COSC460 Honours Report

A Fast Discrete Tchebichef Transform Algorithm for Image Compression

November 2006

Kiyoyuki Nakagaki
kna23@student.canterbury.ac.nz

Supervisor : Dr. Ramakrishnan Mukundan
mukundan@canterbury.ac.nz

Department of Computer Science and Software Engineering
University of Canterbury
Christchurch, New Zealand

Abstract

The Discrete Tchebichef Transform (DTT) is a transform method based on the scaled orthogonal Tchebichef polynomials, which have applications recently found in image compression and pattern recognition. This report summarises the research carried out for the development of a fast DTT algorithm. The mathematical properties and the related algorithmic details of the Discrete Cosine Transform (DCT) have been studied in order to develop a fast DTT algorithm. A novel DTT algorithm suitable for 4×4 image blocks is proposed in this report. A theoretical analysis of the proposed method, the method using the direct application of the definition and the method using the separability and the symmetry shows that the proposed method requires the smallest number of operations while the direct application of the definition performs badly. Presented experimental results also show the improved performance of the proposed algorithm.

Keywords : The Discrete Tchebichef Transform, Image Compression, The Discrete Cosine Transform, Transform Coding, Discrete Orthogonal Polynomials

Contents

Chapter 1

Introduction

This chapter gives a general introduction to image compression and the Discrete Tchebichef Transform followed by the project goals and the outline of the report.

1.1 Introduction

Data compression has become an area of great interest in computing due to the limited data storage and network capabilities. A large amount of image data such as photographs, diagrams, videos and webpage pictures are created and transmitted all over the world as a result of the progress of digital photography and the internet. This has resulted in strong demands towards the development of good image compression systems, which specifically compress and decompress images, taking advantage of specific requirements on the compression of images. Popular image compression formats include Joint Photographic Experts Group (JPEG) format[24], Graphics Interchange Format (GIF)[1] and Portable Network Graphics (PNG) format [2]. These image compression standards are classified into two classes, lossy and lossless compressions.

Lossless data compression is a type of compression in which no information is lost over the process of compression and decompression. In other words, the original data is completely recovered when decompressing compressed data. It is crucial that no information is lost in most text compression and some other fields where the retrieval of original data is required. Entropy coding such as Huffman code [6] is a major example of lossless compression. Other lossless compressions include Lempel-Ziv [29, 30]. One implementation of Lempel-Ziv by [25] is used by GIF. Lossless compression is often combined with lossy compression to achieve further compression.

Lossy data compression is a type of compression in which some information is lost over the process of compression and decompression. Images can be compressed using lossy compression because loss of information in images is, to some extent, acceptable. Since human eyes cannot perceive small differences in two similar pictures, minor errors in images caused by the process of lossy compression are not easy for humans to recognise. JPEG is a popular example of lossy image compression. This report mainly discusses lossy image compression based on transform coding.

Typically, lossy image compression employs the techniques of digital signal processing. That is, original image data given in the spatial domain are transformed into a different domain such as the frequency domain. Such transformation methods include Fourier Transform, which offers both continuous and discrete transforms into the frequency domain based on trigonometric functions and Wavelet Transform, which also has both continuous and discrete transforms into another type of frequency domain based on compactly supported functions called ‘wavelets’. The Discrete Cosine Transform (DCT)[15] is a transform method on discrete signals and used in many signal processing applications including JPEG image compression standard.

The Discrete Tchebichef Transform (DTT) is another transform method introduced recently

[1]http://www.w3.org/Graphics/GIF/spec-gif89a.txt
[2]http://www.libpng.org/pub/png/

and based on the orthonormal version of Tchebichef polynomials. It has been shown that the DTT has a good energy compaction property, performs as well as the DCT for many images and performs better for certain classes of images[18]. It is expected that the good energy compaction property will allow to describe major features of images by relatively few number of values. In fact, Mukundan proposed a framework of image compression based on the DTT and presented a result showing that the DTT based image compression has a comparable reconstruction accuracy to the DCT in [14]. In addition, other applications of the DTT include pattern recognition [13].

One obvious measurement of the quality of image compression systems is the compression ratio the systems can achieve. Since the primary objective of image compression is to reduce the size of images as much as possible without introducing significantly visible errors. An achievable compression ratio depends on what algorithm is used. Another important requirement on image compression systems is the speed of compressing and decompressing images. The fact that image compression is used so frequently suggests that image compression needs to be done as quickly as possible so that it does not cause significant delays in the human activities for which image compression is used. Consequently, once we decide an algorithm that achieves a good compression ratio, the algorithm needs to be improved so that the compression is performed quickly.

It is evident, as discussed previously, that image compression based on the DTT will have a good image compression ratio. However, the secondary objective of image compression, which is to compress images as quickly as possible, has not been achieved for image compression based on the DTT. As the DTT was proposed recently, only few fast methods such as the one in [4] have appeared in literature. Furthermore, those methods are not useful for image compression, but for pattern recognition as discussed in later chapters. Consequently, a fast DTT algorithm needs to be developed before the further development into image compression based on the DTT.

1.2 Project Objectives

The primary objectives of the project are as follows.

- To develop a fast DTT algorithm
- To analyse and compare the algorithm with other methods

The first objective is to develop a fast DTT algorithm so that the DTT, which has a good energy compaction property, can be employed as a transform coding method for image compression. The second objective is to analyse and compare theoretically and experimentally the algorithm with other methods.

In order to achieve the first objective, image compression using transform coding was studied. Furthermore, the DCT, the DTT, their properties and their mathematical framework were studied to derive a fast method for the algorithm. In order to achieve the second objective, methods studied and derived were implemented and tested on images in addition to analysing the methods theoretically.

1.3 Report Outline

Chapter 1 has given a general introduction to image compression and the DTT, in addition to the statement of project objectives and the report outline. Given in Chapter 2 and Chapter 3 are the background to the research project. More specifically, Chapter 2 describes image compression based on transform coding and JPEG image compression standard. Chapter 3 gives the definition of the DCT, which is used in JPEG, and introduces fast algorithms for the DCT. Chapter 4 presents the definition and properties of the DTT. The discussion on various methods for the fast DTT algorithm is in Chapter 5. The theoretical analysis and experimental comparison of the known methods and the proposed algorithms are presented in Chapter 6. Finally, Chapter 7 contains the conclusion of the report and the summary of possible future work in this field.

Chapter 2

Image Compression

Image compression, based on transform coding, such as the DCT and the DTT, are discussed in this chapter. Section 2.1 describes how image compression based on transform coding are performed. In addition, definitions and lemmas used to derive a fast DTT algorithm introduced in later sections are also presented. Section 2.2 discusses block compression. Explained in Section 2.3 is the process performed by JPEG still image compression standard to compress and decompress images.

2.1 Transform Coding Based Image Compression

Image compression based on transform coding is a method of compressing image data by transforming original data in the domain of rather equal importance into the domain of more biased importance and discarding less important features to reduce the size of data. More precisely, image data are originally given in an equally important domain, namely spatial domain with colour or illumination values. Then, the data are transformed so that the image is described by another sequence of data in a different domain (often the frequency domain). Finally, some values that represent less important features of the image are discarded.

Image transform based transform coding is more formally defined as follows. First, a notion of linear combinations is defined.

Definition 1 *Let V be an N-dimensional inner product space. Let $(\bar{a}_i)_1^N$ be an orthonormal basis of V. Then, a vector $\bar{x} \in V$ is expressed as a linear combination of $(\bar{a}_i)$ as*

$$x = \sum_{i=1}^{N} \langle \bar{a}_i, \bar{x} \rangle \bar{a}_i$$

where $\langle \bar{s}, \bar{t} \rangle$ is the inner product of $\bar{s} \in V$ and $\bar{t} \in V$. The set of scalars $\langle \bar{a}_i, \bar{x} \rangle$ for $i = 1..N$ are called Transformed Coefficients throughout the report.

Suppose some elements of the basis $(\bar{a}_i)$ describe less important features of images. Then the transformed coefficients for the elements are discarded to achieve compression. It is equivalent to say that x is projected onto a subspace of V.

Definition 2 *Let V be an N-dimensional inner product space. Let $W \subset V$ be a M-dimensional subspace of V spanned by an orthonormal basis $(\bar{b}_i)_1^M$. Then the projection of $\bar{x} \in V$ onto W, called $\bar{x'}$, is defined as*

$$\bar{x'} = \sum_{i=1}^{M} \langle \bar{b_i}, \bar{x} \rangle \bar{b_i}$$

Using this definition, $\bar{x}$ described originally by N values is described as $\bar{x'}$ by M values, and $\frac{N-M}{N} * 100\%$ of compression is achieved. The following theorem [1], which is sometimes refered to as the best approximation theorem, shows that the projection $\bar{x'}$ of $\bar{x}$ onto W is the closest vector to $\bar{x}$; in other words, $\bar{x'}$ is the best approximation of $\bar{x}$ in the subspace W.

