(3,0)+f(3,1)+f(3,2)+f(3,3))\\
T_{21}=&B\,(-f(0,0)+f(0,3)+f(1,0)-f(1,3)\\
&\quad +f(2,0)-f(2,3)-f(3,0)+f(3,3))\\
&+C\,(-f(0,1)+f(0,2)+f(1,1)-f(1,2)\\
&\quad +f(2,1)-f(2,2)-f(3,1)+f(3,2))\\
T_{22}=&A\,(f(0,0)-f(0,1)-f(0,2)+f(0,3)\\
&\quad -f(1,0)+f(1,1)+f(1,2)-f(1,3)\\
&\quad -f(2,0)+f(2,1)+f(2,2)-f(2,3)\\
&\quad +f(3,0)-f(3,1)-f(3,2)+f(3,3))
\end{aligned}
$$

$$
\begin{aligned}
T_{23}=&B\,(f(0,1)-f(0,2)-f(1,1)+f(1,2)\\
&\quad -f(2,1)+f(2,2)+f(3,1)-f(3,2))\\
&+C\,(-f(0,0)+f(0,3)+f(1,0)-f(1,3)\\
&\quad +f(2,0)-f(2,3)-f(3,0)+f(3,3))\\
T_{30}=&B\,(f(1,0)+f(1,1)+f(1,2)+f(1,3)\\
&\quad -f(2,0)-f(2,1)-f(2,2)-f(2,3))\\
&+C\,(-f(0,0)-f(0,1)-f(0,2)-f(0,3)\\
&\quad +f(3,0)+f(3,1)+f(3,2)+f(3,3))\\
T_{31}=&D\,(-f(1,0)+(1,3)+f(2,0)-f(2,3))\\
&+E\,(f(0,0)-f(0,3)-f(1,1)+f(1,2)\\
&\quad +f(2,1)-f(2,2)-f(3,0)+f(3,3))\\
&+F\,(f(0,1)-f(0,2)-f(3,1)+f(3,2))\\
T_{32}=&B\,(f(1,0)-f(1,1)-f(1,2)+f(1,3)\\
&\quad -f(2,0)+f(2,1)+f(2,2)-f(2,3))\\
&+C\,(-f(0,0)+f(0,1)+f(0,2)-f(0,3)\\
&\quad +f(3,0)-f(3,1)-f(3,2)+f(3,3))\\
T_{33}=&D\,(f(1,1)-f(1,2)-f(2,1)+f(2,2))\\
&+E\,(-f(0,1)+f(0,2)-f(1,0)+f(1,3)\\
&\quad +f(2,0)-f(2,3)+f(3,1)-f(3,2))\\
&+F\,(f(0,0)-f(0,3)-f(3,0)+f(3,3))
\end{aligned}
$$

The principal idea behind our algorithm is that the above expression for the transform coefficients can be viewed as linear combinations of a new set of six basis functions given in (11) instead of 16 basis functions in (1). The even symmetry property in (9) allows us to group terms of the forms $f(x,0)\pm f(x,3)$ and $f(x,1)\pm f(x,2)$ for $x=0, 1, 2, 3$ to further reduce the number of repeated additions and subtractions. Computing these terms appropriately before multiplication gives rise to the signal flow graph presented in Fig. 1. The signal flow graph directly represents our algorithm. Note that the subtraction $(-)$ gate used in the signal graph subtracts the bottom input from the top input and outputs the result.

TABLE II
NUMBER OF MULTIPLICATIONS AND ADDITIONS NEEDED TO TRANSFORM A 4 × 4 BLOCK BY THE NAIVE METHOD, THE METHOD USING SEPARABILITY AND SYMMETRY, AND OUR METHOD

	Naïve Method	Separability & Symmetry	Our Method
#Multiplications	512	64	32
#Additions	240	96	66

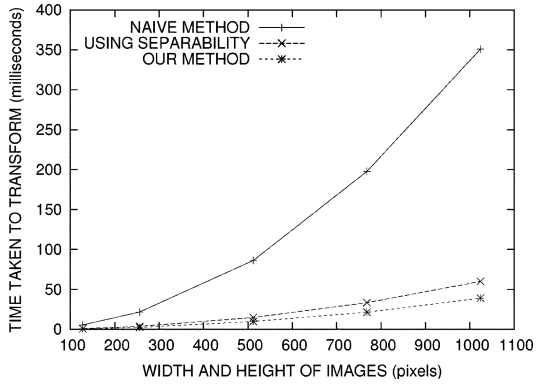

Fig. 2. Time in milliseconds taken to transform images by naive method, the method using separability and symmetry, and our method.

