Universidade Federal de Pernambuco
Centro de Tecnologia e Geociências
Programa de Pós-graduação em Engenharia Elétrica

Raíza dos Santos Oliveira

Design and Implementation of Low-complexity Discrete Transforms for Image and Video Coding

Recife, Julho de 2018.

Raíza dos Santos Oliveira

Design and Implementation of Low-complexity Discrete Transforms for Image and Video Coding

Dissertação submetida ao Programa de Pós-graduação em Engenharia Elétrica da Universidade Federal de Pernambuco como parte dos requisitos para obtenção do grau de Mestre em Engenharia Elétrica

Orientador: Prof. Dr. Renato J. Cintra

Co-orientador: Prof. Dr. Fábio M. Bayer

Recife, Julho de 2018.

Resumo da Dissertação apresentada à UFPE como parte dos requisitos necessários para a obtenção do grau de Mestre em Engenharia Elétrica

Design and Implementation of Low-complexity Discrete Transforms for Image and Video Coding

Raíza dos Santos Oliveira

Julho/2018

Orientador: Prof. Dr. Renato J. Cintra
Co-orientador: Prof. Dr. Fábio M. Bayer
Área de Concentração: Comunicações
Palavras-chaves: Keywords
Número de páginas: 72

No campo de processamento de sinais, as transformadas discretas trigonométricas desempenham um papel muito importante. A aplicação das transformadas nos permite olhar para os dados em análise sob outras perspectivas, domínio das transformadas, trazendo novas informações e interpretações sobre estes dados. Em particular, três transformadas se destacam: a transfromada discreta de Fourier (DFT), a transfromada discreta de Hartley (DHT) e a transformada discre do coseno (DCT). Apesar de serem amplamente utilizadas, a computação dessas transformadas requer um alto custo computacional. Algoritmos rápidos para o cálculo da DFT, DHT e DCT diminuem consideravelmente o custo de sua computação. Entretanto, a necessidade de implementações usando aritmética em ponto flutuante nÃ£o é eliminada. Neste sentido, as aproximações matriciais se tornam uma alternativa de baixa complexidade para a computação da DFT, DHT e DCT. Neste trabalho, é introduzido um método baseado em uma heurística gulosa para obtenção de aproximações para uma determinada matrix. A melhor forma de aplicação do método a DFT, DHT e DCT é discutida. O método é utilizado para obtenção de novas aproximações matriciais para a DCT de comprimento 8. As approximações obtidas são avaliadas por meio de figuras de mérito clássicas na área e no contexto de compressão de imagens por um esquema do tipo JPEG. Diversas aproximações propostas apresentam resultados melhores que os da DCT em termos do SSIM nos experimentos de compressÃ£o de imagens. Um algoritmo rápido é proposto para uma das aproximações introduzidas e sua implementação em FPGA realizada.

Abstract of Dissertation presented to UFPE as a partial fulfillment of the requirements for the degree of Master in Electrical Engineering

Design and Implementation of Low-complexity Discrete Transforms for Image and Video Coding

Raíza dos Santos Oliveira

July/2018

Supervisor: Prof. Dr. Renato J. Cintra
Co-supervisor: Prof. Dr. Fábio M. Bayer
Area of Concentration: Communications
Keywords: Keywords
Number of pages: 72

In the field of signal processing, the discrete trigonometric transforms play an important role. The use of these transforms allows to look at the data under study from different perspectives, the transforms domain, bring out new informations and interpretations of the data. In particular, there are tree transforms that stand out: the discrete Fourier transform (DFT), the discrete Hartley transform (DHT), the discrete cosine transform (DCT). Despite being widely used, the computational cost of implementing these transforms is high. The use of fast algorithms to implement the DFT, DHT, and DCT, substantially reduce its cost. However, the it is still necessary to consider float-point arithmetic. In this sense, low-complexity matrix approximations appear as an alternative way to compute these transforms. In this work, a greedy algorithm to obtain low-complexity approximations for a given matrix. We discuss the best ways to apply the proposed method to approximate the DFT, DHT and DCT. The method is used to find approximations for the 8-point DCT. The proposed approximations are evaluated using classic figures of merit and in the context of image compression. Several of the proposed approximations overcome the DCT in terms of SSIM in the image compression experiments. For one of the introduced approximations we propose a fast algorithm and present its implementation in FPGA.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Motivation and context

A signal might be seen as a function that changes with time and/or space and transmits information about the behavior of the phenomenon under study [1], [2]. The *IEEE Transactions on Signal Processing* internet page states that the "term 'signal' includes, among others, audio, video, speech, image, communication, geophysical, sonar, radar, medical and musical signals" [3].

The field of signal processing comprises, among other things, a collection of techniques to obtain, manipulate, analyze, represent, transmit and extract information from an input signal [2]. In particular, transforms play an important role in this area of research. The use of transforms allow us to look at data from a different perspective, the transform domain, which often adds new interpretations to the data under analysis. For example, the Fourier transform decomposes an input signal into its frequency components and the Karhunen–Loève transform is capable of decorrelating data sequences [4].

Among all the possible transforms, the ones with sinusoidal kernels are particularly important [5]. Special interest is given to the discrete transforms, because they are suitable for real-world applications using digital computers which are inherently capable of discrete, finite calculations only [6].

The discrete Fourier transform (DFT) is one of the most important discrete transforms [7]. It finds application in many different problems such as solving difference equations [8], image processing [9], [10], beamforming [11], [12], analysis of radar signals [13], [14], voice process-

ing [15], time series [16]–[18], spectral estimation [19], harmonic regression [20], and analysis of biomedical signals [21].

The discrete Hartley transform (DHT), introduced by Bracewell in 1983 [22], is also an important transform. The DHT is an attractive discrete transformation mainly due to the following properties: (i) the DHT is isomorphic to the DFT [22]; (ii) the multiplicative complexities of the DHT and the DFT are identical in the sense discussed in Heideman [23]; (iii) unlike the DFT, the DHT is a real transform, which means it does not requires complex arithmetic for its computation [22]; (vi) the forward and inverse transforms are basically the same; and (v) the DHT is very symmetric, which facilitates its computation and implementation [24]. Because of its similarities with the DFT, the DHT is also applied in many different fields of study. Some examples are: image processing [25], [26]; convolution computation [27], [28]; audio processing [29]; biomedical image analysis [30]; and solution of power system problems [31].

The DCT is applied, for example, in areas such as image processing [32]–[34], audio processing [35], watermarking [36], [37], and gait recognition [38]. However, its most popular use is in data compression [4]. In particular, the DCT is applied in several image and video compression patterns, such as the JPEG [39], MPEG [40], H.261 [41], H.263 [42], H.264/AVC [43], and HEVC [44]. The good performance of the DCT for data compression can be justified by the fact the the DCT is asymptotically equivalent to the Karhunen–Loève transform (KLT), which is the optimal transform for data compression, when the input signal has some specific features [4].

Although these transforms are very popular, the computational cost of implementing them requires float-point arithmetic. Fast algorithms can dramatically reduce their computational cost. However, the number of calls in applications of these transforms can be extraordinarily high. For instance, a single image frame of high-definition TV (HDTV), that can be encoded with the DCT, contains 32.400 8×8 image subblocks. Therefore, computational savings in the transformation step may effect significant performance gains, both in terms of speed and power consumption [45], [46].

Being quite a mature area of research [47], there is little room for improvement on the exact computation of the DFT, DHT, and DCT. Thus, one approach to further minimize the computational cost of computing the discrete transforms is the use of matrix approximations [48], [49]. Such approximations provide matrices with similar mathematical behavior to the exact

transform while presenting a dramatically low arithmetic cost.

1.2 Goals

In this work we aim at:

- ▷ Introduce a greedy search algorithm for matrix approximation based on angular distance between vectors;
- ▷ Discuss how the proposed method can be applied to the DFT, DHT and DCT;
- ▷ Use the proposed algorithm to introduce new approximations for the DCT of length $N = 8$;
- ▷ Test the efficiency of the proposed approximations on image compression experiments when compared to the exact DCT and other approximations in literature.

1.3 Structure

The present work is structured as follows:

In Chapter 2, we present the discrete trigonometric transforms that we are going discuss along the work, the DFT, DHT, and DCT. An overview of their mathematical structure is made.

In Chapter 3, we present some popular fast algorithms and low-complexity approximations for the DCT found in literature. The low-complexity approximations shown in this chapter are going to be used further for comparison with the approximations introduced here.

The search algorithm for matrix approximation is introduced in Chapter 4. The proposed method is based on an optimization problem with no constraints. A constrained to orthogonality version of the proposed method is also introduced. Then, since we are also considering the DFT, which has its coefficients defined over the complex field, we also discuss how the method can be used to approximate matrices with complex elements.

Some features of the matrix given as an input to the algorithm might be explored in order to reduce the complexity of the procedure. Thus, in Chapter 5, we explore some features in the structure of the DFT, DHT and DCT, and define approximations schemes based on the combination of features considered and version of the proposed method used.

In Chapter 6, the approximation schemes defined in the previous chapter are used to find new approximations for the DCT of length $N = 8$. The proposed approximations are evaluated

according to popular figures of merit and compared to exact DCT and other approximations in literature.

In Chapter 7, a JPEG-like experiment for image compression is defined and used to evaluate the performance of the proposed approximations in comparison to the DCT and the other approximations in literature.

In Chapter 8, an overview of the topics discussed and all the results obtained is presented.

Chapter 2

Discrete trigonometric transforms

Two N-dimensional vectors, say $\mathbf{x} = \begin{bmatrix} x_0 & x_1 & \dots & x_{N-1} \end{bmatrix}^\top$ and $\mathbf{X} = \begin{bmatrix} X_0 & X_1 & \dots & X_{N-1} \end{bmatrix}^\top$, relate to each other through a discrete sinusoidal transform according to the following expressions:

$$X_k = \sum_{i=0}^{N-1} x_i \cdot \ker(i,k,N), \quad k = 0, 1, \dots, N-1, \tag{2.1}$$

$$x_i = \sum_{k=0}^{N-1} X_k \cdot \ker^{-1}(i,k,N), \quad i = 0, 1, \dots, N-1, \tag{2.2}$$

where $\ker(\cdot,\cdot,\cdot)$ and $\ker^{-1}(\cdot,\cdot,\cdot)$ are the forward and inverse transformation kernels. In this work, although our main goal is to propose new approximations for the discrete cosine transform (DCT), we also discuss the discrete Fourier transform (DFT) and the discrete Hartley transform (DHT), which are related transforms.

2.1 Discrete Fourier transform

The N-point DFT has its coefficients defined as in (2.1) with its kernel given by

$$\ker(i,k,N) = \cos\left(\frac{2\pi ik}{N}\right) - j\sin\left(\frac{2\pi ik}{N}\right) = e^{-j2\pi ik/N}, \quad i,k = 0, 1, \dots, N-1.$$

Its inverse kernel is furnished by

$$\ker^{-1}(i,k,N) = \frac{1}{N}\left[\cos\left(\frac{2\pi ik}{N}\right) + j\sin\left(\frac{2\pi ik}{N}\right)\right] = \frac{1}{N}e^{j2\pi ik/N}, \quad i,k = 0, 1, \dots, N-1,$$

where $j = \sqrt{-1}$.

Matrix representation of the DFT

The DFT of an input signal of length N can be calculated by a matrix operation as follows:

$$\mathbf{X} = \mathbf{F}_N \cdot \mathbf{x}, \tag{2.3}$$

and $\mathbf{F}_N$ is the DFT matrix given by:

$$\mathbf{F}_N = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_N & \omega_N^2 & \dots & \omega_N^{(N-1)} \\ 1 & \omega_N^2 & \omega_N^4 & \dots & \omega_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_N^{(N-1)} & \omega_N^{2(N-1)} & \dots & \omega_N^{(N-1)(N-1)} \end{bmatrix},$$

where $\omega_N = e^{-j2\pi/N}$. The matrix $\mathbf{F}_N$ is orthogonal. Then, we have that $\mathbf{F}_N^{-1} = \frac{1}{N}\mathbf{F}_N^*$, where $\mathbf{F}_N^*$ is the Hermitian matrix of $\mathbf{F}_N$ [50].

Figure 2.1 displays, for some values of N, the image representation of the real and complex parts of $\mathbf{F}_N$. The darker colors represent smaller values while lighter color represent larger values. This kind of representation is useful to see patterns and symmetries in the matrix, which can be explored to simplify computations and identify redundacies in the computation of (2.3).

Computational complexity

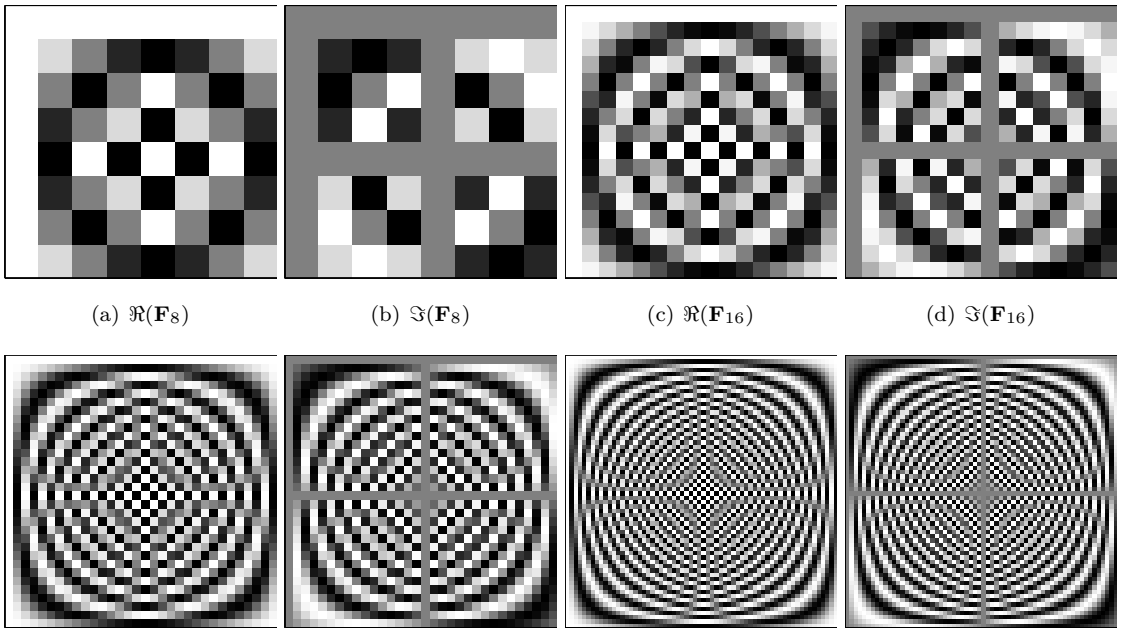
To calculate the DFT coefficients (Equation (2.3)), it is necessary to perform N^2 complex multiplications and $N(N-1)$ complex additions [6]. Each complex multiplication requires four real multiplications and two real additions. Each complex addition requires two real additions. Therefore, the computation of all the DFT coefficients requires at most $4N^2$ real multiplications and $N(4N-2)$ real additions.

2.2 Discrete Hartley transform

The DHT forward and inverse kernels are given by:

$$\ker(i,k,N) = \frac{1}{N}\operatorname{cas}\left(\frac{2\pi ik}{N}\right),\ k = 0,1,\dots,N-1, \tag{2.4}$$

$$\ker^{-1}(i,k,N) = \operatorname{cas}\left(\frac{2\pi ik}{N}\right),\ i = 0,1,\dots,N-1.$$



(a) $\Re(\mathbf{F}_8)$ (b) $\Im(\mathbf{F}_8)$ (c) $\Re(\mathbf{F}_{16})$ (d) $\Im(\mathbf{F}_{16})$

(e) $\Re(\mathbf{F}_{32})$ (f) $\Im(\mathbf{F}_{32})$ (g) $\Re(\mathbf{F}_{64})$ (h) $\Im(\mathbf{F}_{64})$

Figure 2.1: *Image representation of the real and imaginary parts of the DFT matrix considering* $N = 8, 16, 32, 64$. *The functions* $\Re(\cdot)$ *and* $\Im(\cdot)$ *return the real and complex parts of its arguments, respectively.*

where $\text{cas}(x) = \cos(x) + \sin(x)$.