Theorem 3 *Let $(\bar{b_i})_1^M$ be an orthonormal basis of the subspace W of an inner product space V with the norm induced by the inner product of V. Then, the orthogonal projection of $\bar{x}$ denoted by $\bar{x'}$ as*

$$\bar{x'} = \sum_{i=1}^{M} \langle \bar{b_i}, \bar{x} \rangle \bar{b_i}$$

satisfies the following inequality for any $\bar{y} \in W$.

$$||\bar{x'} - \bar{x}|| < ||\bar{y} - \bar{x}||$$

Proof $\bar{x} - \bar{x'}$ is in complementary subspace of W because each $\bar{b_i}$ is perpendicular to $\bar{x} - \bar{x'}$ as follows.

$$\begin{aligned}
\langle \bar{x} - \bar{x'}, \bar{b_i} \rangle &= \langle \bar{x}, \bar{b_i} \rangle - \langle \bar{x'}, \bar{b_i} \rangle \\
&= \langle \bar{x}, \bar{b_i} \rangle - \langle \sum_{j=1}^{M} \langle \bar{x}, \bar{b_j} \rangle \bar{b_j}, \bar{b_i} \rangle \\
&= \langle \bar{x}, \bar{b_i} \rangle - \sum_{j=1}^{M} \left(\langle \bar{x}, \bar{b_j} \rangle \langle \bar{b_j}, \bar{b_i} \rangle \right) \\
&= \langle \bar{x}, \bar{b_i} \rangle - \langle \bar{x}, \bar{b_i} \rangle \\
&= 0
\end{aligned}$$

Furthermore, $\bar{y} - \bar{x'} \in W$ and $\bar{x} - \bar{x'} \in W^{\perp}$ imply that $\bar{y} - \bar{x'} \perp \bar{x} - \bar{x'}$. Therefore, $\langle \bar{y} - \bar{x'}, \bar{x'} - \bar{x} \rangle = 0$. Then,

$$||\bar{y} - \bar{x}||^2 = ||\bar{y} - \bar{x'}||^2 + 2\langle \bar{y} - \bar{x'}, \bar{x'} - \bar{x} \rangle + ||\bar{x'} - \bar{x}||^2 > ||\bar{x'} - \bar{x}||$$

Hence, $||\bar{x'} - \bar{x}|| < ||\bar{y} - \bar{x}||$ for any $\bar{y} \in W$.

Generally in image compression, an input vector $\bar{x}$ is given as a vector of real numbers and V as a real Euclidean space R^N where inner product is defined by dot product. Furthermore, the subspace W and the basis $(\bar{b_i})_1^M$ are selected dynamically depending on images. That is, the image data are transformed onto V itself using Definition 1, and then some transformed coefficients are truncated based on criteria each compression algorithm imposes. The resulting truncated transformed coefficients describe the projection of $\bar{x}$ onto the dynamically selected subspace W of V. The following definition defines the image transform.

Definition 4 (Linear Transform) *Given a signal $\bar{x}$ of length N as a vector in R^N, the transformed coefficients $(T_i)_1^N$ of $\bar{x}$ with respect to a basis $(\bar{a}_i)_1^N$ of R^N are given by*

$$T_i = \bar{a}_i \cdot \bar{x} = \sum_{j=1}^{N} a_i(j) x(j)$$

where $a_i(j)$ denotes the j^{th} component of the vector $\bar{a}_i$ and $x(j)$ denotes the j^{th} component of the vector $\bar{x}$.

Since a linear transform can be expressed as a matrix, the transform can be also written by matrix multiplication form.

Definition 5 *Let A be a matrix that represents the projection operator onto V with respect to the basis $(\bar{a}_i)_1^N$.*

$$A_{ij} = a_i(j)$$

Then, the vector of transformed coefficients $\bar{T}$ is given by

$$\bar{T} = A\bar{x}$$

The inverse transform is to retrieve the original signal from the transform coefficients. This is achieved by projecting the vector of transformed coefficients back to V with respect to the standard basis.

Lemma 6 *Given a vector of transform coefficients $\bar{T}$ with respect to the orthonormal basis $(\bar{a}_i)_1^N$, the original signal $\bar{x}$ is given by*

$$\bar{x} = A^T\bar{T}$$

where $A_{ij} = a_i(j)$ and A^T denotes the transpose of A

Proof By Definition 5, $\bar{T} = A\bar{x}$. The inverse of A exists, as the columns of A forms an orthonormal basis. Therefore, $\bar{x} = A^{-1}\bar{T}$. Since $(\bar{a}_i)_1^N$ is an orthonormal basis, A is a orthogonal matrix. It follows that $A^{-1} = A^T$. Hence, $\bar{x} = A^{-1}\bar{T} = A^T\bar{T}$.

Alternatively, the inverse transform can be given in the form of the sum of products form.

Definition 7 *Given a vector of transform coefficients $\bar{T}$ with respect to the orthonormal basis $(\bar{a}_i)_1^N$, the original signal $\bar{x}$ is given by*

$$\bar{x}(j) = \sum_{i=1}^{M} a_i(j)T_i$$

Definition 7 is equivalent to Definition 1. It is obvious by replacing the transformed coefficients T_i by the inner product, $< a_i, x >$, which correspond to T_i. In fact, the definition of the linear combination is the inverse transfrom from transformed coefficients back to the original vector.

The transform method can be extended to two dimension. Since images are two dimensional data, the two dimensional transform can be applied directly to given images.

Definition 8 *Given a two dimensional data as a function f of size $N \times N$, the matrix of transformed coefficients T with respect to the basis $(\bar{a}_i)_1^N$ is given by*

$$T_{pq} = \sum_{x=1}^{M}\sum_{y=1}^{N} a_p(x)a_q(y)f(x,y)$$

or in vector-matrix form using the matrix that represents $A_{ij} = a_i(j)$,

$$T = AXA^T$$

Any two dimensional transform given in this form induces the separability property. That is, the nested sum can be computed by two separate sums.

Lemma 9 (Separability) *Let T_{pq} be transformed coefficients of a two dimensional transform generated by the kernel $a_m(n)$.*

$$T_{pq} = \sum_{x=1}^{N}\sum_{y=1}^{M} a_p(x)a_q(y)f(x,y)$$

Then, T_{pq} can be computed in $O(N^3)$ time instead of $O(N^4)$ time as follows.

$$T_{pq} = \sum_{x=1}^{N} a_p(x) g_q(x)$$

where

$$g_q(x) = \sum_{y=1}^{N} a_q(y) f(x, y)$$

Proof Factorise the equation for two dimensional transform as follows.

$$\begin{aligned} T_p q &= \sum_{x=1}^{N} \sum_{y=1}^{M} a_p(x) a_q(y) f(x, y) \\ &= \sum_{x=1}^{N} a_p(x) \sum_{y=1}^{M} a_q(y) f(x, y) \end{aligned}$$

Then, letting $g_q(y) = g_q(x) = \sum_{y=1}^{N} a_q(y) f(x, y)$ gives the separate sums in Lemma 9. Furthermore, each sum runs through N multiplications and N-1 additions. T_{pq} are calculated for $p, q = 1..N$; Hence, the computation of T_{pq} takes $O(N^3)$ time.

The inverse two-dimensional transform is given as follows.

Definition 10 *Given transformed coefficients of size $N \times N$, the original data f is given by*

$$f(x, y) = \sum_{p=1}^{M} \sum_{q=1}^{M} a_p(x) a_q(y) T_{pq}$$

or in vector-matrix form using the matrix A such that $A_{ij} = a_i(j)$,

$$T = A^T X A$$

Summary Any image transforms based on linear transforms are defined as in Definition 4. Linear transforms are tailored to transform input into an arbitrary domain by providing the basis $(\bar{a}_i)_1^N$ (preferably orthonormal) of the domain. Generally, $(\bar{a}_i)_1^N$ is called a *kernel* of a transform.

2.2 Block Compression

Many lossy image compression systems use block compression. Block compression is to segment an input image into several blocks of certain size and to compress them independently rather than compressing the entire image. This method has several advantages; one of the significant ones is the fact that block compression can achieve a better compression ratio. Since natural images, such as photographs, have local similarities, the blocks that have highly correlated pixels like graduation or flat colouring tend to be compressed better.