In our algorithm, the 4×4 input data are given as $f(i,j)$, where $i,j = 0 \ldots 3$ and go through a number of addition steps. The added terms are multiplied by one of $A = t_0(0)t_0(0)$, $B = t_0(0)t_1(0)$, $C = t_0(0)t_1(1)$, $D = t_1(0)t_1(0)$, $E = t_1(0)t_1(1)$, or $F = t_1(1)t_1(1)$. Finally, some terms are added to output $T(i,j)$, where $i,j = 0 \ldots 3$ as computed DTT coefficients. Factorization of polynomial expressions with respect to A, B, C, D, E, and F was found to yield a faster algorithm compared to the method using the separability.

V. COMPARATIVE ANALYSIS

The number of multiplications and additions needed by the proposed method, the naive method, and the method using the separability and the even symmetry for $N = 4$ are presented in Table II. The naive method refers to the direct implementation of the forward DTT definition given in (1). The method using symmetry and separability refers to the modified definition given in (10).

Table II shows that our method requires half the number of multiplications of the method using separability and symmetry. The number of additions required by the proposed method is approximately two thirds of the one required by the method using separability and symmetry. The number of multiplications could be further reduced by using bit shifts for computing products involving the constant A.

The proposed algorithm has been implemented and tested on 128×128, 256×256, 512×512, 768×768, and 1024×1024 images. The naive method using the straight definition of the DTT and the method using both separability and even symmetry have been implemented and tested for comparison against the proposed algorithm.

Fig. 2 shows that the proposed method takes about one ninth of the time taken by the naive method and takes two thirds of the time taken by the method using the separability and symmetry.

VI. CONCLUSION

This letter has presented a new algorithm for fast computation of 4×4 discrete Tchebichef transform blocks. The expressions for transform coefficients could be regrouped to form linear combinations of six new basis functions comprising of products of Tchebichef polynomials. This method of computation has yielded a framework where the amount of computation is significantly less than the conventional method using the separability property. Experimental results have shown that the proposed algorithm efficiently reduces the time taken to transform images of different sizes.

The proposed algorithm could be used as the base case for DTT computation using recursive reduction of polynomial order. Further, the representation of transform coefficients using six basis functions could also be used to reduce the number of computations in the inverse DTT used for image reconstruction. Future work in this field is to develop a fast DTT algorithm for arbitrary input size N.

REFERENCES

[1] S. Winograd and E. Feig, "Fast algorithms for the discrete cosine transform," *IEEE Trans. Signal Process.*, vol. 40, no. 9, pp. 2174–2193, Sep. 1992.
[2] K. R. Rao and F. A. Kamangar, "Fast algorithms for the 2-D discrete cosine transform," *IEEE Trans. Comput.*, vol. C-31, no. 9, pp. 899–906, Sep. 1982.
[3] H. S. Hou, "A fast recursive algorithm for computing the discrete cosine transform," *IEEE Trans. Acoust. Speech, Signal Process.*, vol. ASSP-35, no. 10, pp. 1455–1461, Oct. 1987.
[4] R. Mukundan, "Improving image reconstruction accuracy using discrete orthonormal moments," in *Proc. CISST'03*, 2003.
[5] K. R. Rao, N. Ahmed, and T. Natarajan, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, no. 1, pp. 90–93, Jan. 1974.
[6] S. U. Lee and N. I. Cho, "A fast 4 × 4 DCT algorithm for the recursive 2-D DCT," *IEEE Trans. Signal Process.*, vol. 40, no. 9, pp. 2166–2173, Sep. 1992.
[7] S. U. Lee, N. I. Cho, and I. D. Yun, "A fast algorithm for 2-D DCT," in *Proc. ICASSP'91*, 1991.
[8] R. Mukundan and O. Hunt, "A comparison of discrete orthogonal basis functions for image compression," in *Proc. Conf. Image and Vision Computing New Zealand (IVCNZ'04)*, 2004, pp. 53–58.
[9] P. A. Lee, R. Mukundan, and S. H. Ong, "Image analysis by Tchebichef moments," *IEEE Trans. Image Process.*, vol. 10, no. 9, pp. 1357–1364, Sep. 2001.
[10] S. C. Fralick, W. H. Chen, and C. H. Smith, "A fast computational algorithm for the discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-25, no. 9, pp. 1004–1009, Sep. 1977.
[11] G. K. Wallace, "The jpeg still picture compression standard," *Commun. ACM*, vol. 34, no. 4, pp. 30–44, 1991.
[12] M. V. Popovic and Z. Cvetkovic, "New fast recursive algorithms for the computation of discrete cosine and sine transforms," *IEEE Trans. Signal Process.*, vol. 40, no. 8, pp. 2083–2086, Aug. 1992.