The DFT and DHT relate to each other through a very simple expression. Let $X_k^{Fourier}$ and $X_k^{Hartley}$ be kth coefficient of the DFT and DHT spectrum, respectively, computed from $\mathbf{x}$ according to (2.1). Then, the DHT coefficients are calculated in terms of the DFT coefficients as follows

$$X_k^{Hartley} = \Re(X_k^{Fourier}) - \Im(X_k^{Fourier}). \tag{2.5}$$

The matrix representation of the DHT is naturally derived from (2.5) as [51]:

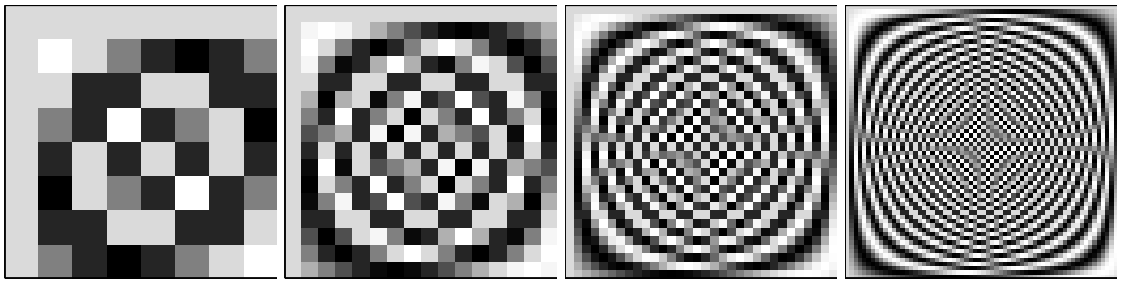
$$\mathbf{H}_N = \Re(\mathbf{F}_N) - \Im(\mathbf{F}_N).$$

On the other hand, the DFT can be obtained from the DHT as follows [51]

$$\Re(\mathbf{F}_N) = \mathcal{E}(\mathbf{H}_N) \text{ and } \Im(\mathbf{F}_N) = \mathcal{O}(\mathbf{H}_N),$$

where $\mathcal{E}(\cdot)$ and $\mathcal{O}(\cdot)$ return the even and odd parts of its input, respectively.

The image representations for $N = 8, 16, 32, 64$, are shown in Figure 2.2. As expected, because the DFT and the DHT share similar mathematical definitions, the image patterns shown in Figures 2.2 and 2.1 are comparable, being more evident as N grows.



(a) $\mathbf{H}_8$ (b) $\mathbf{H}_{16}$ (c) $\mathbf{H}_{32}$ (d) $\mathbf{H}_{64}$

Figure 2.2: *Image representation of the DHT matrix for* $N = 8, 16, 32, 64$.

Computational complexity

To transform an input signal, $\mathbf{x}$, using the DHT, the following matrix computation is perfomed [51]:

$$\mathbf{X} = \frac{1}{N}\mathbf{H}_N\mathbf{x}.$$

The arithmetic complexity of the above matrix multiplication requires at most $N^2 + N$ real multiplications and $N(N-1)$ additions.

2.3 Discrete cosine transform

There are eight different DCTs. However, only DCT-II is shown to optimal in decorrelating some class of signals [4]. In this work, we only considered the DCT-II. Then, we refer to the DCT-II simply as DCT.

The DCT has its forward and inverse kernels defined as:

$$\ker(i,k,N) = (1-(1-1/\sqrt{2})\delta_k)\sqrt{\frac{2}{N}}\cos\left(\frac{2\pi i + k\pi}{2N}\right),\ k = 0,1,\ldots,N-1,$$

$$\ker^{-1}(i,k,N) = (1-(1-1/\sqrt{2})\delta_k)\sqrt{\frac{2}{N}}\cos\left(\frac{2\pi k + i\pi}{2N}\right),\ i = 0,1,\ldots,N-1,$$

where

$$\delta_k = \begin{cases} 0, & \text{if } k = 0 \\ 1, & \text{otherwise.} \end{cases}$$

The derivation of the DCT from the KLT is shown below.

The Karhunen–Loève transform

Let $\mathbf{x}$ be a random input vector with zero mean, which represents the input data to be decorrelated, where the superscript $^\top$ indicates the transposition operation. The KLT is a linear transformation represented by an orthogonal matrix $\mathbf{W}$ which decorrelates the variables in $\mathbf{x}$. The decorrelated output vector $\mathbf{y}$, is obtained according to the following operation:

$$\mathbf{y} = \begin{bmatrix} y_0 & y_1 & \dots & y_{N-1} \end{bmatrix}^\top = \mathbf{W}^\top \cdot \mathbf{x}. \tag{2.6}$$

If the transformation $\mathbf{W}^\top$ decorrelates the input variables, then the covariance matrix of the output vector $\mathbf{y}$ is given by the following diagonal matrix [4]:

$$\mathbf{R}_\mathbf{y} = \mathrm{E}\{\mathbf{y} \cdot \mathbf{y}^\top\} = \mathrm{diag}(\lambda_0, \lambda_1, \dots, \lambda_{N-1}), \tag{2.7}$$

where $\mathrm{E}(\cdot)$ represents the expectation operator, $\mathrm{diag}(\cdot)$ is the diagonal matrix generated by its arguments, and

$$\lambda_k = \mathrm{E}\left\{y_k^2\right\}, \quad k = 0, 1, \dots, N-1,$$

are the variances of the vector $\mathbf{y}$.

Replacing (2.6) in (2.7), it is possible to rewrite the covariance matrix of $\mathbf{y}$ as:

$$\mathbf{R}_\mathbf{y} = \mathrm{E}\left\{\mathbf{W}^\top \cdot \mathbf{x} \cdot \mathbf{x}^\top \cdot \mathbf{W}\right\} = \mathbf{W}^\top \cdot \mathrm{E}\left\{\mathbf{x} \cdot \mathbf{x}^\top\right\} \cdot \mathbf{W} = \mathbf{W}^\top \cdot \mathbf{R}_\mathbf{x} \cdot \mathbf{W},$$

where $\mathbf{R}_\mathbf{x}$ is the covariance matrix of $\mathbf{x}$ which, by construction, is real and symmetric [10], [50]. Since $\mathbf{W}$ is intended to be orthogonal, it must satisfy $\mathbf{W}^{-1} = \mathbf{W}^\top$. Thus, we can write:

$$\mathbf{R}_\mathbf{x} \cdot \left[\mathbf{w}_0 | \mathbf{w}_1 | \cdots | \mathbf{w}_{N-1}\right] = \left[\mathbf{w}_0 | \mathbf{w}_1 | \cdots | \mathbf{w}_{N-1}\right] \cdot \mathbf{R}_\mathbf{y}, \tag{2.8}$$

where $\mathbf{w}_k$, $k = 0, 1, \dots, N-1$, represents the kth column of the matrix $\mathbf{W}$. Therefore, expression (2.8) can be rewritten as the following eigenvalue problem:

$$\mathbf{R}_\mathbf{x} \cdot \mathbf{w}_k = \lambda_k \cdot \mathbf{w}_k, \quad k = 0, 1, \dots, N-1. \tag{2.9}$$

Note that the variances coincide with the eigenvalues. Solving (2.9), we obtain the columns of $\mathbf{W}$, which are ordered according to the decreasing order of their respective eigenvalues [4], thus resulting in the KLT.

Derivation of the discrete cosine transform

If the input vector $\mathbf{x}$ is described by a first-order Markovian model with high correlation [49], then the elements of the correlation matrix associated with $\mathbf{x}$ are given by [4], [10]:

$$[\mathbf{R}_\mathbf{x}]_{m,n} = \rho^{|m-n|}, \quad m, n = 0, 1, \dots, N-1, \tag{2.10}$$

where $\rho \in [0,1]$ is the correlation coefficient. Solving (2.9), we find that the mth component of the kth eigenvector $\mathbf{w}_k$, for $k, m = 0, 1, \ldots, N-1$, is given by [4], [52]:

$$c_{k,m} = \sqrt{\frac{2}{N+\lambda_k}} \cdot \sin\left(\mu_k \left[(m+1) - \frac{N+1}{2}\right] + \frac{(k+1)\pi}{2}\right), \tag{2.11}$$

where

$$\lambda_k = \frac{1-\rho^2}{1-2\rho\cos(\mu_k)+\rho^2} \tag{2.12}$$

is the kth eigenvalue associated to $\mathbf{w}_k$ and μ_k, $k = 0, 1, \ldots, N-1$, are the real-valued roots of the trancendental equation in μ

$$\tan(N\mu) = -\frac{(1-\rho^2)\sin(\mu)}{(1+\rho^2)\cos(\mu) - 2\rho}. \tag{2.13}$$

Assuming highly correlated input data, that is, $\rho \approx 1$, we notice that the right side of (2.13) goes to zero. Therefore, the N real-valued positive roots of (2.13) are given by

$$\mu_k = \frac{k\pi}{N}, \quad k = 0, 1, \ldots, N-1.$$

Thus, replacing the values of μ_k in (2.12), we have that $\lambda_k = 0$ for $k \neq 0$. Now, there is only λ_0 left to compute in order to obtain a closed expression for (2.11). From [53, p. 251], we have that the trace [4] of $\mathbf{R_x}$, defined as

$$\mathrm{tr}(\mathbf{R_x}) = \sum_{n=0}^{N-1} [\mathbf{R_x}]_{n,n},$$

equals the sum of the N eigenvalues. From (2.10), we have that $\mathrm{tr}(\mathbf{R_x}) = N$. Thus

$$\mathrm{tr}(\mathbf{R_y}) = \sum_{k=0}^{N-1} \lambda_k = \lambda_0 = \mathrm{tr}(\mathbf{R_x}) = N \quad \therefore \quad \lambda_0 = N.$$

Finally, we obtain that

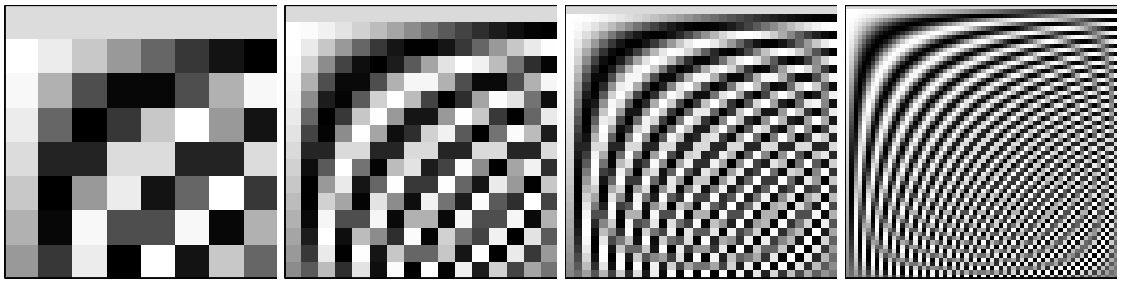
$$\begin{aligned} c_{0,m} &= \frac{1}{\sqrt{N}}, \quad k = 0, \\ c_{k,m} &= \sqrt{\frac{2}{N}} \sin\left(\frac{k(2m+1)}{2N} + \frac{\pi}{2}\right) = \sqrt{\frac{2}{N}} \cos\left(\frac{(2m+1)k\pi}{2N}\right), \quad k \neq 0. \end{aligned}$$

Introducing a constant α_k, we can combine the equations above, obtaining

$$c_{k,m} = \sqrt{\frac{2}{N}} \alpha_k \cos\left(\frac{(2m+1)k\pi}{2N}\right), \tag{2.14}$$

where $\alpha_0 = 1/\sqrt{2}$ and $\alpha_k = 1$, if $k \neq 0$.

The linear transformation whose matrix has elements defined as in (2.14) is called the discrete cosine transform [54], [55]. Therefore, the DCT is asymptotically equivalent to the



(a) $\mathbf{C}_8$ (b) $\mathbf{C}_{16}$ (c) $\mathbf{C}_{32}$ (d) $\mathbf{C}_{64}$

Figure 2.3: *Image representation of the DCT matrix for* $N = 8, 16, 32, 64$.

KLT when $\rho \to 1$. Such relationship justifies the good decorrelation and energy compression properties of the DCT when the input data follows a highly correlated first order stationary Markovian process.

Similar to the DFT and the DHT, the DCT matrix, $\mathbf{C}_N$, also shows some patterns that that are easier to see in its image representation. Figure 2.3 shows those images.

Computational complexity

The computation of the DCT transform of an input signal $\mathbf{x}$,

$$\mathbf{X} = \mathbf{C}_N \cdot \mathbf{x}$$

requires at most N^2 multiplications and $N(N-1)$ additions.

The arithmetic computational complexities of the DFT, DHT and DCT discussed here refer only to its direct implementation. In practice, fast algorithms are employed, reducing the arithmetic cost significantly.

Chapter 3

Fast algorithms and approximations for the 8-point DCT

In this chapter, we present some fast algorithms and low complexity approximations for the DCT found in literature.

3.1 Fast Algorithms

Let $\mathbf{C}_8$ be the 8-point DCT matrix. Because of its symmetries, $\mathbf{C}_8$ can be represented in the following way:

$$\mathbf{C}_8 = \frac{1}{2} \cdot \begin{bmatrix} \gamma_3 & \gamma_3 & \gamma_3 & \gamma_3 & \gamma_3 & \gamma_3 & \gamma_3 & \gamma_3 \\ \gamma_0 & \gamma_2 & \gamma_4 & \gamma_6 & -\gamma_6 & -\gamma_4 & -\gamma_2 & -\gamma_0 \\ \gamma_1 & \gamma_5 & -\gamma_5 & -\gamma_1 & -\gamma_1 & -\gamma_5 & \gamma_5 & \gamma_1 \\ \gamma_2 & -\gamma_6 & -\gamma_0 & -\gamma_4 & \gamma_4 & \gamma_0 & \gamma_6 & -\gamma_2 \\ \gamma_3 & -\gamma_3 & -\gamma_3 & \gamma_3 & \gamma_3 & -\gamma_3 & -\gamma_3 & \gamma_3 \\ \gamma_4 & -\gamma_0 & \gamma_6 & \gamma_2 & -\gamma_2 & -\gamma_6 & \gamma_0 & -\gamma_4 \\ \gamma_5 & -\gamma_1 & \gamma_1 & -\gamma_5 & -\gamma_5 & \gamma_1 & -\gamma_1 & \gamma_5 \\ \gamma_6 & -\gamma_4 & \gamma_2 & -\gamma_0 & \gamma_0 & -\gamma_2 & \gamma_4 & -\gamma_6 \end{bmatrix},$$

where

$$\gamma_0 = \frac{\sqrt{2+\sqrt{2+\sqrt{2}}}}{2} \approx 0{,}9808\ldots, \quad \gamma_1 = \frac{\sqrt{2}}{2} \approx 0{,}707\ldots,$$
$$\gamma_2 = \frac{\sqrt{2+\sqrt{2-\sqrt{2}}}}{2} \approx 0{,}8315\ldots, \quad \gamma_3 = \frac{\sqrt{2+\sqrt{2}}}{2} \approx 0{,}9239\ldots,$$
$$\gamma_4 = \frac{\sqrt{2-\sqrt{2-\sqrt{2}}}}{2} \approx 0{,}5556\ldots, \quad \gamma_5 = \frac{\sqrt{2-\sqrt{2}}}{2} \approx 0{,}3827\ldots,$$
$$\gamma_6 = \frac{\sqrt{2-\sqrt{2+\sqrt{2}}}}{2} \approx 0{,}1951\ldots$$

Thus, an input vector of length 8 might have its components decorrelated by the following expression:

$$\mathbf{X} = \mathbf{C}_8 \cdot \mathbf{x},$$

where $\mathbf{X}$ is the decorrelated vector.