2.3 JPEG still image compression standard

In 1991, Joint Photographic Experts Group proposed a lossy image compression method called JPEG [24]. Since then JPEG has become one of the most popular image compression methods. JPEG uses the DCT for transform coding and applies the Run Length Coding to achieve further

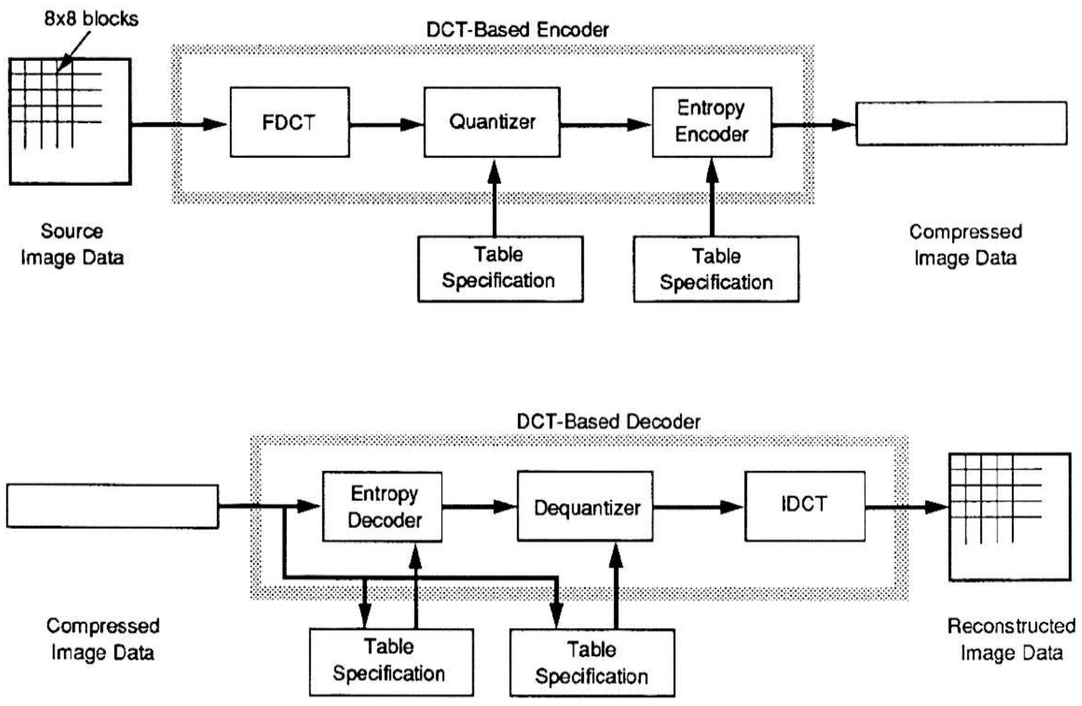

Figure 2.1: The conceptual diagram showing the process of JPEG image compression extracted from [24]

compression of the data. The DCT is discussed in detail in Chapter 3. Figure 2.3 shows the process of JPEG image compression extracted from [24].

As the diagram shows, 8×8 blocks are individually fed into the pipeline. First, the DCT is applied to the blocks to transform the data into the frequency domain. Then the 64 DCT coefficients are quantised according to a quantisation table defined by the application. The quantisation is the step to select coefficients associated with visually important features and to discard coefficients associated with visually less important features by thresholding according to the quantisation table. Then, the quantised coefficients are finally compressed further using entropy coding, in other words lossless compression. Specifically, Huffman coding [6] and Arithmetic coding [26] are applied to the selected DCT coefficients.

The decompression process consists of similar steps in the reverse order. The compressed data are first decompressed by the inverse entropy coding method of the ones used for the compression to obtain the quantised DCT coefficients. Then, the coefficients are dequantised by the same quantisation table as the one used for compression. Finally, the inverse DCT is applied onto the resulting raw DCT coefficients to obtain the signal that approximates the original signal.

Chapter 3

Discrete Cosine Transform

This chapter discusses the Discrete Cosine Transform in general. Section 3.1 gives the definition of the DCT. Section 3.2 discusses some fast DCT algorithms. Many DCT algorithms were reviewed for the development of a fast DTT algorithm as the DCT has similar properties to the DTT.

3.1 Definition

The Discrete Cosine Transform (DCT) has been used in many signal processing applications since it was introduced by Ahmed et al.[15]. The applications of the DCT include signal filtering, speech coding, image coding, pattern recognition, and many more in addition to image compression [8].

The DCT is a discrete transform whose kernel is defined by the cosine function. It is closely related to the Fourier Transform; therefore, the properties of the DCT are similar to those possessed by the Fourier Transform. Furthermore, many of the fast DCT algorithms are based on the Fast Fourier Transform algorithm [7]. There are many different definitions for the DCT; however, one generally referred to as DCT-II and often called 'the DCT' is given as follows [8].

Definition 11 (Discrete Cosine Transform (DCT-II)) *Let X be a vector of transformed coefficients and x be a vector of input sequence. Then, the DCT-II is defined as:*

$$X(m) = \sqrt{\tfrac{2}{N}} k_m \sum_{n=0}^{N-1} x(n) cos \left[\frac{(2n+1)m\pi}{2N} \right]$$

where

$$k_m = \begin{cases} \frac{1}{\sqrt{2}} & for \quad m = 0 \\ 1 & for \quad m \neq 0 \end{cases}$$

This definition of the DCT is a linear transform as in Definition 4 with the kernel defined as follows.

$$a_m(n) = \sqrt{\frac{2}{N}} k_m cos \left[\frac{(2n+1)m\pi}{2N} \right] \tag{3.1}$$

3.1.1 Properties of the DCT

The kernel of the DCT has several properties. Firstly, the kernel forms an orthonormal basis[8]. This property is useful for the inverse DCT. Lemma 6 states that the inverse transform of a linear transform with an orthonormal basis is defined by the multiplication of transformed coefficients by the transpose of the kernel matrix. In summation form, this is defined as follows.

Definition 12 (The Inverse Discrete Cosine Transform) *Given the transformed coefficients $X(m)$, the original vector $x(n)$ is retrieved by the Inverse DCT as follows.*

$$x(n) = \sqrt{\tfrac{2}{N}} k_m \sum_{m=0}^{N-1} X(m) cos\left[\frac{(2n+1)m\pi}{2N}\right]$$

Secondly, the kernel of the DCT is a even symmetric function.

Lemma 13 (Even symmetry of the DCT) *Let $c_m(n)$ be the kernel of the DCT.*

$$c_m(n) = \sqrt{\tfrac{2}{N}} k_m cos\left[\tfrac{(2n+1)m\pi}{2N}\right]$$

Then, $c_m(n)$ satisfies the even symmetry; that is, $c_m(n)$ satisfies the following equation.

$$c_m(n) = (-1)^m c_m(N-n-1)$$

Proof The proof for the lemma follows.

$$\begin{aligned}
& c_m(n) - (-1)^m c_m(N-n-1) \\
= & \sqrt{\frac{2}{N}} k_m cos\left[\frac{(2n+1)m\pi}{2N}\right] - (-1)^m \sqrt{\frac{2}{N}} k_m cos\left[\frac{(2(N-n-1)+1)m\pi}{2N}\right] \\
= & \sqrt{\frac{2}{N}} k_m \left(cos\left[\frac{(2n+1)m\pi}{2N}\right] - (-1)^m \cos\left[\frac{(2(N-n-1)+1)m\pi}{2N}\right]\right) \\
= & \sqrt{\frac{2}{N}} k_m \left(cos\left[\frac{(2n+1)m\pi}{2N}\right] - (-1)^m \cos\left[\frac{(2(N-n-1)+1)m\pi}{2N}\right]\right) \\
= & \sqrt{\frac{2}{N}} k_m \left(cos\left[\frac{(2n+1)m\pi}{2N}\right] - \cos\left[m\pi + \frac{(2N-2n-1)m\pi}{2N}\right]\right) \\
= & \sqrt{\frac{2}{N}} k_m \left(cos\left[\frac{(2n+1)m\pi}{2N}\right] - \cos\left[2m\pi + \frac{-(2n+1)m\pi}{2N}\right]\right) \\
= & \sqrt{\frac{2}{N}} k_m \left(cos\left[\frac{(2n+1)m\pi}{2N}\right] - \cos\left[\frac{-(2n+1)m\pi}{2N}\right]\right) \\
= & \sqrt{\frac{2}{N}} k_m \left(cos\left[\frac{(2n+1)m\pi}{2N}\right] - \cos\left[\frac{(2n+1)m\pi}{2N}\right]\right) \\
= & 0
\end{aligned}$$

Thirdly, the two dimensional DCT defined as follows has the separability property(Lemma 9) as the DCT is a linear transform. The DCT has many other properties including Shift in time, Scaling in time and Convolution property [8].

3.2 Fast DCT Algorithm

Due to the popularity of the DCT in many applications, there have been many fast algorithms for the DCT proposed since the discovery of the DCT. Some of the earlier ones [5, 12] computed the coefficients via the FFT, taking advantage of the close relationship between the DCT and the Fourier Transform [11]. Chen et.al. proposed one based on matrix factorisation[23]. Other algorithms such as the one by Corrington [2] and the one by Cvetkovic [28] also use matrix factorisation to achieve fast computation. Yip and Rao proposed a recursive reduction of N-point DCT to $N/2$-point DCT in both frequency [27] and time [19]. The two dimensional DCT is conventionally computed by simply applying the one dimensional DCT on both rows and columns using the separability property; however, algorithms specifically proposed for two dimensional transform have also been proposed for further optimisation[17, 3, 22, 10]. Most fast algorithms for the DCT are based on recursive reduction; therefore, the development of a fast algorithm for the base case of the recursion provides a speed improvement in any recursive algorithms. Cho and Lee proposed a 4×4 algorithm useful for the base case of fast recursive two dimensional algorithms [16].

Many DCT algorithms reviewed typically have recursive structures that allows the reduction of N point DCT down to $N/2$ point DCT. Figure 3.1 given in [8] presents the signal flow graph of the 16 point DCT using the Decimation-In-Frequency algorithm proposed in [27]. The graph illustrates that the 16 point DCT are reduced to the two 8 point DCTs. The algorithm requires $O(NlogN)$ additions and $O(NlogN)$ multiplications for the 1-D DCT[27]. Furthermore, it requires $O(N^2logN)$ additions and $O(N^2logN)$ multiplications for the 2-D DCT.

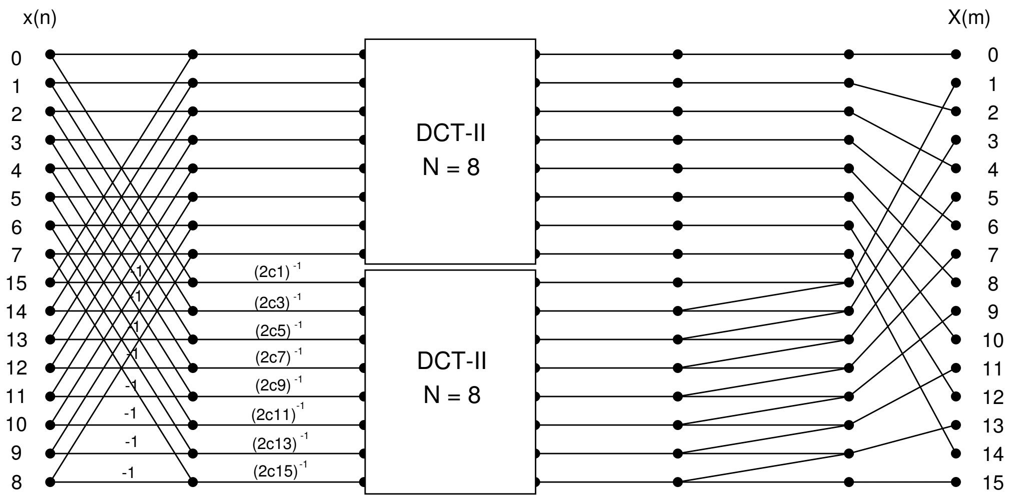

Figure 3.1: The signal flow graph of the Decimation-In-Frequency algorithm extracted from [8]

Chapter 4

Discrete Tchebichef Transform

This chapter discusses the definition of the DTT in Section 4.1 and some properties of the DTT in Section 4.2, which are used to derive a fast DTT algorithm in Chapter 5.

4.1 Definition

The Discrete Tchebichef Transform (DTT) is a linear transform with the kernel defined by the orthonormal version of Tchebichef polynomials. It has been shown in [18] that the DTT has higher energy compactness than the DCT in images that have high illumination value variations such as artificial diagrams. Furthermore, he showed that the energy compactness for natural images such as photographs are very similar with both the DCT and the DTT. The good energy compaction property of the DTT leads to the expectation of the development towards the image compression using the DTT. The family of the DTT is also used in the field of image feature extraction [13].

The DTT is defined as follows.

Definition 14 (The Discrete Tchebichef Transform) *Given an input vector, x, of N values, the vector of the transformed coefficients, X, of the Discrete Tchebichef Transform is defined as follows:*

$$X(m) = \sum_{n=0}^{N-1} x(n) t_m(n)$$

The kernel, $t_m(n)$, is given by the orthonormal version of the Tchebichef Polynomials:

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

where

$$\begin{aligned} t_0(x) &= \frac{1}{\sqrt{N}} \\ t_1(x) &= (2x + 1 - N)\sqrt{\frac{3}{N(N^2-1)}} \end{aligned}$$

and

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

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

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

Since the kernel forms an orthonormal basis, the inverse Tchebichef Transform is defined as follows:

Definition 15 (The Inverse Discrete Tchebichef Transform) *Given an N-dimensional vector, X, of the transformed coefficients of the Discrete Tchebichef Transform of the input vector x, the inverse Tchebichef Transform is defined as:*

$$x(n) = \sum_{m=0}^{N-1} X(m)t_m(n)$$

where the kernel $t_m(n)$ is given by the orthonormal version of the Tchebichef polynomials as in Definition 14.

The definition of the DTT is extended to two dimension.

Definition 16 (The two dimensional Discrete Tchebichef Transform) *Given a two dimensional input $f(x, y)$ of size $N \times N$, the transformed coefficients, T_{pq} of the two dimensional Discrete Tchebichef Transform, is defined as follows:*

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

where the kernel $t_m(n)$ is given by the orthonormal version of the Tchebichef polynomials as in Definition 14.

Furthermore, the inverse two dimensional Discrete Tchebichef Transform is defined as follows:

Definition 17 (The inverse two dimensional Discrete Tchebichef Transform) *Given transformed coefficients T of the two dimensional Discrete Tchebichef Transform of an $N \times N$ input f, the inverse two dimensional Discrete Tchebichef Transform is defined as follows:*

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

where the kernel $t_m(n)$ is given by the orthonormal version of the Tchebichef polynomials as in Definition 14.

4.2 Properties of the DTT

4.2.1 Separability

The two dimensional DTT satisfies the separability property (Lemma 9) as the DTT is a linear transform. The definition of the DTT can be written in separable form as follows:

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

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

$$
\begin{aligned}
g_q(x) &= \sum_{y=0}^{N-1} t_q(y) f(x,y) && (4.1)\\
T_{pq} &= \sum_{x=0}^{N-1} t_p(x) g_q(x) && (4.2)
\end{aligned}
$$

4.2.2 Symmetry

The orthonormal version of Tchebichef polynomials satisfies the even symmetry as follows [21].

Lemma 18 (Even Symmetry of the Tchebichef polynomials) *The Tchebichef polynomials, $t_p(q)$, defined in Definition 14 satisfies the following even symmetry equation.*

$$t_p(x) = (-1)^x t_p(N-x-1)$$

Proof The proof is by induction on p. For base case,

$$
\begin{aligned}
(-1)^0 t_0(N-x-1) &= \frac{1}{\sqrt{N}}\\
&= t_0(x)
\end{aligned}
$$

and

$$
\begin{aligned}
(-1)^1 t_1(N-x-1) &= -(2(N-x-1)+1-N)\sqrt{\frac{3}{N(N^2-1)}}\\
&= -(2N-2x-2+1-N)\sqrt{\frac{3}{N(N^2-1)}}\\
&= -(N-2x-1)\sqrt{\frac{3}{N(N^2-1)}}\\
&= (2x+1-N)\sqrt{\frac{3}{N(N^2-1)}}\\
&= t_1(x)
\end{aligned}
$$

Suppose the following.

$$
\begin{aligned}
t_{p-1}(x) &= (-1)^{p-1} t_{p-1}(N-x-1)\\
t_{p-2}(x) &= (-1)^{p-2} t_{p-2}(N-x-1)
\end{aligned}
$$

Then, the recursive formula for the Tchebichef polynomials in Definition 14,

$$
\begin{aligned}
(-1)^p t_p(N-x-1) &= (-1)^p(\alpha_1(N-x-1)+\alpha_2)t_{p-1}(N-x-1)+(-1)^p\alpha_3 t_{p-2}(N-x-1)\\
&= (-\alpha_1(N-x-1-\alpha_2)(-1)^{p-1}t_{p-1}(N-x-1)+(-1)^{p-2}\alpha_3 t_{p-2}(N-x-1)\\
&= (-\alpha_1(N-x-1)-\alpha_2)t_{p-1}(x)+\alpha_3 t_{p-2}(x)\\
&= (\alpha_1 x+\alpha_2)t_{p-1}(x)+\alpha_3 t_{p-2}(x)\\
&= t_p(x)
\end{aligned}
$$

because

$$\begin{aligned}
-\alpha_1(N-x-1)-\alpha_2 &= -\sqrt{\frac{4p^2-1}{N^2-p^2}}\left((N-x-1)\frac{2}{p}+\frac{1-N}{p}\right)\\
&= -\sqrt{\frac{4p^2-1}{N^2-p^2}}\left(\frac{2N-2x-2+1-N}{p}\right)\\
&= \sqrt{\frac{4p^2-1}{N^2-p^2}}\left(\frac{-N+2x+1}{p}\right)\\
&= x\frac{2}{p}\sqrt{\frac{4p^2-1}{N^2-p^2}}+\frac{1-N}{p}\sqrt{\frac{4p^2-1}{N^2-p^2}}\\
&= \alpha_1 x+\alpha_2
\end{aligned}$$

Hence, by induction, $t_p(x) = (-1)^x t_p(N-x-1)$ has been proved.

The definition of the DTT given in Definition 14 is rewritten using the separability and even symmetry as follows.

$$T_{pq} = \sum_{x=0}^{\frac{N}{2}-1} t_p(x)(g_q(x) + (-1)^p g_q(N-1-x)) \tag{4.3}$$

where

$$g_q(x) = \sum_{y=0}^{\frac{N}{2}-1} t_q(y)(f(x,y) + (-1)^q f(x,N-1-y)) \tag{4.4}$$

Chapter 5

Fast 2-D DTT Algorithm

This chapter begins with the fast $N \times N$ DTT algorithm, and then, a 4×4 DTT algorithm using a property specific to 4×4 transform is discussed.