Explicitly, we have that

$$\begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ X_7 \end{bmatrix} = \begin{bmatrix} c_{0,0} & c_{0,1} & \cdots & c_{0,7} \\ c_{1,0} & c_{1,1} & \cdots & c_{1,7} \\ \vdots & \vdots & \ddots & \vdots \\ c_{7,0} & c_{7,1} & \cdots & c_{7,7} \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_7 \end{bmatrix} = \begin{bmatrix} c_{0,0}x_0 + c_{0,1}x_1 + \cdots + c_{0,7}x_7 \\ c_{1,0}x_0 + c_{1,1}x_1 + \cdots + c_{1,7}x_7 \\ \vdots \\ c_{7,0}x_0 + c_{7,1}x_1 + \cdots + c_{7,7}x_7 \end{bmatrix}. \tag{3.1}$$

Therefore, the arithmetic cost to decorrelate the input vector using $\mathbf{C}_8$ is 64 multiplications and 54 additions.

Besides having a closed form, an import factor for the usage of the DCT is the existence of fast algorithms that allows its efficient calculation. Classic fast algorithms for the computation of the DCT length 8 include (i) Yuan *et al.* [56], (ii) Arai *et al.* [54], (iii) Chen *et al.* [57], (iv) Feig e Winograd [58] and Loeffler *et al.* [59]. The arithmetic cost of those and other methods is listed in Table 3.1.

The theoretical minimum of multiplicative complexity for this length is 11 multiplications, which is reached, for example, by the Loeffler *et al.* [59] algorithm. This result is obtained when we consider: (i) the computation of the DCT as a cyclical convolution, and (ii) the results presented in [75], as demonstrated in [27].

Table 3.1: *Arithmetic cost of the fast algorithms for the exact DCT of length 8*

Algorithm	Multiplication	Additions
Loeffler *et al.* [59]–[61]	11	29
Wang [62]	13	29
Suehiro [63]	12	29
Yuan *et al.* [56], [64]	12	29
Lee [65]–[67]	12	29
Hou [68]–[70]	12	29
Arai *et al.* [54], [60], [71]	13	29
Chen *et al.* [57], [60], [72]	16	26
Vetterli [73]	12	29
Feig–Winograd [58], [74]	22	28

3.2 Matrix approximations

As shown on the previous section, the entries of the DCT matrix are irrational quantities. Thus, given the limited precision of computers, its practical implementation with exact numeric precision it is unfeasible [39]. In this sense, the fast algorithms previously mentioned are implemented by means of truncation and/or rounding of its coefficients with its precision defined according to the desired application [76]. Despite substantially reducing the computational cost of its implementation, the fast algorithms for the DCT considered do not eliminate the need for the use of float point arithmetic. The cost of the elementary arithmetic operations in float point numeric representation are usually bigger than the cost from operations in fixed point arithmetic or simpler representations [4]. For this reason, the hardware implementation using float point arithmetic requires greater consumption of power and area resources. Additionally, given the maturity of the area of fast algorithms, there is little space for improvement over the ones in the literature.

Other approach to further reduce the computational cost of the DCT computation is the use of matrix approximations [48], [49]. Such approximations are, basically, matrices with low computational cost that have similar mathematical structure to the exact transforms. That is, let $\mathbf{C}_N$ be the DCT of length N, an approximation for $\mathbf{C}_N$, $\widehat{\mathbf{C}}_N$, is a matrix such that

$$\widehat{\mathbf{X}} = \widehat{\mathbf{C}}_N \cdot \mathbf{x} \approx \mathbf{C}_N \cdot \mathbf{x} = \mathbf{X}.$$

Thus, $\widehat{\mathbf{X}} \approx \mathbf{X}$ according to some criteria, as, for example, proximity or coding measures [4]. An approximation for $\mathbf{C}$ can be obtained from a low complexity multiplierless matrix $\mathbf{T}$. That

is, a matrix whose elements are such as zeros or powers of two, since in binary arithmetic multiplications by powers of two represent only bit shifting. Multiplication by elements like the ones described are called trivial multiplications [6].

Next, we present the matrix approximations existing in literature for the DCT.

3.2.1 DCT approximations

Several approximations for $\mathbf{C}_8$ can be found in literature. Some of those are presented below.

The Walsh-Hadamard transform: The order N Walsh-Hadamard transform (WHT) [77] is given by a binary $N \times N$ matrix, $\mathbf{T}_{\text{WHT–}N}$, with entries in $\{\pm 1\}$ that satisfies:

$$\mathbf{T}_{\text{WHT–}N} \cdot \mathbf{T}_{\text{WHT–}N}^{\top} = N \cdot \mathbf{I}_N,$$

where $\mathbf{I}_N$ represents the identity matrix. The WHT of length 8 is given by:

$$\mathbf{T}_{\text{WHT–8}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix}$$

The WHT is used in image processing due to its good performance and simplicity of implementation. Then, even though the WHT was not proposed as an approximation for the DCT, it is used as an alternative to the DCT.

The signed DCT (SCDT): The first matrix in the literature proposed as an approximation for the DCT was introduced by Haweel in [78]. The signed DCT (SDCT) is a non orthogonal matrix obtained from the application of the sign function to each element of $\mathbf{C}_8$. The sign function is given by $\text{sign}(x) = |x|/x$, $x \neq 0$ and $\text{sign}(0) = 0$. Thus, the low complexity matrix associated to the 8-point SDCT is given by:

$$\mathbf{T}_{\text{SDCT}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix}.$$

The level 1 approximation by Lengwehasatit and Ortega: Lengwehasatit and Ortega proposed five levels of approximation for the DCT based on the input signal features [79]. The

level one approximation is generated by the low complexity orthogonal matrix below:

$$\mathbf{T}_{\mathrm{LO}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & -1 & -1 & -1 \\ 1 & \frac{1}{2} & -\frac{1}{2} & -1 & -1 & -\frac{1}{2} & \frac{1}{2} & 1 \\ 1 & 0 & -1 & -1 & 1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 \\ \frac{1}{2} & -1 & 1 & -\frac{1}{2} & -\frac{1}{2} & 1 & -1 & \frac{1}{2} \\ 0 & -1 & 1 & -1 & 1 & -1 & 1 & 0 \end{bmatrix}.$$

The series of approximations BAS: The series of approximations BAS was proposed by Bouguezel, Ahmad and Swamy [80]–[85]. Many of these approximations were obtained from SDCT modifications [86]. Table 3.2 displays the matrices considered in this work.

Table 3.2: *BAS approximations for* $\mathbf{C}_8$

Transform	Matrix	Orthogonal?
$\mathbf{T}_{\mathrm{BAS-1}}$ [80]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & \frac{1}{2} & -\frac{1}{2} & -1 & -1 & -\frac{1}{2} & \frac{1}{2} & 1 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 \\ \frac{1}{2} & -1 & 1 & -\frac{1}{2} & -\frac{1}{2} & 1 & -1 & \frac{1}{2} \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \end{bmatrix}$	Yes
$\mathbf{T}_{\mathrm{BAS-2}}$ [81]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 0 & -1 & 0 & 0 & 1 & 0 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & 0 & 0 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix}$	No
$\mathbf{T}_{\mathrm{BAS-3}}$ [82]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \end{bmatrix}$	Yes
$\mathbf{T}_{\mathrm{BAS-4}}$ [83]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 2 & 1 & -1 & -2 & -2 & -1 & 1 & 2 \\ 2 & 1 & -1 & -2 & 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -2 & 2 & -1 & -1 & 2 & -2 & 1 \\ 1 & -2 & 2 & -1 & 1 & -2 & 2 & -1 \end{bmatrix}$	Yes
$\mathbf{T}_{\mathrm{BAS-5}}$ [84]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & 0 & 0 & -1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & -1 & 1 & 0 & 0 & 1 & -1 & 0 \end{bmatrix}$	No
$\mathbf{T}_{\mathrm{BAS-6}}$ [85]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \end{bmatrix}$	Yes

The rounded DCT (RDCT): Given $x \in \mathbb{R}$, let $\lfloor x \rfloor$ be the largest integer that does not exceed x. The round function as implemented in Matlab/Octave, is defined by: $\mathrm{round}(x) =$

$\text{sign}(x) \cdot \lfloor x + 0{,}5 \rfloor$. Applied to matrices, the round function operated element wise.

The rounded DCT was proposed by Cintra and Bayer em [48]. The low complexity orthogonal matrix RDCT is obtained by the application of the rounding function to the DCT as follows:

$$\mathbf{T}_{\text{RDCT}} = \text{round}(2 \cdot \mathbf{C}) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & -1 & -1 & -1 \\ 1 & 0 & 0 & -1 & -1 & 0 & 0 & 1 \\ 1 & 0 & -1 & -1 & 1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 \\ 0 & -1 & 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & -1 & 1 & -1 & 1 & 0 \end{bmatrix}.$$

The modified RDCT: The modified RDCT (MRDCT) was introduced by Bayer and Cintra in [87]. The MRDCT is an orthogonal matrix obtained by replacing some elements of the RDCT matrix by zeros. Its explicit form is presented next:

$$\mathbf{T}_{\text{MRDCT}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & -1 & -1 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

The difference matrix is given by:

$$\mathbf{T}_{\text{RDCT}} - \mathbf{T}_{\text{MRDCT}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & -1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 \end{bmatrix}.$$

The series of approximations CBT: In [49], Cintra, Bayer and Tablada obtained a series of approximations, which we are going to refer to as CBT, by means of applying several different rounding approximations to the DCT matrix. The matrices introduced in [49] are shown in Table 3.3.

Table 3.3: *Series of approximations CBT for* $\mathbf{C}_8$

Transform	Matrix	Orthogonal?
$\mathbf{T}_{\text{CBT–1}}$ [49]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 0 & 0 & -1 & -1 & -2 \\ 0 & 1 & -1 & 0 & 0 & -1 & 1 & 0 \\ 1 & 0 & -2 & -1 & 1 & 2 & 0 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -2 & 0 & 1 & -1 & 0 & 2 & -1 \\ 1 & 0 & 0 & -1 & -1 & 0 & 0 & 1 \\ 0 & -1 & 1 & -2 & 2 & -1 & 1 & 0 \end{bmatrix}$	Yes
$\mathbf{T}_{\text{CBT–2}}$ [49]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 0 & 0 & -1 & -1 & -2 \\ 2 & 0 & 0 & -2 & -2 & 0 & 0 & 2 \\ 1 & 0 & -2 & -1 & 1 & 2 & 0 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -2 & 0 & 1 & -1 & 0 & 2 & -1 \\ 0 & -2 & 2 & 0 & 0 & 2 & -2 & 0 \\ 0 & -1 & 1 & -2 & 2 & -1 & 1 & 0 \end{bmatrix}$	Yes
$\mathbf{T}_{\text{CBT–3}}$ [49]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 0 & -1 & -1 & 1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 0 & -1 & 1 & -1 & 1 & -1 & 1 & 0 \end{bmatrix}$	Yes
$\mathbf{T}_{\text{CBT–4}}$ [49]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 0 & 0 & -1 & -1 & -2 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 0 & -2 & -1 & 1 & 2 & 0 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -2 & 0 & 1 & -1 & 0 & 2 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 0 & -1 & 1 & -2 & 2 & -1 & 1 & 0 \end{bmatrix}$	Yes
$\mathbf{T}_{\text{CBT–5}}$ [49], [88]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 0 & 0 & -1 & -1 & -2 \\ 2 & 1 & -1 & -2 & -2 & -1 & 1 & 2 \\ 1 & 0 & -2 & -1 & 1 & 2 & 0 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -2 & 0 & 1 & -1 & 0 & 2 & -1 \\ 1 & -2 & 2 & -1 & -1 & 2 & -2 & 1 \\ 0 & -1 & 1 & -2 & 2 & -1 & 1 & 0 \end{bmatrix}$	Yes
$\mathbf{T}_{\text{CBT–6}}$ [49]	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & 0 & 0 & -1 & -1 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 & 0 & 1 & 0 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 0 & -1 & 0 & 1 & -1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 \end{bmatrix}$	No
$\mathbf{T}_{\text{CBT–7}}$ [49]	$\begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 1 & 1 & -1 & -1 & -2 & -2 \\ 2 & 1 & -1 & -2 & -2 & -1 & 1 & 2 \\ 2 & -1 & -2 & -1 & 1 & 2 & 1 & -2 \\ 2 & -2 & -2 & 2 & 2 & -2 & -2 & -2 \\ 1 & -2 & 1 & 2 & -2 & -1 & 2 & -1 \\ 1 & -2 & 2 & -1 & -1 & 2 & -2 & 1 \\ 1 & -1 & 2 & -2 & 2 & -2 & 1 & -1 \end{bmatrix}$	No

Chapter 4

Search method

Approximate transforms are low-complexity transformations capable of preserving desirable properties of the exact transform. The design of approximate transforms is often based on structural aspects of the original, exact transforms, such as: symmetries [89], fast algorithms [59], [68], parametrization [58], and numerical properties [56]. In a general manner, a low-complexity factor $\mathbf{T}$ of an approximation for a trigonometric transform, $\mathbf{C}$, is obtained by solving the optimization problem below:

$$\mathbf{T} = \arg\min_{\mathbf{T}'} \text{approx}(\mathbf{T}', \mathbf{C}),$$

where $\text{approx}(\cdot, \cdot)$ is a specific objective function—such as proximity or performance measures [4]—submitted to several constraints, such as orthogonality and low complexity of the candidate matrices $\mathbf{T}'$.

In this chapter, we introduce a new search method for matrix approximation. Notice that, even though the development of the method was motivated by the trigonometric transforms, the proposed method are completely general and might be used to approximate any matrix. Although our main goal is to find new approximations for the DCT, we are going to explore in this chapter the ramifications of the proposed method when applied to the DHT and DFT also.

4.1 Overall structure and initial concepts

Let $\mathbf{A}$ be an arbitrary $N \times M$ matrix with elements in $\mathbb{R}$, and $\mathbf{a}_k = \begin{bmatrix} a_{k,0} & a_{k,1} & \dots & a_{k,M-1} \end{bmatrix}$, $k = 0, 1, \dots, N-1$, be a row vector that represents the kth row of $\mathbf{A}$. Note that $\mathbf{A}$ might be

described by its rows as follows:

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \vdots \\ \mathbf{a}_{N-1} \end{bmatrix}.$$

Aiming at finding a low complexity approximation $\mathbf{T}$ for $\mathbf{A}$, we broke down the problem of approximating the whole matrix into the problem of approximating its rows by low complexity row vectors. Such heuristic can be categorized as greedy [90]. Therefore, our goal is to derive integer low complexity matrices

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_0 \\ \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_{N-1} \end{bmatrix},$$

such that its rows $\mathbf{t}_k$, $k = 0, 1, \ldots, N-1$, satisfy

$$\mathbf{t}_k = \arg\min_{\mathbf{t} \in \mathcal{D}_{\mathcal{P}}} \text{error}(\mathbf{t}, \mathbf{a}_k), \quad k = 0, 1, \ldots, N-1, \tag{4.1}$$

where $\mathcal{D}_{\mathcal{P}}$ is the search space, presented next.