5.1 NxN DTT Algorithm

One advantage of the DTT is that the transform requires the evaluation of algebraic expressions given in polynomials while the DCT requires the evaluation of trigonometric functions, which is achieved by a lookup table or a numerical approximation if higher level of precision is required. However, the Tchebichef polynomials do not have the recursive structure possessed by the cosine function as in the DCT. Therefore, not so many fast algorithms based on recursive reduction of polynomials have appeared in literature. The algorithm recently developed by Wang et al. [4] uses the Clenshaw's recurrence relations [20] to compute Tchebichef moments recursively; however, their algorithm dynamically computes Tchebichef polynomials $t_p(x)$ so that the algorithm can be used for pattern recognition, which requires the transformation of an entire image. The Tchebichef polynomials can be pre-computed in the case of image compression as block compression used in image compression fixes the size of blocks. As a result, the algorithm takes longer time than directly applying the definition with pre-computed Tchebichef polynomials although the algorithm is useful where the transform of arbitrary size is required. This section introduces two methods based on the definition and properties of the DTT.

5.1.1 Direct application of the definition

It is possible to deduce a transform method by applying directly the Definition 14. The definition is again stated below for convenience. The transformed coefficients T of the DTT is computed by:

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

where $t_p(x)$ is the orthonormal Tchebichef polynomials and N is the size of the blocks of the transform. Provided that the block size N is known prior to the implementation of the algorithm, it is possible to use a lookup table of pre-computed values for the product $t_p(x)t_q(y)$. The lookup table method allows the implementation of the DTT by a simple nested loop of one multiplication over $x = 0..N-1$ and $y = 0..N-1$. In Image compression, this method is suitable for block compression as the size of blocks are fixed. However, if the block size is unknown or arbitrarily precise computation of transformed coefficients is required, then the recursive computation of $t_p(x)$ and the dynamic creation of a lookup table for the product $t_p(x)t_q(y)$ is necessary.

5.1.2 Using separability and even symmetry

In this subsection, a DTT method using the separability and even symmetry properties, introduced in Section 4.2, is presented. As discussed in Subsection 4.2.2, it is possible to rewrite the definition of the DTT as follows.

$$T_{pq} = \sum_{x=0}^{\frac{N-1}{2}} (t_p(x)g_q(x) + (-1)^p t_p(N-1-x)g_q(N-1-x)) \tag{5.1}$$

where

$$g_q(x) = \sum_{y=0}^{\frac{N-1}{2}} (t_q(y)f(x,y) + (-1)^q t_q(N-1-y)f(x,N-1-y)) \tag{5.2}$$

Then, $t_p(x)$ is either computed recursively or extracted from a previously created lookup table as discussed in Subsection 5.1.1. However, the transformed coefficients are computed by two separate summations, whereas the method using the direct application of the definition given in Subsection 5.1.1 required doubly nested summation. Removal of nested summation will result in substantial reduction in time complexity and the number of multiplications and additions as detailed in Subsection 6.1.1.

5.2 4x4 DTT Algorithm

In this section, we propose a new 4×4 DTT algorithm using a property specific to the 4×4 DTT. Substituting $N = 4$ and each $p \in \{0, 1, 2, 3\}$ into Definition 14, the orthonormal Tchebichef polynomials $t_p(x)$ is rewritten as follows:

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

The polynomials for $x = 0...3$ are evaluated as in Table 5.1.

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

Table 5.1: Evaluation of the Tchebichef polynomials for $N = 4$ at $x = 0, 1, 2, 3$

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

$$t_0(0) = t_0(1) = t_0(2) = t_0(3) = t_2(0) = -t_2(1) = -t_2(2) = t_2(3) = \frac{1}{2}$$

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

From the above, it is obvious that

$$\begin{aligned} t_1(0) &= 3t_1(1) \\ t_1(0)^2 + t_1(1)^2 &= 0.5 \end{aligned}$$

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

$$\begin{array}{ll} A = t_0(0)t_0(0) & D = t_1(0)t_1(0) \\ B = t_0(0)t_1(0) & E = t_1(0)t_1(1) \\ C = t_0(0)t_1(1) & F = t_1(1)t_1(1) \end{array} \tag{5.3}$$

Factorising the definition of the forward DTT for N=4,

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

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

$$\begin{aligned}
T_{00} &= A\Big(f(0,0)+f(0,1)+f(0,2)+f(0,3)+f(1,0)+f(1,1)+f(1,2)+f(1,3) \\
&\quad +f(2,0)+f(2,1)+f(2,2)+f(2,3)+f(3,0)+f(3,1)+f(3,2)+f(3,3)\Big) \\
T_{01} &= B\Big(f(0,0)-f(0,3)+f(3,0)-f(3,3)+f(1,0)-f(1,3)+f(2,0)-f(2,3)\Big) \\
&\quad +C\Big(f(0,1)-f(0,2)+f(3,1)-f(3,2)+f(1,1)-f(1,2)+f(2,1)-f(2,2)\Big) \\
T_{02} &= A\Big(f(0,0)-f(0,1)-f(0,2)+f(0,3)+f(1,0)-f(1,1)-f(1,2)+f(1,3) \\
&\quad +f(2,0)-f(2,1)-f(2,2)+f(2,3)+f(3,0)-f(3,1)-f(3,2)+f(3,3)\Big) \\
T_{03} &= B\Big(f(0,1)-f(0,2)+f(1,1)-f(1,2)+f(2,1)-f(2,2)+f(3,1)-f(3,2)\Big) \\
&\quad +C\Big(-f(0,0)+f(0,3)-f(1,0)+f(1,3)-f(2,0)+f(2,3)-f(3,0)+f(3,3)\Big) \\
T_{10} &= B\Big(-f(0,0)-f(0,1)-f(0,2)-f(0,3)+f(2,0)+f(2,1)+f(2,2)+f(2,3)\Big) \\
&\quad +C\Big(-f(1,0)-f(1,1)-f(1,2)-f(1,3)+f(3,0)+f(3,1)+f(3,2)+f(3,3)\Big) \\
T_{11} &= D\Big(f(0,0)-f(0,3)-f(3,0)+f(3,3)\Big) \\
&\quad +E\Big(f(0,1)-f(0,2)+f(1,0)-f(1,3)-f(2,0)+f(2,3)-f(3,1)+f(3,2)\Big) \\
&\quad +F\Big(f(1,1)-f(1,2)-f(2,1)+f(2,2)\Big) \\
T_{12} &= B\Big(-f(0,0)+f(0,1)+f(0,2)-f(0,3)+f(3,0)-f(3,1)-f(3,2)+f(3,3)\Big) \\
&\quad +E\Big(-f(1,0)+f(1,1)+f(1,2)-f(1,3)+f(2,0)-f(2,1)-f(2,2)+f(2,3)\Big) \\
T_{13} &= D\Big(-f(0,1)+f(0,2)+f(3,1)-f(3,2)\Big) \\
&\quad +E\Big(f(0,0)-f(0,3)-f(1,1)+f(1,2)+f(2,1)-f(2,2)-f(3,0)+f(3,3)\Big) \\
&\quad +F\Big(f(1,0)-f(1,3)-f(2,0)+f(2,3)\Big) \\
T_{20} &= A\Big(f(0,0)+f(0,1)+f(0,2)+f(0,3)-f(1,0)-f(1,1)-f(1,2)-f(1,3)
\end{aligned}$$

$$\begin{aligned}
& -f(2,0)-f(2,1)-f(2,2)-f(2,3)+f(3,0)+f(3,1)+f(3,2)+f(3,3)\Big) \\
T_{21} \;=\;& B\Big(-f(0,0)+f(0,3)+f(1,0)-f(1,3)+f(2,0)-f(2,3)-f(3,0)+f(3,3)\Big) \\
& +C\Big(-f(0,1)+f(0,2)+f(1,1)-f(1,2)+f(2,1)-f(2,2)-f(3,1)+f(3,2)\Big) \\
T_{22} \;=\;& A\Big(f(0,0)-f(0,1)-f(0,2)+f(0,3)-f(1,0)+f(1,1)+f(1,2)-f(1,3) \\
& -f(2,0)+f(2,1)+f(2,2)-f(2,3)+f(3,0)-f(3,1)-f(3,2)+f(3,3)\Big) \\
T_{23} \;=\;& B\Big(f(0,1)-f(0,2)-f(1,1)+f(1,2)-f(2,1)+f(2,2)+f(3,1)-f(3,2)\Big) \\
& +C\Big(-f(0,0)+f(0,3)+f(1,0)-f(1,3)+f(2,0)-f(2,3)-f(3,0)+f(3,3)\Big) \\
T_{30} \;=\;& B\Big(f(1,0)+f(1,1)+f(1,2)+f(1,3)-f(2,0)-f(2,1)-f(2,2)-f(2,3)\Big) \\
& +C\Big(-f(0,0)-f(0,1)-f(0,2)-f(0,3)+f(3,0)+f(3,1)+f(3,2)+f(3,3)\Big) \\
T_{31} \;=\;& D\Big(-f(1,0)+(1,3)+f(2,0)-f(2,3)\Big) \\
& +E\Big(f(0,0)-f(0,3)-f(1,1)+f(1,2)+f(2,1)-f(2,2)-f(3,0)+f(3,3)\Big) \\
& +F\Big(f(0,1)-f(0,2)-f(3,1)+f(3,2)\Big) \\
T_{32} \;=\;& B\Big(f(1,0)-f(1,1)-f(1,2)+f(1,3)-f(2,0)+f(2,1)+f(2,2)-f(2,3)\Big) \\
& +C\Big(-f(0,0)+f(0,1)+f(0,2)-f(0,3)+f(3,0)-f(3,1)-f(3,2)+f(3,3)\Big) \\
T_{33} \;=\;& D\Big(f(1,1)-f(1,2)-f(2,1)+f(2,2)\Big) \\
& +E\Big(-f(0,1)+f(0,2)-f(1,0)+f(1,3)+f(2,0)-f(2,3)+f(3,1)-f(3,2)\Big) \\
& +F\Big(f(0,0)-f(0,3)-f(3,0)+f(3,3)\Big)
\end{aligned}$$

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 Equation 5.3 instead of 16 basis functions in Equation 14. The even symmetry property in Equation 18 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 Figure 5.1. 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.