4.1.1 Search Space

In order to obtain a low-complexity matrix $\mathbf{T}$, its entries must be computationally simple [4], [6]. We define the search space as the collection of M-point row vectors whose entries are in a set, say $\mathcal{P}$, of low-complexity elements. That is, the search space $\mathcal{D}_{\mathcal{P}}$ is composed by all the possible permutations of length M of the elements in $\mathcal{P}$. Therefore, the cardinality of the search space is given by $|\mathcal{D}_{\mathcal{P}}| = |\mathcal{P}|^M$. A particular vector in $\mathcal{D}_{\mathcal{P}}$ is denoted by $\mathcal{D}_{\mathcal{P}}(i)$, $i = 1, 2, \ldots, |\mathcal{D}_{\mathcal{P}}|$. Some choices for $\mathcal{P}$ include: $\mathcal{P}_1 = \{0, \pm 1\}$ and $\mathcal{P}_2 = \{0, \pm 1, \pm 2\}$.

For example, if considering approximating an 8×8 matrix, Tables 4.1 and 4.2 display some elements of the search spaces $\mathcal{D}_{\mathcal{P}_1}$ and $\mathcal{D}_{\mathcal{P}_2}$. These search spaces have cardinality $|\mathcal{P}_1|^8 = 3^8 = 6,561$ and $|\mathcal{P}_2|^8 = 5^8 = 390,625$ elements, respectively.

4.1.2 Objective Function

The problem posed in (4.1) requires the identification of an error function to quantify the "distance" between the candidate row vectors from $\mathcal{D}_{\mathcal{P}}$ and the rows of the exact matrix $\mathbf{A}$.

Table 4.1: *Examples of approximated vectors from the search space $\mathcal{D}_{\mathcal{P}_1}$*

i	$\mathcal{D}_{\mathcal{P}_1}(i)$
1	$\begin{bmatrix}-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1\end{bmatrix}$
2	$\begin{bmatrix}-1 & -1 & -1 & -1 & -1 & -1 & -1 & 0\end{bmatrix}$
$\vdots$	$\vdots$
3200	$\begin{bmatrix}0 & 0 & 0 & -1 & 0 & 0 & 0 & 1\end{bmatrix}$
3201	$\begin{bmatrix}0 & 0 & 0 & -1 & 0 & 0 & 1 & -1\end{bmatrix}$
$\vdots$	$\vdots$
6560	$\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1 & 1 & 0\end{bmatrix}$
6561	$\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\end{bmatrix}$

Table 4.2: *Examples of approximated vectors from the search space $\mathcal{D}_{\mathcal{P}_2}$*

i	$\mathcal{D}_{\mathcal{P}_2}(i)$
1	$\begin{bmatrix}-2 & -2 & -2 & -2 & -2 & -2 & -2 & -2\end{bmatrix}$
2	$\begin{bmatrix}-2 & -2 & -2 & -2 & -2 & -2 & -2 & -1\end{bmatrix}$
$\vdots$	$\vdots$
150000	$\begin{bmatrix}-1 & 2 & 1 & -2 & -2 & -2 & -2 & -2\end{bmatrix}$
150001	$\begin{bmatrix}-1 & 2 & 1 & -2 & -2 & -2 & -2 & -1\end{bmatrix}$
$\vdots$	$\vdots$
390624	$\begin{bmatrix}2 & 2 & 2 & 2 & 2 & 2 & 2 & 1\end{bmatrix}$
390625	$\begin{bmatrix}2 & 2 & 2 & 2 & 2 & 2 & 2 & 2\end{bmatrix}$

Related literature often consider error functions based on matrix norms [91], proximity to orthogonality [86], and coding performance [4].

In this work, we propose the utilization of a distance based on the angle between vectors as the objective function to be minimized. Let $\mathbf{u}$ and $\mathbf{v}$ be two M-dimensional vectors defined over $\mathbb{R}^M$. The angle between vectors is simply given by:

$$\text{angle}(\mathbf{u}, \mathbf{v}) = \arccos\left(\frac{\langle \mathbf{u}, \mathbf{v} \rangle}{\|\mathbf{u}\| \cdot \|\mathbf{v}\|}\right), \tag{4.2}$$

where $\langle \cdot, \cdot \rangle$ is the inner product and $\| \cdot \|$ indicates the norm induced by the inner product [53].

4.2 Angle based method

Based on the previous concepts, we are able to propose the angle based method, which is based on the optimization problem stated as follows:

$$\mathbf{t}_k = \arg\min_{\mathbf{t}\in\mathcal{D}_{\mathcal{P}}} \text{angle}(\mathbf{a}_k, \mathbf{t}), \quad k = 0, 1, \ldots, N-1. \tag{4.3}$$

First, we determine the set $\mathcal{P}$ and built the search space $\mathcal{D}_{\mathcal{P}}$. Then, for each row of $\mathbf{A}$, we generate a subset of the search space, $\mathcal{D}_{\mathcal{P}}^{(k)}$, $k = 0, 1, \ldots, N-1$, containing all the vectors in $\mathcal{D}_{\mathcal{P}}$ that are solutions to the problem in (4.3). Lastly, each approximate matrix is obtained as a combination of the vectors in $\mathcal{D}_{\mathcal{P}}^{(k)}$, $k = 0, 1, \ldots, N-1$. The number of matrices obtained is given by $\prod_{k=0}^{N-1} |\mathcal{D}_{\mathcal{P}}^{(k)}|$. Therefore,

$$\mathbf{T}^{(i)} = \begin{bmatrix} \mathbf{t}_0 \\ \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_{N-1} \end{bmatrix}, \quad i = 1, 2, \ldots, \prod_{k=0}^{N-1} |\mathcal{D}_{\mathcal{P}}^{(k)}|,$$

where $\mathbf{t}_k \in \mathcal{D}_{\mathcal{P}}^{(k)}$.

Example 4.1

Let $\mathbf{A}$ *be a* 4×4 *matrix,* $|\mathcal{D}_{\mathcal{P}}^{(1)}| = |\mathcal{D}_{\mathcal{P}}^{(3)}| = 1$*, and* $|\mathcal{D}_{\mathcal{P}}^{(2)}| = |\mathcal{D}_{\mathcal{P}}^{(4)}| = 2$*. In this case, we obtain* $\prod_{k=1}^{4} |\mathcal{D}_{\mathcal{P}}^{(k)}| = 1 \cdot 2 \cdot 1 \cdot 2 = 4$ *approximate matrices, given by:*

$$\mathbf{T}^{(1)} = \begin{bmatrix} \mathcal{D}_{\mathcal{P}}^{(1)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(2)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(3)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(4)}(1) \end{bmatrix}, \quad \mathbf{T}^{(2)} = \begin{bmatrix} \mathcal{D}_{\mathcal{P}}^{(1)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(2)}(2) \\ \mathcal{D}_{\mathcal{P}}^{(3)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(4)}(1) \end{bmatrix}, \quad \mathbf{T}^{(3)} = \begin{bmatrix} \mathcal{D}_{\mathcal{P}}^{(1)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(2)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(3)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(4)}(2) \end{bmatrix}, \quad \mathbf{T}^{(4)} = \begin{bmatrix} \mathcal{D}_{\mathcal{P}}^{(1)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(2)}(2) \\ \mathcal{D}_{\mathcal{P}}^{(3)}(1) \\ \mathcal{D}_{\mathcal{P}}^{(4)}(2) \end{bmatrix},$$

where $\mathcal{D}_{\mathcal{P}}^{(k)}(i)$*,* $k = 0, 1, \ldots, N-1$*,* $i = 1, 2, \ldots, |\mathcal{D}_{\mathcal{P}}^{(k)}|$*, represents the* i*th vector in* $\mathcal{D}_{\mathcal{P}}^{(k)}$*.* □

The procedure for the angle based method is shown in Algorithm 1.

4.3 Angle based method - restricted to orthogonality

Note that the previous method does not guarantee that the obtained matrices are orthogonal. However, orthogonality is a desirable feature. So in order to ensure that, we need to consider some other factors, discussed below.

Algorithm 1 Pseudo algorithm for angle based method

Input: $\mathbf{A}$, $\mathcal{D}_{\mathcal{P}}$

Output: **approximations** (3 dimensional array containing all the obtained approximate matrices)

```
for k ← 0, 1, ..., N − 1 do
    angles ← null vector of length |D_P|
    for i ← 1, 2, ..., |D_P| do
        angles(i) ← angle(a_k, D_P(i));
    end for
    indexes ← indexes of the vectors in D_P for which angles = min(angles);
    D_P^(k) ← D_P(indexes);
end for
approximations ← Null array with dimensions N × M × ∏_{k=0}^{N−1} |D_P^(k)|;
approximations ← All combinations of the vectors in D_P^(k), k = 0, 1, ..., N − 1;
```

4.3.1 Orthogonality

Definition 4.1 – Orthogonality

We say that $\mathbf{A}$ *is an orthogonal matrix if* $\mathbf{A} \cdot \mathbf{A}^\top = \mathbf{D}$, *where* $\mathbf{D}$ *is a diagonal matrix.* □

Definition 4.2 – Orthonormality

If $\mathbf{A} \cdot \mathbf{A}^\top$ *is the identity matrix, then* $\mathbf{A}$ *is said to be orthonormal.* □

If $\mathbf{T}$ is orthogonal, then its inverse is given by $\mathbf{T}^{-1} = \mathbf{T}^\top \cdot \mathbf{D}^{-1}$, where $\mathbf{D}$ the diagonal matrix resulting from $\mathbf{T} \cdot \mathbf{T}^\top$. In particular, if $\mathbf{T}$ is orthonormal, then $\mathbf{T}^{-1} = \mathbf{T}^\top$. As a consequence of that, if $\mathbf{T}$ is orthogonal and can be decomposed into the product of p matrices, that is,

$$\mathbf{T} = \mathbf{A}_1 \times \mathbf{A}_2 \times \cdots \times \mathbf{A}_p,$$

then

$$\mathbf{T}^{-1} = \mathbf{T}^\top \times \mathbf{D}^{-1} = (\mathbf{A}_1 \times \mathbf{A}_2 \times \cdots \times \mathbf{A}_p)^\top \times \mathbf{D}^{-1} = \mathbf{A}_p^\top \times \mathbf{A}_{p-1}^\top \times \cdots \times \mathbf{A}_1^\top \times \mathbf{D}^{-1}.$$

It means that a fast algorithm for $\mathbf{T}$ can be easily converted into a fast algorithm for $\mathbf{T}^{-1}$. Also, since $\mathbf{T}$ is a low-complexity matrix, $\mathbf{T}^{-1}$ would be low-complexity as well.

4.3.2 Search sequence

For the unrestricted method, since there is no constraints to the optimization problem, the rows are approximated independently from each other. However, if considering orthogonality

as a constraint, we define a dependency relation among the rows. Hence, the sequence in which we approximate the rows must be considered.

There are N rows to be approximated. One way of doing it is—under some criteria–to approximate the rows in the order they are displayed, that is, approximate row 1, then row 2, constrained to orthogonality with row 1, then row 3, constrained to orthogonality with rows 1 and 2, and so on. That procedure corresponds to the sequence $\wp_1 = (1, 2, 3, \ldots, N)$. However, this is only a particular search sequence. Therefore, for a systematic procedure, we must consider all the $N!$ possible permutations of $\wp_1$. Let $\wp_m$, $m = 1, 2, \ldots, N!$, be the mth permutation of $\wp_1$, and $\wp_m(k)$, $k = 0, 1, \ldots, N-1$, be a particular element of $\wp_m$. For example, if $N = 8$, there are $N! = 40320$ possible search sequences. In this case, we have $\wp_{1250} = (1, 3, 7, 6, 5, 4, 8, 2)$ and $\wp_{1250}(2) = 7$.

4.3.3 Optimization problem

Taking those last factors in consideration, we can now determine the optimization problem in which to base the restricted to orthogonality version of the angle based algorithm. Fixing a search sequence $\wp_m$, the optimization problem is stated as follows:

$$\mathbf{t}_{\wp_m(k)} = \arg\min_{\mathbf{t} \in \mathcal{D}_{\mathcal{P}}} \text{angle}(\mathbf{a}_{\wp_m(k)}, \mathbf{t}), \quad k = 0, 1, \ldots, N-1, \tag{4.4}$$

subject to

$$\langle \mathbf{t}_{\wp_m(i)}, \mathbf{t}_{\wp_m(j)} \rangle = 0, \quad i \neq j. \tag{4.5}$$

The solution of the problem above returns N row vectors $\mathbf{t}_{\wp_m(0)}, \mathbf{t}_{\wp_m(2)}, \ldots, \mathbf{t}_{\wp_m(N-1)}$ that are used as the rows of the approximate matrix $\mathbf{T}$.

Algorithm 2 displays the procedure for the restricted to orthogonality version of the angle based method.

4.4 Complex adaptation

The proposed method is based on the calculation of the angle between two vectors whose elements are in $\mathbb{R}$: a row of the input matrix and the candidate approximate vector. However, one of the transforms we intent to approximate is the DFT, which has its coefficients defined over the complex space, $\mathbb{C}$. In this case, some adaptations need to be made.

Algorithm 2 Algorithm for the angle based method restricted to orthogonality

Input: $\mathbf{A}$; $\mathcal{D}_{\mathcal{P}}$; $\wp$ ($N! \times N$ matrix containing all the possible search sequences).

Output: **approximations** (3 dimensional array containing all the obtained approximate matrices).

```
approximations ← Null array with dimensions N × M × N!;
for m ← 1, 2, ..., N! do
  for k ← 0, 1, ..., N − 1 do
    θ_min ← 2π;
    index ← 1;
    for i ← 1, 2, ..., |D_P| do
      aux ← approximations(:, :, m) · (D_P(i))^⊤
      if sum(aux) = 0 then
        θ ← angle(a_{℘_m(k)}, D_P(i));
        if θ < θ_min then
          θ_min ← θ;
          index ← i;
        end if
      end if
    end for
    approximations(℘_m(k), :, m) ← D_P(index);
  end for
end for
```

Option A

A natural first option is to decompose the complex matrix in its real and complex components and approximate each one using any version (unrestricted or restricted) of the proposed method. Then, the DFT approximations are the combinations of the approximations found for the real and complex components. Observe that, if using the restricted version of the method, the approximations for the real and complex parts are going to be orthogonal, but there is no guarantee that the resulting approximate matrices are orthogonal as well.

Option B

Other option would be to calculate the angle in the complex space. According to Scharnhorst [92], one way of computing the angle between two M-dimensional complex vectors, say $\mathbf{r}$ and $\mathbf{s}$: is by considering its isometric vector space $\mathbb{R}^{2M}$.

Then, the angle between $\mathbf{r}$ and $\mathbf{s}$ is calculated as in (4.2) with

$$\text{angle}(\mathbf{r}, \mathbf{s}) = \text{angle}(\mathbf{r}^{\star}, \mathbf{s}^{\star})$$

where $\mathbf{r}^\star$ and $\mathbf{s}^\star$ are defined in $\mathbb{R}^{2N}$ by the relation

$$r^\star_{2k} = \Re(r_k) \text{ and } r^\star_{2k+1} = \Im(r_k),\ k = 0, 1, \ldots, M-1. \tag{4.6}$$

The inverse operation is given by:

$$r_k = r^\star_{2k} + j r^\star_{2k+1}\ k = 0, 1, \ldots, M-1. \tag{4.7}$$

Let $\mathbf{A}$ be a $N \times M$ matrix whose coefficients are complex, that is, $a_{i,j} \in \mathbb{C}$, $i = 0, 1, \ldots, N-1$, $j = 0, 1, \ldots, M-1$. By performing the mapping on (4.6) for each row of $\mathbf{A}$, we obtain a new real matrix $\mathbf{B}$ with dimensions $N \times 2M$, i.e.

$$\text{Complex } N \times N \text{ matrix } \mathbf{A} \xrightarrow{(4.6)} \text{ Real } N \times 2M \text{ matrix } \mathbf{B}. \tag{4.8}$$

Then, $\mathbf{B}$ can be approximated by any version of the proposed method as they were described earlier. Next, each approximate matrix obtained, $\hat{\mathbf{B}}$, must be converted from real to complex again by applying (4.7) to each of its rows, i.e.

$$\text{Real } N \times 2M \text{ matrix } \hat{\mathbf{B}} \xrightarrow{(4.7)} \text{ Complex } N \times N \text{ matrix } \hat{\mathbf{A}}. \tag{4.9}$$

The matrices obtained from (4.9) are the complex approximations for the input matrix $\mathbf{A}$.

In this case, the restricted version of the proposed method can not guarantee orthogonality of the approximate matrices obtained. The approximations for the real $N \times 2M$ matrix are going to be orthogonal. However, there is nothing that assures that its rows are still going to be orthogonal after being converted back to complex vectors. Also, due the mapping in (4.8), we are now approximating $2M$-dimensional vectors. Therefore, the cardinality of the search space is now given by $|\mathcal{D}_\mathcal{P}| = |\mathcal{P}|^{2M}$.

Table 4.3 summarizes the modifications in each version of the method so they can approximate complex matrices.

4.5 Remarks

Here we list some observations about the proposed method. Some of the following notes are just general considerations while other points are going to be further explored in the next chapter.

General

▷ If $\mathbf{A}$ already has any low complexity rows, they might be previously fixed and any of the methods can be used to approximate only the remaining rows. This reduces processing time;

Table 4.3: *Procedures to approximate complex matrices using the unrestricted and restricted versions of the angle based method*

	Angle based method	
Complex angle	Unrestricted	Restricted to orthogonality
Option A	• Separate **A** into its real and complex components • Run Algorithm 1 for both components • Combine the approximations obtained for the real and complex components	• Separate **A** into its real and complex components • Run Algorithm 2 for both components • Combine the approximations obtained for the real and complex components
Option B	• Apply (4.8) to **A** • Run Algorithm 1 • Apply (4.9) to the approximations obtained	• Apply (4.8) to **A** • Run Algorithm 2 • Apply (4.9) to the approximations obtained

▷ Matrix symmetries may also be explored in order to reduce computational time.

Unrestricted version of the proposed method

▷ Since the unrestricted version of the method does not have to consider the search sequence, it is faster than the restricted version;

▷ The rows are approximated independently, which allows the use of parallelization in order to run it even faster;

▷ By construction, the approximations generated by a specific search space are all different. Although different search spaces may generate the same approximations.

Restricted to orthogonality version of the proposed method

▷ For a particular search sequence, the algorithm may reach a point where it can not find a vector in the search space that is orthogonal to the vectors already fixed in the approximation matrix. From that point on, all the rows still to be approximated are going to be set as null vectors in the approximate matrix;

▷ Some search sequences may generate the same approximation matrix;

▷ As for the two items above, the 3-dimensional array obtained from the method must be "cleaned" in order to eliminate the singular matrices and the repeated ones. Only the remaining matrices are actually valid approximations;

- ▷ Notice in Algorithm 2 we only change the candidate vector in the approximate matrix if the angle between this vector and the matrix row is smaller than the previous minimum angle. By doing so, we fix in the approximated matrix the first vector in the search space which generates that minimum angle. Changing this condition may generate a different approximate matrix.

Chapter 5

Approximation schemes

As pointed on the previous chapter, some features of the matrix we aim at approximating may be used in order to reduce the computational complexity of the approximation process. In this chapter, we study some of these features and define the approximation schemes that can be used to approximate the discrete trigonometric transforms we discussed.

5.1 Search space reduction

As presented on the previous chapter, our search space is built from a set of low complexity elements. Some common choices for this set and the size of the corresponding search spaces when approximating a $N \times M$ matrix are displayed in Table 5.1.

Table 5.1: *Examples of common sets and the size of the corresponding search space*

Set ($\mathcal{P}$)	Size of the corresponding search space
$\{-1, 0, 1\}$	3^M
$\{-2, -1, 0, 1, 2\}$	5^M
$\{-1, -1/2, 0, 1/2, 1\}$,	5^M
$\{-2, -1, -1/2, 0, 1/2, 1, 2\}$	7^M

Observing the sets shown in Table 5.1 we can see that they are all symmetric around zero. And it is important to have those negative and positive elements since the target matrices also have positive and negative entries. In this sense, one way to reduce the size of the search space, would be to consider only the positive part of the sets and add the signs later. Then, we propose the following procedure to approximate an input matrix $\mathbf{A}$:

1. Select the set $\mathcal{P}$ with only zero and positive elements;
2. Approximate $\text{abs}(\mathbf{A})$ using Method I, where $\text{abs}(\cdot)$ returns the absolute value of its input. When applied to matrices, the abs function operates element wise;
3. Define the approximations for $\mathbf{A}$ as

$$\widehat{\mathbf{A}} = \widehat{\text{abs}(\mathbf{A})} \odot \text{sign}(\mathbf{A}), \tag{5.1}$$

where $\widehat{\text{abs}(\mathbf{A})}$ is an approximation obtained from step 2, and $\odot$ represents the element wise multiplication.

By performing the procedure above, the size of the search space is reduced and the sign structure of the input matrix is maintained. As an example, Table 5.2 shows the proportional reduction of the search space when $\mathbf{A}$ is a $N \times 8$ matrix.

Table 5.2: *Reduction of the size of the search space for some sets when $M = 8$*

Set	Size of the original search space (Table 5.1)	Size of the reduced search space	Reduction of the search space
$\{0, 1\}$	$3^8 \approx 6.56 \times 10^3$	$2^8 = 2.56 \times 10^2$	96.10%
$\{0, 1, 2\}$	$5^8 \approx 3.90 \times 10^5$	$3^8 \approx 6.56 \times 10^3$	98.32%
$\{0, 1/2, 1\}$,	$5^8 \approx 3.90 \times 10^5$	$3^8 \approx 6.56 \times 10^3$	98.32%
$\{0, 1/2, 1, 2\}$	$7^8 \approx 5.76 \times 10^6$	$4^8 \approx 6.55 \times 10^4$	98.86%

Note that the procedure proposed above can be used only in association with the unrestricted version of the method. This is due the fact that for the unrestricted version the rows are approximated independently. Observe that, for the proposed procedure, we are approximating a non negative matrix with non negative elements. In this case, two row vectors are orthogonal if, and only if, they have non zero elements in different positions, which causes the inner product to be zero. Thus, if using this procedure associated with the restricted version of the method, we have the following possible situations:

- ▷ If a row is approximated by a vector with more than one non zero element, the output matrix is necessarily going to have at least one all zero row. Which means the matrix is singular and it is not interesting for our proposes;
- ▷ Otherwise, the output matrix is going to be a permuted and/or scaled version of the identity matrix. Which is also not interesting for decorrelation proposes.

Therefore, for the restrited version of the method it is necessary to consider the original sets and the search space remains the same size.

5.2 Fixing low complexity rows

All the transform matrices we are considering have some low complexity rows. In this case, those rows may be previously fixed and any version of the proposed method can be used to approximate the remaining rows. Note that, in this case, the restricted version may also be used because all the transform matrices considered are orthogonal. Then, any set of rows we select are orthogonal among each other and the approximations for the remaining rows are constrained to be orthogonal to the set of rows initially fixed.

DFT and DHT

The DFT and DHT have the same low complexity rows, which is expected given their similar kernels. The kernels of both transforms are built as a combination of cosine and sine functions with their argument being $\frac{2\pi ik}{N}$. Then, for those two transforms, the low complexity rows are the ones for which $k = 0, N/4, N/2, 3N/4$. For each of these values we have that:

- ▷ If $k = 0$, then $\frac{2\pi ik}{N} = 0$;
- ▷ If $k = \frac{N}{4}$, then $\frac{2\pi ik}{N} = \frac{\pi}{2}i$;
- ▷ If $k = \frac{N}{2}$, then $\frac{2\pi ik}{N} = \pi i$;
- ▷ If $k = \frac{3N}{4}$, then $\frac{2\pi ik}{N} = \frac{3\pi}{2}i$.

Table 5.3 displays the sequences generated by the cosine ans sine functions when their argument are the ones obtained above and $i = 0, 1, 2, \ldots$.

As seen in Table 5.3, all the sequences have elements on the low complexity set $\{-1, 0, 1\}$. For the DFT, its real and complex parts are given by the cosine and sine sequences shown in Table 5.3, respectively, as shown in Figure 5.1.

For the DHT, its low complexity rows are going to be the sum of those sequences. The graphic representation of those sequences is shown in Figure 5.2.

DCT

The DCT matrix does not have low complexity rows. However, for $k = 0, \frac{N}{2}$, the DCT rows are scaled low complexity sequences, as show in Table 5.4. In this case, we can define

Table 5.3: *Cosine and sine sequences generated when* $k = 0, N/4, N/2, 3N/4$

k	Argument	Generated sequences	
0	0	$\cos(0)$	$1, 1, 1, 1, 1, 1, 1, \ldots$
		$\sin(0)$	$0, 0, 0, 0, 0, 0, 0, \ldots$
$\frac{N}{4}$	$\frac{\pi}{2}i$	$\cos(\frac{\pi}{2}i)$	$1, 0, -1, 0, 1, 0, -1, \ldots$
		$\sin(\frac{\pi}{2}i)$	$0, 1, 0, -1, 0, 1, 0, \ldots$
$\frac{N}{2}$	πi	$\cos(\pi i)$	$1, -1, 1, -1, 1, -1, 1, \ldots$
		$\sin(\pi i)$	$0, 0, 0, 0, 0, 0, 0, \ldots$
$\frac{3N}{4}$	$\frac{3\pi}{2}i$	$\cos(\frac{3\pi}{2}i)$	$1, 0, -1, 0, 1, 0, -1, \ldots$
		$\sin(\frac{3\pi}{2}i)$	$0, -1, 0, 1, 0, -1, 0, \ldots$

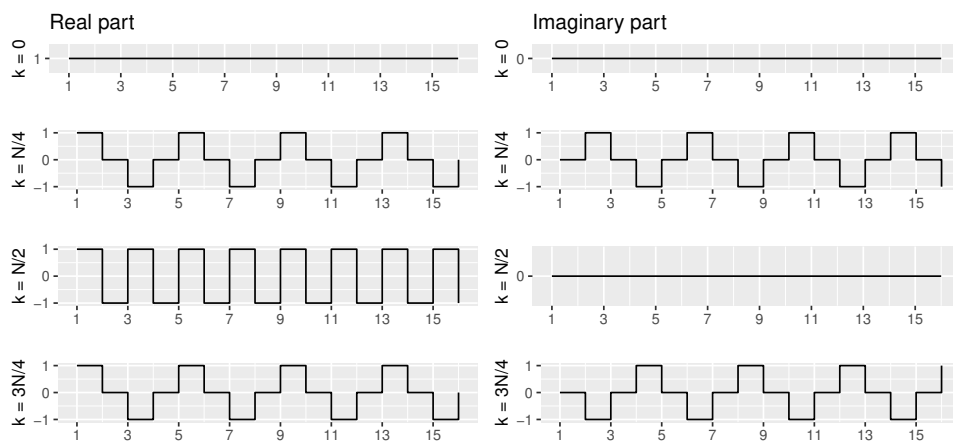


Figure 5.1: *Graphic representation of the real and imaginary parts of the low complexity sequences that form the rows of the DFT matrix for* $k = 0, N/4, N/2, 3N/4$.

Table 5.4: *DCT row sequence for* $k = 0, N/2$

k	$c_{k,m}$	Row sequence ($m = 0, 1, 2, \ldots$)
0	$\frac{1}{\sqrt{N}}$	$\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \ldots = \frac{1}{\sqrt{N}}(1, 1, 1, 1, 1, 1, 1 \ldots)$
$N/2$	$\sqrt{\frac{2}{N}}\cos((2m+1)\frac{\pi}{4})$	$\frac{1}{\sqrt{N}}, -\frac{1}{\sqrt{N}}, -\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, -\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}}, \ldots = \frac{1}{\sqrt{N}}(1, -1, -1, 1, 1, -1, -1 \ldots)$

that those rows as the low complexity sequences in Table 5.4.

5.2.1 Unrestricted version of the method

For the unrestricted version of the proposed method, we can not only fix some rows but combine this with the search space reduction procedure proposed on the previous section. Figures 5.3, 5.4 and 5.5 display the image representation of the absolute value of the transforms

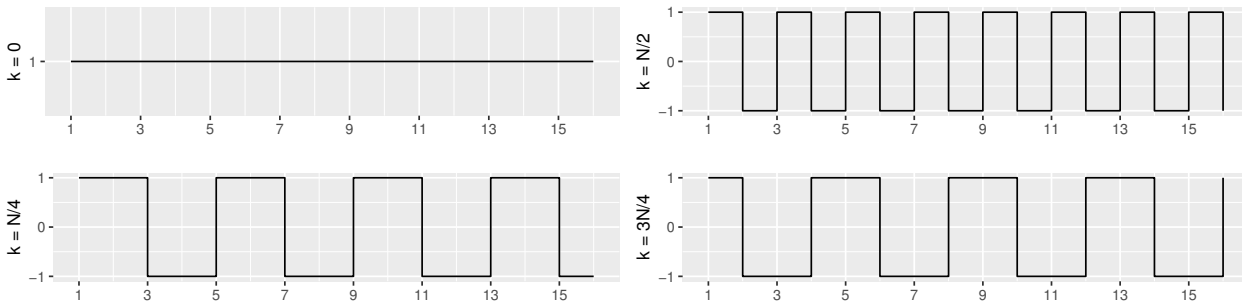
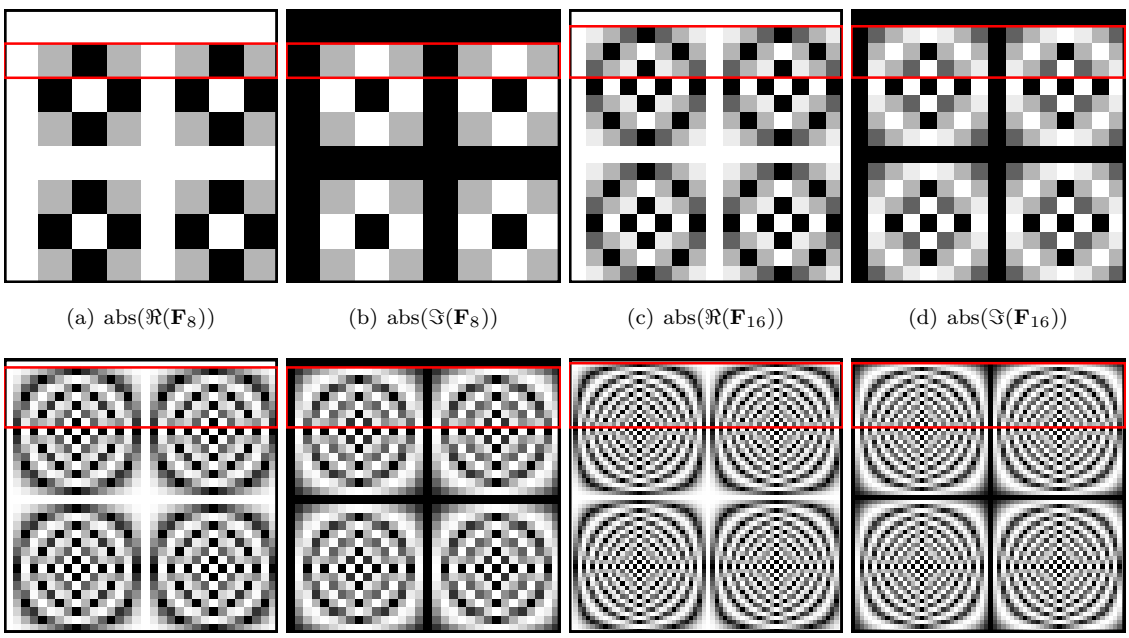


Figure 5.2: *Graphic representation of the low complexity sequences that form the rows of the DHT matrix for* $k = 0, N/4, N/2, 3N/4$.

considered for $N = 8, 16, 32, 64$.