In our algorithm, the 4x4 input data are given as $f(i,j)$ where $i,j = 0..3$ 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 $i,j = 0..3$ as computed DTT coefficients. Factorisation of polynomial expressions with respect to A, B, C, D, E, F was found to yield a faster algorithm compared to the method using the separability.

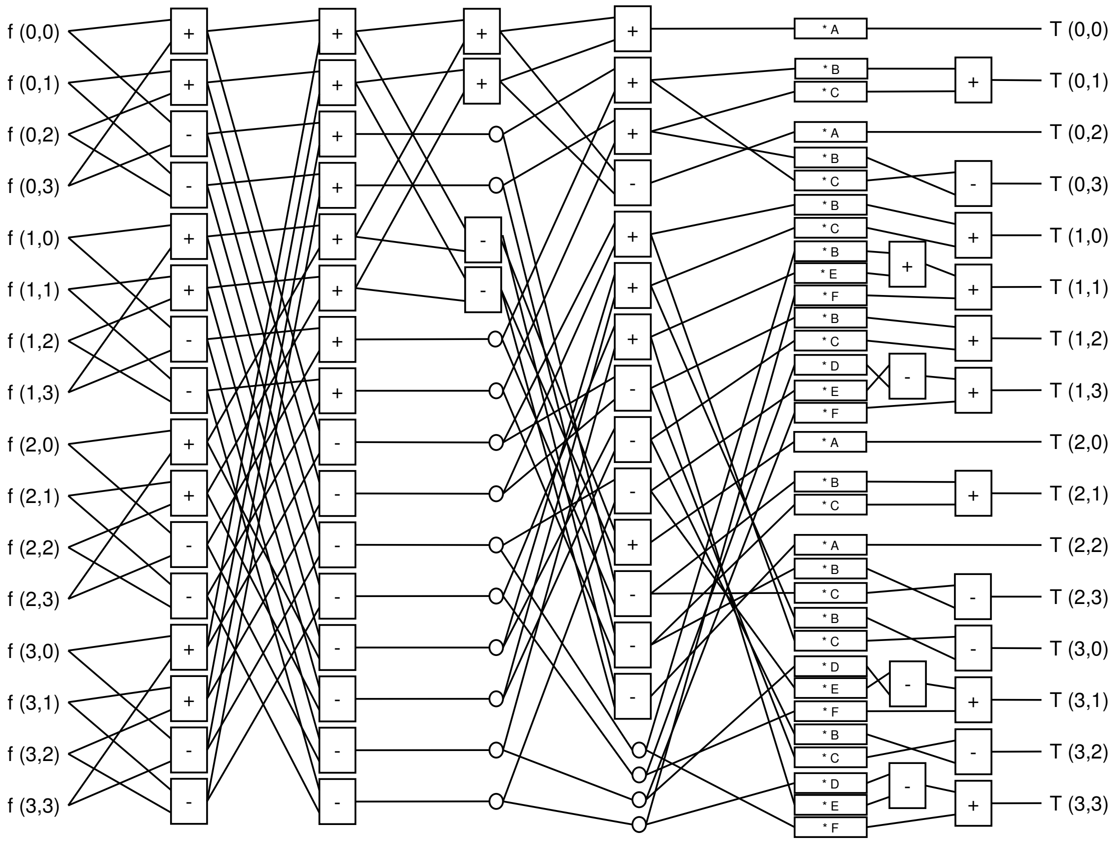

Figure 5.1: Signal Flow graph of the new 4x4 DTT algorithm

Chapter 6

Comparative Analysis

This chapter gives comparative analysis of the methods discussed in Chapter 5. Section 6.1 discusses the theoretical performance of the methods in terms of the Big-O notation and the number of multiplications and additions. Section 6.2 presents the summary of an experiment conducted to test the performance of the methods.

6.1 Theoretical Analysis

6.1.1 NxN DTT Algorithm

This subsection analyses the theoretical performance of the two methods discussed in Section 5.1. For convenience, the definition of $N \times N$ is restated below.

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

The method using direct application of the above definition discussed in Subsection 5.1.1 consists of a doubly nested loop. Each variables x and y go through from 0 up to $N-1$. In addition, the inner sum requires $(N-1)$ additions and the outer sum requires $(N-1)$ additions of the inner sums. Therefore, it requires $(N-1)N+(N-1) = (N-1)(N+1)$ additions. Furthermore, two multiplications are required for each product term, so the number of multiplications required to compute T_{pq} is $2N^2$. T_{pq} needs to be computed for $p, q = 0..N-1$. Hence, the total number of multiplications and additions required to compute all the transformed coefficients are $2N^4$ and $N^2(N-1)(N+1$ respectively.

The method using the separability and the symmetry are detailed in Subsection 5.1.2. The modified definition of the DTT using the properties is restated below for convenience.

$$T_{pq} = \sum_{x=0}^{\frac{N}{2}-1} t_p(x)(g_q(x) + (-1)^p g_q(N-1-x)) \tag{6.1}$$

where

$$g_q(x) = \sum_{y=0}^{\frac{N}{2}-1} t_q(y)(f(x,y) + (-1)^q f(x, N-1-y)) \tag{6.2}$$

In both equations, the term $(-1)^p$ simply inverts the sign of the following term depending on p, so it does not require any operations like powering or multiplication. Equation 6.1 requires one addition and one multiplication per term and the summation runs through $x = 0..\frac{N-1}{2}$; therefore, $\frac{N}{2} + \frac{N}{2} - 1 = N-1$ additions and $\frac{N}{2} - 1$ multiplications. Equation 6.2 has the exactly same structure as Equation 6.1, so the number of operations required are the same. Hence, the number of additions and multiplications required for the computation of T_{pq} are $2N-2$ and $N-2$ respectively. Furthermore, T_{pq} needs to be calculated for all of $p, q = 0..N-1$, so this method in total needs $N^2(2N-2)$ additions and $N^2(N-2)$ multiplications.

The number of operations required by the two methods can be described in terms of Landau's Big O notation [9]. The time complexity of the direct application of the definition is $O(N^4)$, while that of the method using separability and even symmetry is $O(N^3)$. Table 6.1 summarises the number of multiplications and additions required by the two methods in the exact number notation and Big O notation.

	Additions		Multiplications	
	Exact Number	Big O	Exact Number	Big O
Direct Definition	$N^2(N-1)(N+1)$	$O(N^4)$	$2N^4$	$O(N^4)$
Using Separability and Symmetry	$N^2(2N-2)$	$O(N^3)$	$N^2(N-2)$	$O(N^3)$

Table 6.1: The number of additions and multiplications required in exact number and Big O notation by two methods : the direct application of the definition and the method using the separability and even symmetry

6.1.2 4x4 DTT Algorithm

For 4x4 DTT algorithm, the block size N is fixed, so it is possible to calculate the exact number of additions and multiplications for the direct application of the definition and the method using the symmetry and even symmetry. Furthermore, the number of operations and required for the method detailed in Section 5.2 can be counted in the signal flow graph presented in Figure 5.1. Table 6.2 summarises the number of operations required by the three methods.

	Naive method	Separability & Symmetry	Our method
#Multiplication	512	64	32
#Addition	240	96	66

Table 6.2: The number of multiplications and additions needed to transform a 4x4 block by the direct application of the definition, the method using the separability and even symmetry and the method proposed in Section 5.2

6.2 Experimental Analysis

This section presents an experiment conducted to test the performance of the DTT methods discussed.

6.2.1 Experimental Method

The direct application of the DTT definition given in Section 5.1.1, the method using the separability and even symmetry discussed in Section 5.1.2 and the proposed 4x4 method presented in Section 5.2 were implemented in C programming language and compiled using *gcc*. Since the first two are given as summations of additions and multiplications, the implementation was straight conversion of summations into loops. The signal flow graph given in Figure 5.1 was implemented for the proposed 4x4 method. In addition the program offers an option for the Discrete Cosine Transform.

In addition to the implementations of the DTT algorithms, several supporting programs were created to support the experiment and the development of the algorithm. Specifically, a program that segments an input image in *pgm* format into arbitrary sized blocks was created. The program also concatenates blocks back to one entire image. A program that allows to view *pgm* files was also created to ensure that inverse transforms after applying forward transforms produce the original image. A program that computes products of two Tchebichef polynomials was made to induce some properties and tendencies of the DTT.

The experiment was conducted on an image of 5 different sizes: 128×128, 256×256, 512×512, 768×768 and 1024×1024. It is not necessary to test the algorithm on different image of the same sizes because the number of operations required by the transform is purely dependent on the size of the images. The time taken by each method were measured by the *time* facility of Linux.

Another experiment was conducted to verify the correctness of the proposed 4×4 algorithm. Two monochrome images, 'Lenna' and 'Baboon' presented in Figure 6.1, were forward transformed using the proposed method. Then, the images were reconstructed with different number of selected components along the zig-zag order used in JPEG [24] using the direct implementation of the definition of the inverse DTT transform. Then, it was tested to see the reconstructed images are identical to the original images with some minor errors introduced by the zig-zag component selection.

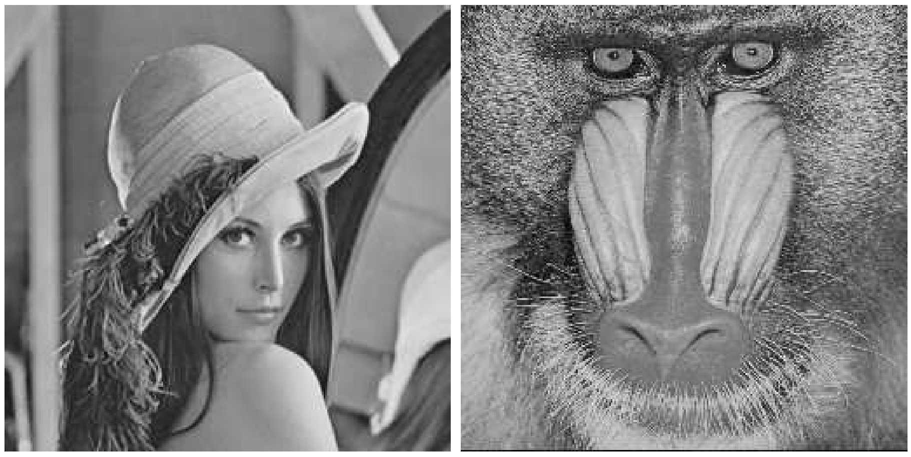

Figure 6.1: The original images, 'Lenna' and 'Baboon', of size 256×256 used to test the correctness of the proposed method

6.2.2 Results

The results from the experiment explained in Subsection 6.2.1 are presented in Figure 6.2. The x-axis of the graph is the width (or height as images are square) of an image in pixels. The y-axis of the graph is the time taken to tansform an image by the three methods in milliseconds. The naive method refers to the direct application of the definition. The results from the other experiment to verify the correctness of the proposed method are presented in Figure 6.3 and Figure 6.4. The results show that reconstructions from transformed coefficients of the proposed method with varying number of component selection along the zig-zag order. Discussion on the results is given in Section 6.3.

6.3 Discussion

The theoretical analysis of the $N \times N$ DTT transform methods given in Subsection 6.1.1 has clarified that the use of the separability and even symmetry reduced the time complexity from $O(N^4)$ of the naive direct application of the definition to $O(N^3)$ of the method using the separability and even symmetry. In fact, the separability property alone has an effect of reducing the

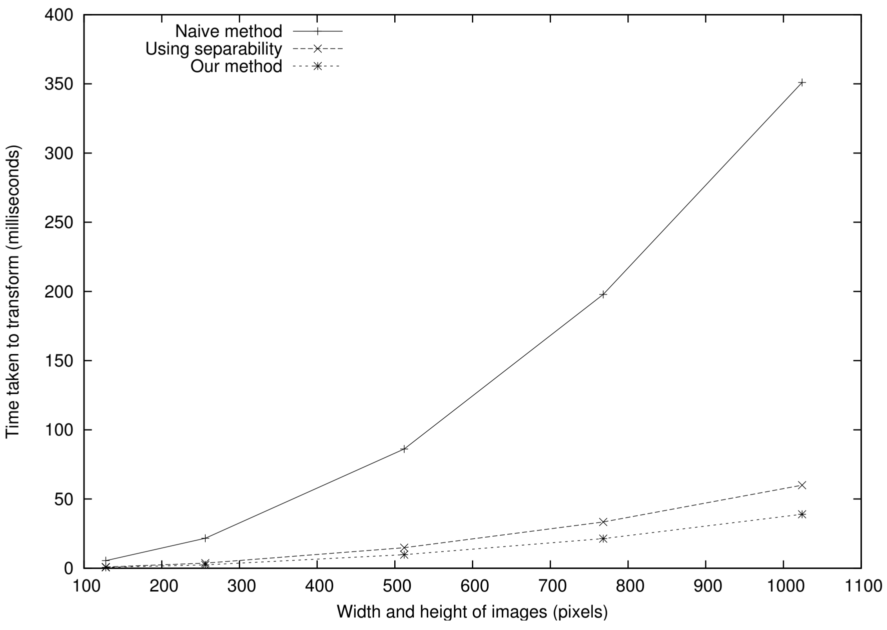

Figure 6.2: Time in milliseconds taken to transform images by the $4x4$ block DTT by the direct application of the definition (Naive method), the method using the separability and even symmetry and the method proposed in Section 5.2

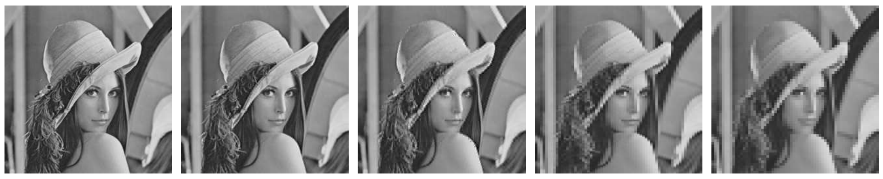

Figure 6.3: Reconstructed images of 'Lenna' with 16 components, 8 components, 4 components, 2 components and 1 component from left to right

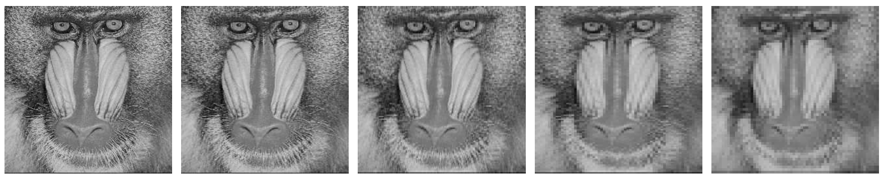

Figure 6.4: Reconstructed images of 'Baboon' with 16 components, 8 components, 4 components, 2 components and 1 component from left to right

order from $O(N^4)$ to $O(N^3)$ as proved in Lemma 9. The primary effect of the even symmetry is that it reduces the number of multiplications by half. This is visible by comparing Equation 4.3 and Equation 4.4 to Equation 4.2 and Equation 4.1.

The number of multiplications and additions required by the direct application of the definition, the method using the separability and even symmetry and the proposed 4×4 method for transforming a 4×4 block is presented in Subsection 6.1.2. Table 6.2 illustrates that the separability and even symmetry properties reduce the number additions from 240 of the direct application of the definition to 96 and the number of multiplications from 512 to 64. The reduction in the number of applications finally resulted in the great improvement of the program speed visible in Figure 6.2. For example, the method using the separability and even symmetry took approximately 50 milliseconds while the method using the direct application of the definition took approximately 350 milliseconds.

Table 6.2 has showed that the proposed 4×4 method requires the fewest number of additions and multiplications. More specifically, the number of multiplications required by the proposed 4×4 method is the half of that required by the method using the separability and the even symmetry. The number of additions required by the proposed 4×4 method is approximately two thirds of that required by the method using the separability and even symmetry. The experimental result presented in Figure 6.2 supports the reduction of the number of operations. That is, the time taken to transform 4×4 blocks of a image by the proposed 4×4 method was approximately two thirds of the time taken by the method using the separability and even symmetry and approximately 10% of the time taken by the direct application of the definition.

The reconstructed images of 'Lenna' and 'Baboon' presented in Figure 6.3 and Figure 6.4 have been tested to see if they are identical to the original images with some minor errors caused by the component selection. It is obvious that the reconstructed images are the same images as the original images. Therefore, the proposed method performed the correct forward DTT so that the inverse DTT could reproduce the images, which are almost identical to the original images presented in Figure 6.1. Although it is not a scope of this project, the results also show the effect of component selection. The reconstructed images with 16 components without discarding any components are visually identical to the original image without any visual distortions, while the reconstructed images with only 1 component after discarding 15 components are greatly distorted. In fact, the first component simply represents the average of the 4×4 blocks, the selection of only one component has a effect of averaging 4×4 blocks to the lower resolution. The distortion is more visible as the number of selection decreased.