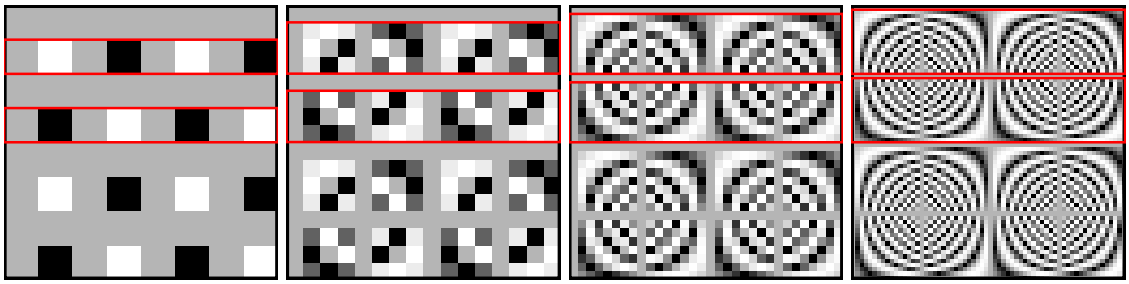


(a) abs($\Re(\mathbf{F}_8)$) (b) abs($\Im(\mathbf{F}_8)$) (c) abs($\Re(\mathbf{F}_{16})$) (d) abs($\Im(\mathbf{F}_{16})$)

(e) abs($\Re(\mathbf{F}_{32})$) (f) abs($\Im(\mathbf{F}_{32})$) (g) abs($\Re(\mathbf{F}_{64})$) (h) abs($\Im(\mathbf{F}_{64})$)

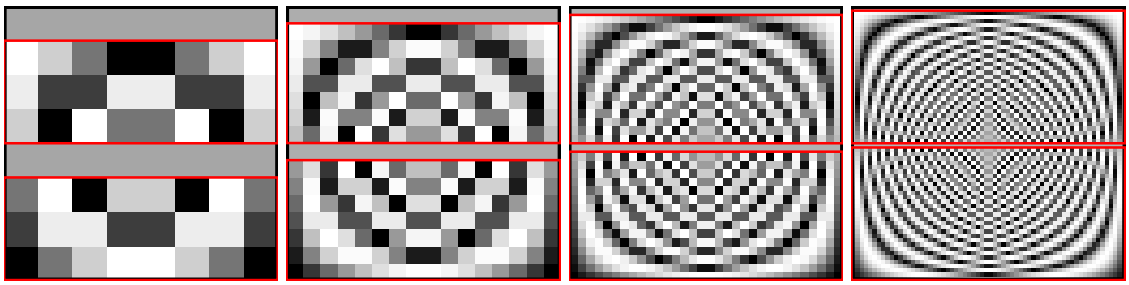
Figure 5.3: *Image representation for the absolute value of the real and complex parts of the DFT matrix for* $N = 8, 16, 32, 64$.

In the images on Figures 5.3, 5.4 and 5.5 it is possible to identify the low complexity rows we can previously fix on the approximation matrix. In particular, for the DFT and DHT matrices, we can also see that some of the remaining rows are repeated. For example, in abs($\Re(\mathbf{F}_{16})$) (Figure 5.3(c)), rows $k = 1, 7, 9, 15$ are the same. Note that if abs($\mathbf{A}$) has repeated rows, then, by definition of the unrestricted version of the method, the optimal solutions for those rows are going to be the same. As a consequence, we only need to approximate the unique rows. In



(a) abs($\mathbf{H}_8$) (b) abs($\mathbf{H}_{16}$) (c) abs($\mathbf{H}_{32}$) (d) abs($\mathbf{H}_{64}$)

Figure 5.4: *Image representation for the absolute value of the DHT transform matrix for* $N = 8, 16, 32, 64$.



(a) abs($\mathbf{C}_8$) (b) abs($\mathbf{C}_{16}$) (c) abs($\mathbf{C}_{32}$) (d) abs($\mathbf{C}_{64}$)

Figure 5.5: *Image representation for the absolute value of the DC T transform matrix for* $N = 8, 16, 32, 64$.

summary, if (i) using the unrestricted version; (ii) using the search space reduction procedure proposed; and (iii) fixing the low complexity rows already in the transform matrix; it is only necessary to approximate the rows highlighted in Figures 5.3, 5.4 and 5.5.

5.3 Matrix symmetries

Looking a little bit further into Figures 5.3, 5.4 and 5.5, it is possible to identify, inside the highlighted regions, some symmetry patterns. That means it is possible to approximate only a portion of those rows and obtain the whole matrix by reflexions on the rows, columns, or both, of the approximated partition.

DFT and DHT

Notice that for the absolute value of the DFT and DHT in Figures 5.3 and 5.4, that are not only low complexity rows but also low complexity columns, $i = 0, N/4, N/2, 3N/4$. Those columns may also be fixed previously on the approximation matrix. Figures 5.6 and 5.7 show

which portion of the highlighted rows in Figures 5.3 and 5.4 we need to approximate. That is, which portion of the matrix we need to approximate in order to obtain the whole matrix apart from the low complexity rows and columns already existent in the original matrix.

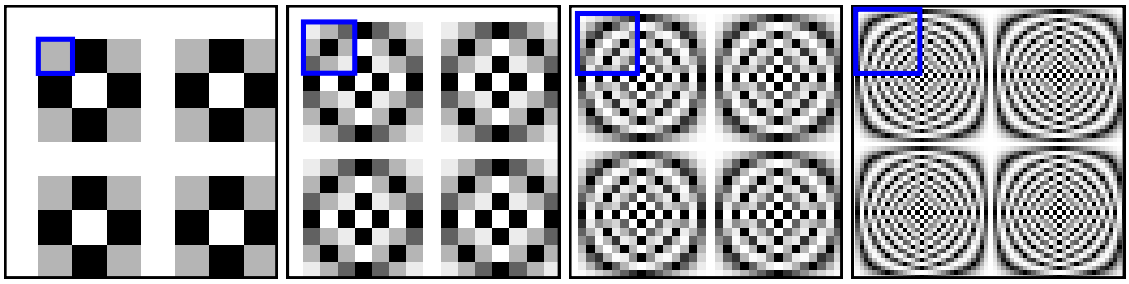


Figure 5.6: *Portion of the DFT matrix to be approximated for* $N = 8, 16, 32, 64$.

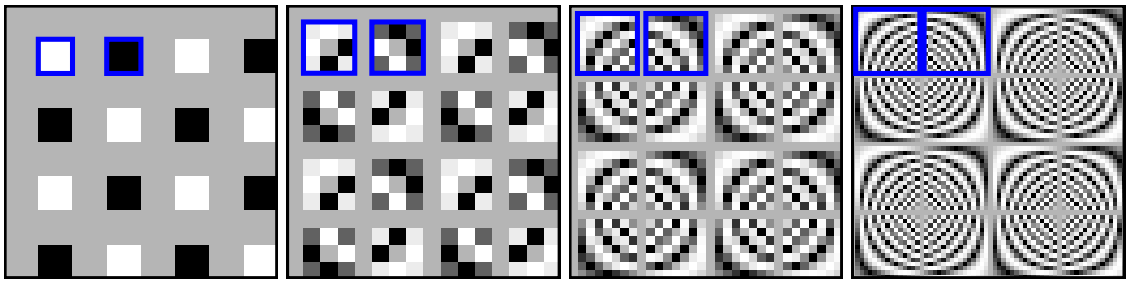


Figure 5.7: *Portion of the DHT matrix to be approximated for* $N = 8, 16, 32, 64$.

DCT

For the DCT there is no low complexity columns that can be previously fixed. However, there symmetries in the rows that can be explored, as show in Figure 5.8.

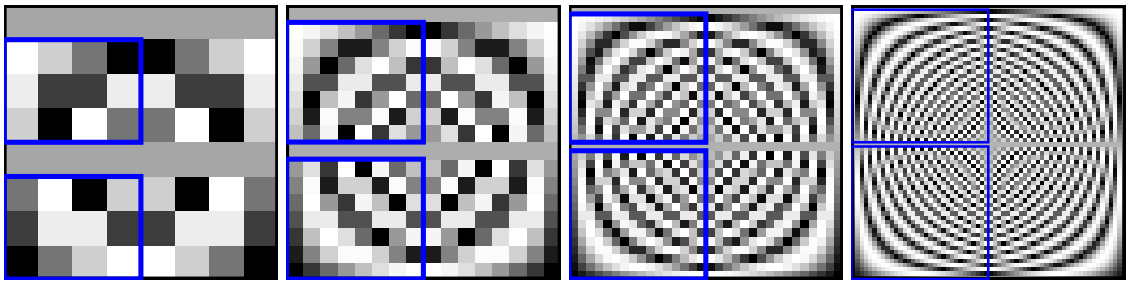


Figure 5.8: *Portion of the DCT matrix to be approximated for* $N = 8, 16, 32, 64$.

Table 5.5 summarizes the information about the rows and columns to be approximated for both versions of the method considering all the possible modifications presented above to reduce the complexity of the approximation process. Table 5.6 also gives the same information

of Table 5.5 for the case when we are only considering previously fixing the low complexity rows. Table 5.7 summarizes which of the modifications presented above may be used for each

Table 5.5: *Summary of the rows e columns to be approximated when using the unrestricted version of the proposed method considering all the possible modifications to reduce the approximation procedure*

Transform	Rows to be approximated	Columns to be approximated	Number of rows to be approximated	Number of columns to be approximated
DFT	1 to $N/4-1$	1 to $N/4-1$	$N/4-1$	$N/4-1$
DHT	1 to $N/4-1$	1 to $N/4-1$ and $N/4+1$ to $N/2-1$	$N/4-1$	$N/2-2$
DCT	1 to $N/2-1$ and $N/2+1$ to $N-1$	0 to $N/2-1$	$N-2$	$N/2$

Table 5.6: *Summary of the rows e columns to be approximated when using the restricted version of the proposed method and previously fixing the low complexity rows of the original matrix*

Transform	Fixed Rows	Columns to be approximated	Number of rows to be approximated	Number of columns to be approximated
DFT	$0, N/4, N/2, 3N/4$	All columns	$N-4$	N
DHT	$0, N/4, N/2, 3N/4$	All columns	$N-4$	N
DCT	$0, N/2$	All columns	$N-2$	N

method.

It is noteworthy that all the modifications can actually be used in association with the restricted version of the proposed method. But, in our case (the matrices we are interested are orthogonal), only previously fixing the low complexity rows guarantees that the output matrix is orthogonal (which is the whole point of this version of the method).

5.4 Approximation schemes

Based on the discussion above, we can define the approximation schemes that can be used to approximate the DCT, DHT and DFT. The DHT and DCT can be approximated using schemes I and II, displayed in Figure 5.9. For the DFT, schemes III and IV in Figure 5.10, which consider the complex adaptation, can be used.

Table 5.7: *Comparison of Methods I and II in terms of the complexity reduction procedures they admit*

	Method I	Method II
Search space reduction	✓	✗
Fix low complexity rows	✓	✓ (only if the input matrix is orthogonal)
Explore matrix symmetries	✓	✗
Complex adaptation	✓	✗

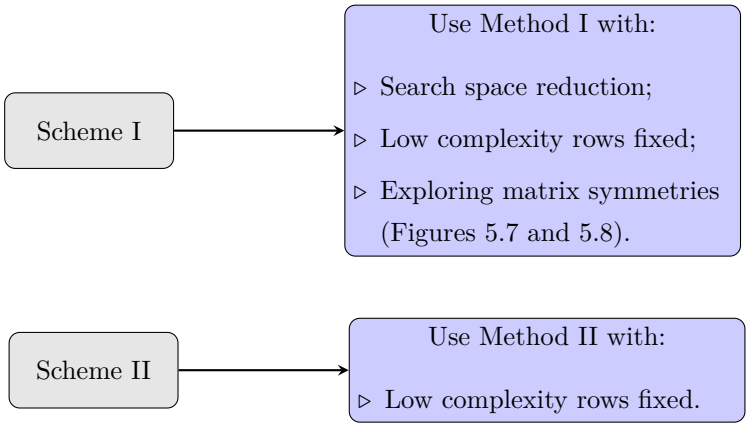


Figure 5.9: *Approximation schemes for the DHT and DCT.*

Notice that the approximation schemes proposed for the DFT consider only the unrestricted version of the method. This is because none of the complex adaptations when used with the restricted version guarantees that the output matrices obtained are orthogonal. Then, the use of the restricted version loses its point. Also, when considering the complex adaptation A, which means we are going to approximate the real and imaginary parts independently, we need to verify if the size of the matrix is $N = 8$. In this case, we have two 8×8 matrices to approximate and, if we check the portion of those matrices that need to be approximated (Figure 5.6(a)), we can see that this region is reduced to a single element, not a vector, which is required by the method. Then, when $N = 8$, we are going to approximate the entire second row, as in Figure 5.3(a)-(b). For the approximation Scheme IV this is not a concern, since when $N = 8$, the 8×8 initial matrix is converted in a 8×16 matrix by the mapping in equation (4.6) and that single element is now a vector with two elements (the real and imaginary parts of original complex element).

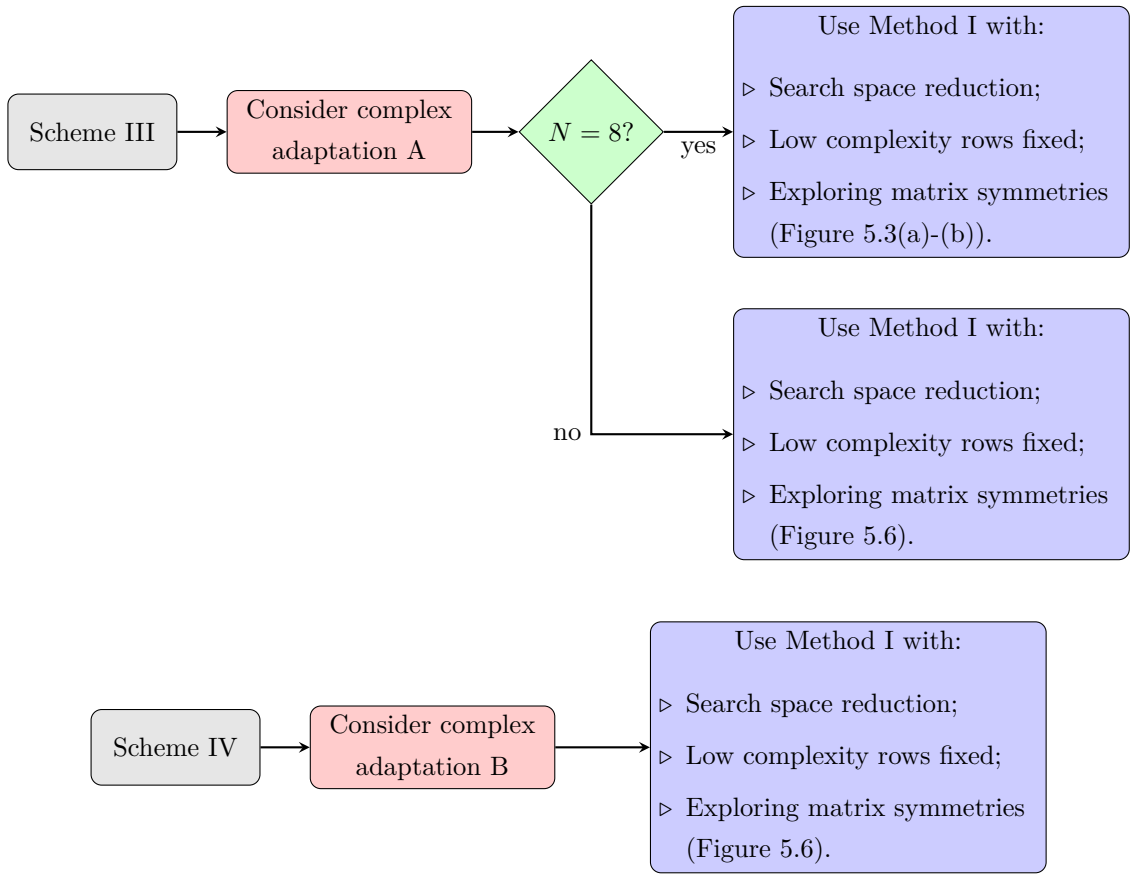


Figure 5.10: *Approximation schemes for the DFT.*

Chapter 6

New angle-based approximations for the DCT

In this chapter, we use the approximation schemes previously defined to obtain new approximations for the DCT of length $N = 8$. In order to evaluate the obtained matrices, some figures of merit are presented.

6.1 Figures of merit

To evaluate the performance of the proposed approximations, we selected traditional figures of merit: (i) total error energy ($\epsilon(\cdot)$) [91]; (ii) mean square error (MSE($\cdot$)) [4], [93]; (iii) coding gain ($C_g(\cdot)$) [4], [94], [95]; and (iv) transform efficiency ($\eta(\cdot)$) [4]. The MSE and total error energy are suitable measures to quantify the difference between the exact DCT and its approximations [4]. The coding gain and transform efficiency are appropriate tools to quantify compression, redundancy removal, and data decorrelation capabilities [4]. Additionally, since for the unrestricted version of the method there is no guarantee of orthogonality, we also considered the orthogonality deviation measure [96].

Hereafter we adopt the following quantities and notation: the interpixel correlation is $\rho = 0.95$ [4], [60], [94], $\widehat{\mathbf{C}}$ is an approximation for the DCT, and $\widehat{\mathbf{R}}_{\mathbf{y}} = \widehat{\mathbf{C}} \cdot \mathbf{R}_{\mathbf{x}} \cdot \widehat{\mathbf{C}}^{\top}$, where $\mathbf{R}_{\mathbf{x}}$ is the covariance matrix of $\mathbf{x}$, whose elements are given by $\rho^{|i-j|}$, $i, j = 1, 2, \ldots, 8$. We detail each of these measures below.

Total Energy Error

The total energy error is a similarity measure given by [91]:

$$\epsilon(\widehat{\mathbf{C}}) = \pi \cdot \|\mathbf{C} - \widehat{\mathbf{C}}\|_{\mathrm{F}}^2,$$

where $\|\cdot\|_{\mathrm{F}}$ represents the Frobenius norm [97].

Mean Square Error

The MSE of a given approximation $\widehat{\mathbf{C}}$ is furnished by [4], [93]:

$$\mathrm{MSE}(\widehat{\mathbf{C}}) = \frac{1}{8} \cdot \mathrm{tr}\left((\mathbf{C} - \widehat{\mathbf{C}}) \cdot \mathbf{R}_{\mathbf{x}} \cdot (\mathbf{C} - \widehat{\mathbf{C}})^{\top}\right).$$

where $\mathrm{tr}(\cdot)$ represents the trace operator [4]. The total energy error and the mean square error are appropriate measures for capturing the approximation error in a Euclidean distance sense.

Coding Gain

The coding gain quantifies the energy compaction capability and is given by [4]:

$$C_g(\widehat{\mathbf{C}}) = 10 \cdot \log_{10}\left\{\frac{\frac{1}{8}\sum_{i=1}^{8} r_{i,i}^2}{\left(\prod_{i=1}^{8} r_{i,i}^2 \cdot \|\widehat{\mathbf{c}}_i\|^2\right)^{1/8}}\right\},$$

where $r_{i,i}$ is the ith element of the diagonal of $\widehat{\mathbf{R}}_{\mathbf{y}}$ [4] and $\widehat{\mathbf{c}}_i$ is the ith row of $\widehat{\mathbf{C}}$.

However, as pointed in [95], the previous definition is suitable for orthogonal transforms only. For non-orthogonal transforms, such as SDCT [78] and MRDCT [87], we adopt the unified coding gain [95]. For $i = 1, 2, \ldots, 8$, let $\widehat{\mathbf{c}}_i$ and $\widehat{\mathbf{g}}_i$ be ith row of $\widehat{\mathbf{C}}$ and $\widehat{\mathbf{C}}^{-1}$, respectively. Then, the unified coding gain is given by:

$$C_g^*(\widehat{\mathbf{C}}) = 10 \cdot \log_{10}\left\{\prod_{i=1}^{8} \frac{1}{\sqrt[8]{A_i \cdot B_i}}\right\},$$

where $A_i = \mathrm{su}\left[(\widehat{\mathbf{c}}_i^{\top} \cdot \widehat{\mathbf{c}}_i) \odot \mathbf{R}_x\right]$, $\mathrm{su}(\cdot)$ returns the sum of all elements of the input matrix, the operator $\odot$ represents the element-wise product, and $B_i = \|\widehat{\mathbf{g}}_i\|^2$.

Transform Efficiency

The transform efficiency is an alternative measure to the coding gain, being expressed according to [4]:

$$\eta(\widehat{\mathbf{C}}) = \frac{\sum_{i=1}^{8} |r_{i,i}|}{\sum_{i=1}^{8}\sum_{j=1}^{8} |r_{i,j}|} \cdot 100,$$

where $r_{i,j}$ is the (i,j)th entry of $\widehat{\mathbf{R}}_{\mathbf{y}}$, $i, j = 1, 2, \ldots, 8$ [4].

Orthogonality deviation

The orthogonality deviation [96] is a measure to quantify how close a matrix is from a diagonal matrix. It is given by:

$$\delta(\mathbf{T}) = 1 - \frac{\|\operatorname{diag}(\mathbf{T})\|_{\mathrm{F}}^2}{\|\mathbf{T}\|_{\mathrm{F}}^2}.$$

6.2 Important definitions

A large number of new approximations were obtained considering the approximation schemes introduced in the previous chapter. In order to optimize the presentation of those approximate matrices, we are going to need the next two definitions:

Definition 6.1 – Equivalence

We say that two matrices are equivalent to each other when they present the same results for a set of evaluation metrics considered. □

Definition 6.2 – Class of equivalence

A set of matrices equivalent to each other form a class of equivalence. □

Although for some cases the number of approximate matrices obtained was large, we were able to identify a reduced number of classes of equivalence. Then, instead of presenting all the matrices obtained, we are going to present only one representative of each class of equivalence. The metrics that define the equivalence relationship between two matrices, the metric to select the representative matrix of each class, and the results obtained are discussed in the next section.

6.3 New approximations

The new approximations were obtained running approximations schemes I and II, which are the appropriate ones for the DCT, as explained in Chapter 5. The low-complexity sets considered to generate the search spaces are displayed in Table 6.1.

From this point on, the new approximation matrices proposed in this work are going to be referred to as $\mathbf{T}_{\mathrm{S}y\mathrm{C}z}$, which means $\mathbf{T}$ is the representative approximation of equivalence class Cz obtained using approximation scheme y. For example, $\mathbf{T}_{\mathrm{SIC1}}$ is the representative approximation of equivalence class C1 obtained using approximation scheme I.

Table 6.1: *Low-complexity sets considered*

Set	Set elements
$\mathcal{P}_1$	$\{-1, 0, 1\}$
$\mathcal{P}_2$	$\{-1, -\frac{1}{2}, 0, -\frac{1}{2}, 1\}$
$\mathcal{P}_3$	$\{-2, -1, 0, 1, 2\}$
$\mathcal{P}_4$	$\{-3, -1, 0, 1, 3\}$
$\mathcal{P}_5$	$\{-1, -\frac{1}{2}, -\frac{1}{4}, 0, -\frac{1}{4}, -\frac{1}{2}, 1\}$
$\mathcal{P}_6$	$\{-2, -1, -\frac{1}{2}, 0, -\frac{1}{2}, 1, 2\}$
$\mathcal{P}_7$	$\{-3, -1, -\frac{1}{2}, 0, -\frac{1}{2}, 1, 3\}$
$\mathcal{P}_8$	$\{-2, -1, -\frac{1}{2}, -\frac{1}{4}, 0, -\frac{1}{4}, -\frac{1}{2}, 1, 2\}$
$\mathcal{P}_9$	$\{-3, -2, -1, -\frac{1}{2}, 0, -\frac{1}{2}, 1, 2, 3\}$

The following evaluation metrics were considered to define the the classes of equivalence:

- ▷ Total error energy;
- ▷ Mean square error;
- ▷ Coding gain; and
- ▷ Transform efficiency.

For approximations obtained using the approximation scheme I, which considers the version of the method not constrained to orthogonality, the approximate matrix chosen to be the representative of each class was the one with the minimum orthogonality deviation. For the ones obtained using the approximation scheme II, the representative matrix of each class was the one with the minimum arithmetic complexity. Table 6.2 summarizes the results obtained. The actual matrices obtained are presented in Appendix A of this work.

Table 6.2: *Total matrices and classes of equivalence obtained for the* 8*-point DCT*

Approximation scheme	Number of matrices obtained	Number of classes of equivalence
Scheme I	151	6
Scheme II	15	10

Among the matrices obtained, three had already been introduced in literature. We verified that $\mathbf{T}_{\text{SIC1}} = \mathbf{T}_{\text{SIIC1}} = \mathbf{T}_{\text{RDCT}}$ and $\mathbf{T}_{\text{SIIC2}} = \mathbf{T}_{CBT--4}$. Thus, for further analysis we focus on the 13 new approximations obtained. Table 6.3 displays an overview of the representative matrices of each class of equivalence.

Table 6.3: *Overview of the new approximations obtained from the angle based method*

Approximation scheme	Class of equivalence	Representative matrix	Representative approximation	$\delta(\mathbf{T})$	Additions	Bit-shiftings
Scheme I	C2	$\mathbf{T}_{\text{SIC2}}$	$\widehat{\mathbf{C}}_{\text{SIC2}}$	0.0300	48	16
Scheme I	C3	$\mathbf{T}_{\text{SIC3}}$	$\widehat{\mathbf{C}}_{\text{SIC3}}$	0.0130	80	24
Scheme I	C4	$\mathbf{T}_{\text{SIC4}}$	$\widehat{\mathbf{C}}_{\text{SIC4}}$	0.0005	56	32
Scheme I	C5	$\mathbf{T}_{\text{SIC5}}$	$\widehat{\mathbf{C}}_{\text{SIC5}}$	0.0086	52	16
Scheme I	C6	$\mathbf{T}_{\text{SIC6}}$	$\widehat{\mathbf{C}}_{\text{SIC6}}$	0.0017	76	40
Scheme II	C3	$\mathbf{T}_{\text{SIIC3}}$	$\widehat{\mathbf{C}}_{\text{SIIC3}}$	0	48	24
Scheme II	C4	$\mathbf{T}_{\text{SIIC4}}$	$\widehat{\mathbf{C}}_{\text{SIIC4}}$	0	48	16
Scheme II	C5	$\mathbf{T}_{\text{SIIC5}}$	$\widehat{\mathbf{C}}_{\text{SIIC5}}$	0	80	24
Scheme II	C6	$\mathbf{T}_{\text{SIIC6}}$	$\widehat{\mathbf{C}}_{\text{SIIC6}}$	0	80	24
Scheme II	C7	$\mathbf{T}_{\text{SIIC7}}$	$\widehat{\mathbf{C}}_{\text{SIIC7}}$	0	56	32
Scheme II	C8	$\mathbf{T}_{\text{SIIC8}}$	$\widehat{\mathbf{C}}_{\text{SIIC8}}$	0	56	32
Scheme II	C9	$\mathbf{T}_{\text{SIIC9}}$	$\widehat{\mathbf{C}}_{\text{SIIC9}}$	0	72	40
Scheme II	C10	$\mathbf{T}_{\text{SIIC10}}$	$\widehat{\mathbf{C}}_{\text{SIIC10}}$	0	72	40

In Table 6.4, the measurements obtained for the approximations in literature along with the results for the new approximations for the figures of merit considered to define the classes of equivalence are shown. The DCT and integer DCT (IDCT) [98] results were included as reference. The top five results for each measure are displayed in bold and were *all obtained from new approximations proposed in this work.*

Table 6.4: *Performance measures for the DCT approximations in literature and the new approximations proposed*

Approximation	$\epsilon(\widehat{\mathbf{C}})$	MSE($\widehat{\mathbf{C}}$)	$C_g^*(\widehat{\mathbf{C}})$	$\eta(\widehat{\mathbf{C}})$
DCT [99]	0	0	8.8259	93.9912
IDCT (HEVC) [98]	0.0020	8.66×10^{-6}	8.8248	93.8236
$\widehat{\mathbf{C}}_{\text{WHT}}$ [77]	47.6126	0.2241	7.9461	85.3138
$\widehat{\mathbf{C}}_{\text{ORTEGA}}$ [79]	0.8695	0.0061	8.3902	88.7023
$\widehat{\mathbf{C}}_{\text{SDCT}}$ [78]	3.3158	0.0207	6.0261	82.6190
$\widehat{\mathbf{C}}_{\text{RDCT}}$ [48]	1.7945	0.0098	8.1827	87.4297
$\widehat{\mathbf{C}}_{\text{MRDCT}}$ [87]	8.6592	0.0594	7.3326	80.8969
$\widehat{\mathbf{C}}_{\text{BAS-2008a}}$ [80]	5.9294	0.0238	8.1194	86.8626
$\widehat{\mathbf{C}}_{\text{BAS-2008b}}$ [81]	4.1875	0.0191	6.2684	83.1734
$\widehat{\mathbf{C}}_{\text{BAS-2009}}$ [82]	6.8543	0.0275	7.9126	85.3799
$\widehat{\mathbf{C}}_{\text{BAS-2010}}$ [83]	4.0935	0.0210	8.3251	88.2182
$\widehat{\mathbf{C}}_{\text{BAS-2011}}$ [84]	26.8462	0.0710	7.9118	85.6419
$\widehat{\mathbf{C}}_{\text{BAS-2013}}$ [85]	35.0639	0.1023	7.9461	85.3138
$\widehat{\mathbf{C}}_{\text{CBT–1}}$ [49]	8.5953	0.0375	8.1361	86.8051
$\widehat{\mathbf{C}}_{\text{CBT–2}}$ [49]	1.7945	0.0100	8.1361	86.8051
$\widehat{\mathbf{C}}_{\text{CBT–3}}$ [49]	1.7945	0.0098	8.1834	87.1567
$\widehat{\mathbf{C}}_{\text{CBT–4}}$ [49]	1.7945	0.0100	8.1369	86.5359
$\widehat{\mathbf{C}}_{\text{CBT–5}}$ [49]	0.8695	0.0062	8.3437	88.0594
$\widehat{\mathbf{C}}_{\text{CBT–6}}$ [49]	3.3158	0.0208	6.0462	83.0814
$\widehat{\mathbf{C}}_{\text{CBT–7}}$ [49]	2.1473	0.0665	6.4434	63.7855
$\widehat{\mathbf{C}}_{\text{SIC2}}$	**0.4022**	**0.0028**	8.4721	90.1603
$\widehat{\mathbf{C}}_{\text{SIC3}}$	**0.5765**	0.0040	**8.4412**	90.5152
$\widehat{\mathbf{C}}_{\text{SIC4}}$	**0.1691**	**0.0011**	**8.7184**	**91.9696**
$\widehat{\mathbf{C}}_{\text{SIC5}}$	**0.4022**	**0.0028**	8.4520	**90.6123**
$\widehat{\mathbf{C}}_{\text{SIC6}}$	**0.1272**	**0.0008**	**8.7654**	**92.8767**
$\widehat{\mathbf{C}}_{\text{SIIC3}}$	1.2194	0.0046	8.6337	90.4615
$\widehat{\mathbf{C}}_{\text{SIIC4}}$	1.2194	0.0127	8.1024	87.2275
$\widehat{\mathbf{C}}_{\text{SIIC5}}$	2.4482	0.0084	8.4301	90.5362
$\widehat{\mathbf{C}}_{\text{SIIC6}}$	2.4482	0.0265	7.8837	87.7395
$\widehat{\mathbf{C}}_{\text{SIIC7}}$	1.5452	0.0043	**8.6693**	**91.4370**
$\widehat{\mathbf{C}}_{\text{SIIC8}}$	1.5452	0.0176	8.0161	88.4340
$\widehat{\mathbf{C}}_{\text{SIIC9}}$	1.0145	**0.0029**	**8.7393**	**92.3530**
$\widehat{\mathbf{C}}_{\text{SIIC10}}$	1.0145	0.0114	8.1454	88.5210