Chapter 7

Conclusion

In this project, the fast Discrete Tchebichef Transform algorithm was studied. More specifically, the two main objectives of the research were to develop a new fast method for the DTT for image compression and to compare various methods of the DTT experimentally and theoretically. To accomplish the fast objective, image compression based on transform coding, its theories and its properties were studied in addition to the Discrete Cosine Transform, which exhibit similar properties to the DTT. A new method that works for 4×4 blocks using a property specific to 4×4 DTT was proposed. The method was analysed and compared against the method using the direct application of the definition of the DTT and the method using the separability and even symmetry properties to accomplish the second goal.

The separability property, which both the 2-D DTT and 2-D DCT exhibits, is used to reduce the order of computation from $O(N^4)$ to $O(N^3)$. The even symmetry, which also both the 2-D DTT and 2-D DCT exhibits, is useful as it allows the reduction of the number of multiplications by half. As a result the method using the separability and even symmetry requires fewer additions and multiplications than those required by the method using the direct application of the definition of the DTT. The kernel of the DTT does not have an explicit recursive structure the cosine function, which is the kernel of the DCT, has, so a method for the recursive reduction of computational complexity like the one DCT allows is by far not possible. Therefore, the method using the separability and even symmetry takes the shortest time for transformation of blocks of arbitrary size.

A new 4×4 DTT method was proposed in Section 5.2. The method uses the property that the Tchebichef polynomials have three distinct values regardless of signs for 4×4 case. The definition of the DTT was factorised so that the transformed coefficients are expressed in terms of six values, the three Tchebichef polynomial values generate. This process effectively reduces the size of basis from 16 to 6. The signal graph given in Figure 5.1 represents the algorithm and can be implemented in both hardware and software. It was clarified that the method takes the smallest number of additions and multiplications among three. The experiment conducted on the three methods have shown that the proposed algorithm takes the shortest time among three while the method using the separability and even symmetry took much shorter time than the direct application of the definition. The correctness of the proposed method was verified by another experiment. Images transformed by the proposed method were reconstructed using the inverse DTT and appeared to be identical to the original images with some minor errors caused by the lossy nature of the method.

7.1 Future Work

Finally, this section discusses possible future works to conclude the report.

7.1.1 Towards DTT Based Image Compression

As discussed in 2.1, lossy image compression based on transforming generally consists of three steps. It is expected to develop a DTT based image compression using the three steps. The first step is to transform the input image blockwise by the DTT. The second step is to select components by quantising the transformed coefficients. This step is dependent on the transform method used; that is, the transformed efficients should be quantised so that the components that represent visually important features of images should be selected and ones that represent visually less important features should be truncated. As the transformed coefficients generated from different transforms represent different features of images, quantisation methods suitable for the DTT need to be studied. Furthermore, the quantised transformed coefficients have different order of importance and different entropy, so different way of selecting components and entropy coding also needs to be studied. The proposed 4×4 method can be used in the development of a codec.

7.1.2 Recursive Reduction of the Polynomial Order

To achieve a major reduction in the time complexity of the DTT, a method that allows the recursive reduction of the transform from order of N down to order of $N/2$ or smaller needs to be developed. Such method would take $O(NlogN)$ time for 1-D transfrom and $O(N^2logN)$ time for 2-D transform. Furthermore, if exists, a recursive structure of the transformed coefficients similar to wavelets transform would result in $O(N)$ time for 1-D transfrom and $O(N^2)$ time for 2-D transform.

7.1.3 A Fast Inverse Discrete Tchebichef Transform Algorithm

Compress images need to be decompressed. The process of decompression involves the inverse DTT. The inverse DTT was only briefly touched in this research project. Therefore, methods for the inverse DTT needs to be studied in greater detail to derive a suitable algorithm that requires minimal amount of computation. The proposed method could be suitably modified for the inverse transform. Recent research in this field has also looked at the possibility of using techniques such as Clenshaw's recurrence formula [20] for the inverse DTT transform.

Publications

A paper on the 4x4 DTT algorithm discussed in Section 5.2 was submitted to IEEE Signal Processing Letters[1], and is currently under review. The details of the paper are as follows:

Title	A Fast 4x4 Forward Discrete Tchebichef Transform Algorithm
Author	Kiyoyuki Nakagaki and Ramakrishnan Mukundan

[1]http://www.ieee.org/organizations/society/sp/spl.html

Bibliography

[1] E. Cheney. *Introduction to Approximation Theory.* MacGraw Hill, 1966.

[2] M. S. Corrington. Implementation of fast cosine transforms using real arithmetic. In *Proceedings of NAECON, Dayton Ohio, '78;*, pages 350–357, 1978.

[3] K. R. Rao F. A. Kamangar. Fast algorithms for the 2-d discrete cosine transform. *IEEE Trans. on Computers*, Vol. 31:pp. 899–906, 1982.

[4] Wang G. Wang, S. Recursive computation of tchebichef moment and its inverse transform. *Pattern Recognition*, 39(1):47–56, 2006.

[5] R. M. Haralick. A storage efficient way to implement the discrete cosine transform. *IEEE Transactions on Computers*, C-25(7):764–765, 1976.

[6] D. A. Huffman. A method for the construction of minimum redundancy codes. *In Proceedings of the IRE*, 40:1098–1101, 1952.

[7] J.W. Turkey J. W. Cooley. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[8] P. Yip K. R. Rao. *Discrete cosine transform: algorithms, advantages, applications.* Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[9] Donald E. Knuth. *The art of computer programming, volume 1 (3rd ed.): fundamental algorithms.* Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1997.

[10] PeiZong Lee, Fang-Yu Huang, Chorng-Yuan Huang, and Hwann-Tzong Chen. Efficient implementations of two variant subset sum problems: a case study of how to process appraisal books resulting from fire-destroyed money. In *Selected Areas in Cryptography*, pages 230–237, 1996.

[11] K. R. Rao M. Chelemal-D. Fast computational algorithms for the discrete cosine transform. In *Nineteeth Asilomar Conference on Circuits, Systems and Computers 1985*, 1985.

[12] A. M. Peterson M. J. Narasimha. On the computation of the discrete cosine transform. *IEEE Trans. on communications*, 26:934–936, 1978.

[13] R. Mukundan. Radial tchebichef invariants for pattern recognition. In *Proc. of IEEE-Tencon05 Conference Melboroune*, pages 2098–2103, 2005.

[14] R. Mukundan. Transform coding using discrete tchebichef using discrete tchebichef polynomials. In *Visualization, Imaging, and Image Processing 2006*, 2006.

[15] K. R. Rao N. Ahmed, T. Natarajan. Discrete cosine transform. *IEEE Transactions on Computers*, C-23:90–93, 1974.

[16] S. U. Lee N. I. Cho. A fast 4x4 dct algorithm for the recursive 2-d dct. *IEEE Trans. on Signal Processing*, Vol. 40, No. 9:pp. 2166–2173, 1992.

[17] S. U. Lee N. I. Cho, I. D. Yun. A fast algorithm for 2-d dct. In *ICASSP'91*, 1991.

[18] R. Mukundan O. Hunt. A comparison of discrete orthogonal basis functions for image compression. In *Proc. of Conf. on Image and Vision Computing IVCNZ'04 New Zealand*, pages 53–58, 2004.

[19] K. R. Rao P. Yip. Fast decimation-in-time algorithms for a family of discrete sine and cosine transforms. *Circuits, Systems, and Signal Processing*, 3(4):387–408, 1984.

[20] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, New York, NY, USA, 1992.

[21] P.A. Lee R. Mukundan, S.H. Ong. Image analysis by tchebichef moments. *IEEE Transactions on Image Processing*, Vol 10:1357–1364, 2001.

[22] M. Vetterli. Fast 2-d discrete cosine transform. In *IEEE International Conference on Acoustics, Speech, and Signal Processing '85*, 1985.

[23] S.C. Fralick W. H. Chen, C. H. Smith. A fast computational algorithm for the discrete cosine transform. *IEEE Trans. Commun.*, vol. COM-25:pp. 1004–1009, 1977.

[24] Gregory K. Wallace. The jpeg still picture compression standard. *Commun. ACM*, 34(4):30–44, 1991.

[25] T. Welch. A technique for high performance data compression. *IEEE computer*, 17(6):8–19, 1984.

[26] Ian H. Witten, Radford M. Neal, and John G. Cleary. Arithmetic coding for data compression. *Commun. ACM*, 30(6):520–540, 1987.

[27] P. Yip and K. R. Rao. The decimation-in-frequency algorithms for a family of discrete sine and cosine transform. *Circuits, Systems, and Signal Processing*, 7(1):4–19, 1988.

[28] M. V. Popovic Z. Cvetkovic. New fast recursive algorithms for the computation of discrete cosine and sine transforms. *IEEE Trans. Signal Process.*, SP-40:pp. 2083–2086, 1992.

[29] Jacob Ziv and Abraham Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343, 1977.

[30] Jacob Ziv and Abraham Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, 1978.