Chapter 7

Approximations performance on image processing

7.1 Image compression experiments

To evaluate the efficiency of the proposed transformation matrices, we performed a JPEG-like image compression experiment as described in [45], [48], [49]. Input images were divided into sub-blocks of size 8×8 pixels and submitted to a bi-dimensional (2-D) transformation, such as the DCT or one of its approximations. Let $\mathbf{A}$ be a sub-block of size 8×8. The 2-D approximate transform of $\mathbf{A}$ is an 8×8 sub-block $\mathbf{B}$ obtained as follows [49], [91]:

$$\mathbf{B} = \widehat{\mathbf{C}} \cdot \mathbf{A} \cdot \widehat{\mathbf{C}}^{\top}.$$

Considering the zig-zag scan pattern as detailed in [100] and shown in Figure 7.1, the initial r, $r = 1, 2, 3, \ldots, 64$, elements of $\mathbf{B}$ were retained; whereas the remaining $(64-r)$ elements were discarded. The previous operation results in a matrix $\mathbf{B}'$ populated with zeros which is suitable for entropy encoding [39]. Each processed sub-block was submitted to the corresponding 2-D inverse transformation and the image was reconstructed. The 2-D inverse transform is given by:

$$\mathbf{A} = \begin{cases} \widehat{\mathbf{C}}^{\top} \cdot \mathbf{B} \cdot \widehat{\mathbf{C}}, & \text{if } \mathbf{T} \text{ for orthogonal,} \\ \widehat{\mathbf{C}}^{-1} \cdot \mathbf{B} \cdot \left(\widehat{\mathbf{C}}^{-1}\right)^{\top}, & \text{otherwise.} \end{cases}$$

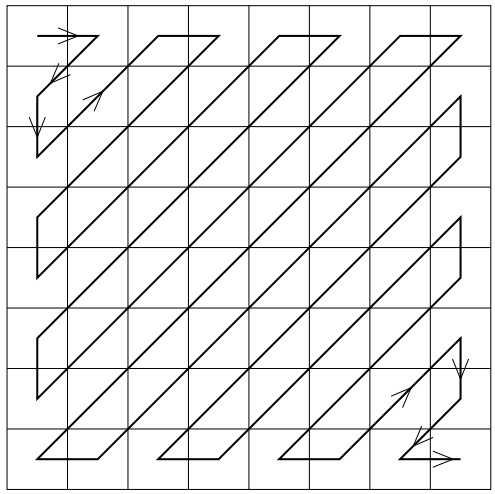


Figure 7.1: *Zig-zag pattern*

We considered 44 8-bit standardized images obtained from the USC-SIPI image bank [101] and submitted them to the above described procedure. The reconstructed images were compared with the original images and evaluated quantitatively according to popular figures of merit: the mean square error (MSE) [4], the peak signal-to-noise ratio (PSNR) [102] and the structural similarity index (SSIM) [103]. We consider the MSE and PSNR measures because of its good properties and historical usage. However, as discussed in [93], the MSE and PSNR are not the best measures when it comes to predict human perception of image fidelity and quality, for which SSIM has been shown to be a better measure [93], [103].

Additionally, for better visualization of the results, we considered the relative difference for each measures. The relative difference is given by:

$$\mathrm{RD}_\mu = \frac{\mu(\mathbf{C}) - \mu(\widehat{\mathbf{C}})}{\mu(\mathbf{C})} = 1 - \frac{\mu(\widehat{\mathbf{C}})}{\mu(\mathbf{C})}$$

where $\mu(\mathbf{C})$ and $\mu(\widehat{\mathbf{C}})$ indicate the exact DCT measure and the measure of an approximation, respectively, and $\mu \in \{\mathrm{MSE}, \mathrm{PSNR}, \mathrm{SSIM}\}$.

For the MSE, we aim at the lowest possible results. That is, we look for approximations whose MSE is the closest possible to the DCT MSE or even smaller. In general, for the approximations in literature, $\mathrm{MSE}(\widehat{\mathbf{C}}) > \mathrm{MSE}(\mathbf{C})$ or, equivalently, $\mathrm{MSE}(\widehat{\mathbf{C}})/\,\mathrm{MSE}(\mathbf{C}) > 1$. In this sense, we search for approximations such that,

$$\frac{\mathrm{MSE}(\widehat{\mathbf{C}})}{\mathrm{MSE}(\mathbf{C})} \to 1^+,$$

where $\to 1^+$ represents right convergence. In other words, $\mathrm{RD}_{\mathrm{MSE}} \to 0$, or even $\widehat{\mathbf{C}}$ e $\mathbf{C}$ are

equivalents. Ideally, we want approximations such that

$$\frac{\mathrm{MSE}(\widehat{\mathbf{C}})}{\mathrm{MSE}(\mathbf{C})} < 1.$$

That is, $\mathrm{RD}_{\mathrm{MSE}} > 0$, which means $\widehat{\mathbf{C}}$ presents better results than $\mathbf{C}$ in terms of MSE.

In other hand, for the PSNR and SSIM, we aim at the largest possible values. So in this case, we want approximations such that the PSNR or SSIM are as large as DCT PSNR or SSIM or even larger. Let $\mu \in \{\mathrm{PSNR}, \mathrm{SSIM}\}$. Usually, approximations in literature satisfy $\mu(\widehat{\mathbf{C}}) < \mu(\mathbf{C})$, i.e., $\mu(\widehat{\mathbf{C}})/\mu(\mathbf{C}) < 1$. In this sense, we want approximation such that

$$\frac{\mu(\widehat{\mathbf{C}})}{\mu(\mathbf{C})} \to 1^-,$$

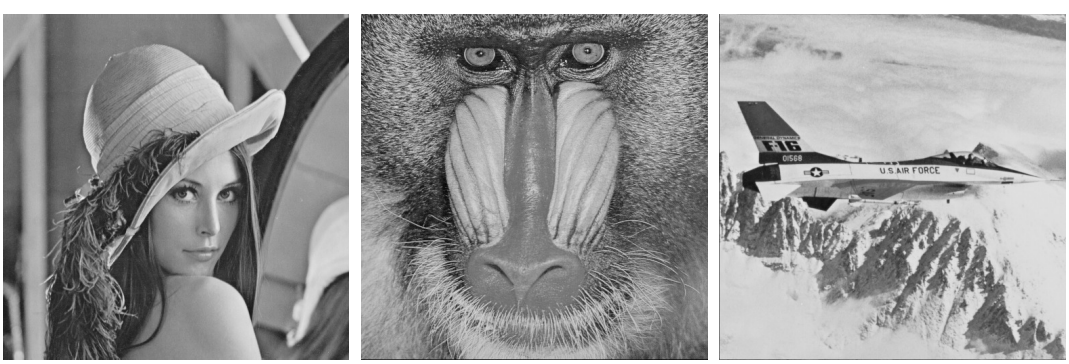
where $\to 1^-$ represents left convergence. That is the same as saying that $\mathrm{RD}_\mu \to 0$. Ideally, we look for approximations such that

$$\frac{\mu(\widehat{\mathbf{C}})}{\mu(\mathbf{C})} > 1,$$

which indicates $\mathrm{RD}_\mu < 0$. That means that $\widehat{\mathbf{C}}$ presents better results than $\mathbf{C}$ in terms of PSNR or SSIM.

7.2 Results

First, we selected three images from the USC-SIPI image bank [101] and performed the procedure describe above. Figure 7.2 displays the selected images.



(a) Lena (b) Baboon (c) Plane

Figure 7.2: *Sample images*

The MSE, PSNR and SSIM of the reconstructed images obtained from using each approximation presented in this work are shown in Table 7.1.

For the experiments, we considered $r = 10$. In Table 7.1, the 5 best results for each measure and sample image are highlighted. All the approximations among the 5 best for all three images besides the DCT ($\widehat{\mathbf{C}}_{\text{SIC4}}$, $\widehat{\mathbf{C}}_{\text{SIC6}}$, $\widehat{\mathbf{C}}_{\text{SIIC7}}$, $\widehat{\mathbf{C}}_{\text{SIIC9}}$) were introduced in this work.

Table 7.1: *MSE, PSNR and SSIM of each sample image compressed and reconstructed considering the approximations 8-point DCT and* $r = 10$

Image	Lena			Baboon			Plane		
Transform	MSE	PSNR	SSIM	MSE	PSNR	SSIM	MSE	PSNR	SSIM
DCT	**40.2377**	**32.0845**	**0.9763**	**338.3289**	**22.8374**	0.9064	**57.3786**	**30.5433**	**0.9832**
$\widehat{\mathbf{C}}_{\text{WHT}}$	62.0565	30.2029	0.9737	377.2737	22.3642	**0.9138**	103.5323	27.9800	0.9792
$\widehat{\mathbf{C}}_{\text{SDCT}}$	110.2814	27.7058	0.9225	455.8198	21.5429	0.8304	158.1937	26.1389	0.9363
$\widehat{\mathbf{C}}_{\text{ORTEGA}}$	52.2906	30.9466	0.9722	350.1966	22.6877	0.9008	78.0402	29.2076	0.9789
$\widehat{\mathbf{C}}_{\text{RDCT}}$	58.6558	30.4477	0.9658	363.8937	22.5211	0.8723	85.8379	28.7940	0.9727
$\widehat{\mathbf{C}}_{\text{MRDCT}}$	129.0950	27.0217	0.8935	449.8371	21.6003	0.7437	179.5797	25.5882	0.9018
$\widehat{\mathbf{C}}_{\text{BAS–2008a}}$	59.8314	30.3615	0.9633	376.4999	22.3732	0.8845	93.3886	28.4279	0.9710
$\widehat{\mathbf{C}}_{\text{BAS–2008b}}$	53.1955	30.8721	0.9725	362.1343	22.5421	0.8931	84.4295	28.8659	0.9787
$\widehat{\mathbf{C}}_{\text{BAS–2009}}$	66.2534	29.9187	0.9638	389.3617	22.2273	0.8876	107.8282	27.8035	0.9714
$\widehat{\mathbf{C}}_{\text{BAS–2010}}$	49.9871	31.1422	0.9728	358.4581	22.5864	0.9088	78.4957	29.1823	0.9787
$\widehat{\mathbf{C}}_{\text{BAS–2011}}$	65.7003	29.9551	0.9567	389.5474	22.2252	0.8561	100.7354	28.0990	0.9647
$\widehat{\mathbf{C}}_{\text{BAS–2013}}$	62.0565	30.2029	0.9737	377.2737	22.3642	**0.9138**	103.5323	27.9800	0.9792
$\widehat{\mathbf{C}}_{\text{CBT–1}}$	93.8702	28.4055	0.9418	437.7631	21.7184	0.8257	153.5626	26.2680	0.9424
$\widehat{\mathbf{C}}_{\text{CBT–2}}$	61.1506	30.2668	0.9641	372.2386	22.4226	0.8676	86.9480	28.7382	0.9711
$\widehat{\mathbf{C}}_{\text{CBT–3}}$	59.1153	30.4138	0.9727	363.6265	22.5242	0.9038	92.9502	28.4483	0.9792
$\widehat{\mathbf{C}}_{\text{CBT–4}}$	61.7111	30.2272	0.9711	372.0019	22.4254	0.8996	94.0452	28.3974	0.9778
$\widehat{\mathbf{C}}_{\text{CBT–5}}$	55.0551	30.7228	0.9707	358.9443	22.5805	0.8966	79.3029	29.1379	0.9777
$\widehat{\mathbf{C}}_{\text{CBT–6}}$	113.2215	27.5915	0.9113	433.0143	21.7658	0.7985	166.6957	25.9116	0.9178
$\widehat{\mathbf{C}}_{\text{CBT–7}}$	50.1714	31.1262	0.9744	349.7947	22.6927	0.9023	71.2007	29.6060	0.9810
$\widehat{\mathbf{C}}_{\text{SIC2}}$	51.1731	31.0404	0.9647	350.2942	22.6865	0.8950	70.3013	29.6612	0.9756
$\widehat{\mathbf{C}}_{\text{SIC3}}$	50.5512	31.0935	0.9736	350.4899	22.6840	0.8967	69.7947	29.6926	0.9811
$\widehat{\mathbf{C}}_{\text{SIC4}}$	**42.7374**	**31.8227**	0.9754	**340.5167**	**22.8094**	0.9063	**60.9795**	**30.2790**	0.9825
$\widehat{\mathbf{C}}_{\text{SIC5}}$	51.1138	31.0454	0.9647	350.6485	22.6821	0.8942	**69.5870**	**29.7055**	0.9756
$\widehat{\mathbf{C}}_{\text{SIC6}}$	**40.9147**	**32.0120**	**0.9760**	**338.3480**	**22.8372**	0.9071	**58.9464**	**30.4262**	**0.9829**
$\widehat{\mathbf{C}}_{\text{SIIC3}}$	46.8882	31.4202	0.9745	349.1215	22.7010	0.9097	72.4104	29.5328	0.9817
$\widehat{\mathbf{C}}_{\text{SIIC4}}$	68.2633	29.7889	0.9601	370.5620	22.4422	0.8767	97.3407	28.2479	0.9663
$\widehat{\mathbf{C}}_{\text{SIIC5}}$	52.3678	30.9402	**0.9765**	360.4583	22.5623	**0.9118**	82.1987	28.9822	**0.9834**
$\widehat{\mathbf{C}}_{\text{SIIC6}}$	106.0005	27.8777	0.9166	412.5139	21.9764	0.8030	146.8858	26.4610	0.9243
$\widehat{\mathbf{C}}_{\text{SIIC7}}$	**45.0054**	**31.5982**	**0.9774**	**348.9218**	**22.703**5	**0.9165**	71.7508	29.5725	**0.9840**
$\widehat{\mathbf{C}}_{\text{SIIC8}}$	85.7326	28.7993	0.9421	388.7694	22.2339	0.8477	120.5415	27.3194	0.9491
$\widehat{\mathbf{C}}_{\text{SIIC9}}$	**43.1098**	**31.7850**	**0.9772**	**343.9778**	**22.7655**	**0.9141**	**67.6742**	**29.8266**	**0.9839**
$\widehat{\mathbf{C}}_{\text{SIIC10}}$	69.7013	29.6984	0.9584	370.1255	22.4473	0.8748	99.3286	28.1601	0.9652

Among the approximations in literature, only $\widehat{\mathbf{C}}_{\text{WHT}}$ and $\widehat{\mathbf{C}}_{\text{BAS–2013}}$ showed up in the top 5, although it only happened for the Babbon image and SSIM measure.

In order to have a more general idea about the behavior of those transforms, we cared out the experiments describe before for all the 44 images in the dataset, considering all the values of $r = 1, 2, \ldots, 64$. The MSE, PSNR and SSIM were calculated in each case. The average

curves obtained are displayed in the plots in Figure 7.3. The curves for all the approximations were calculated, but only the ones with better results, i.e., the ones with results closer to the DCT results were kept in the plots in order to provide a clear visualization.

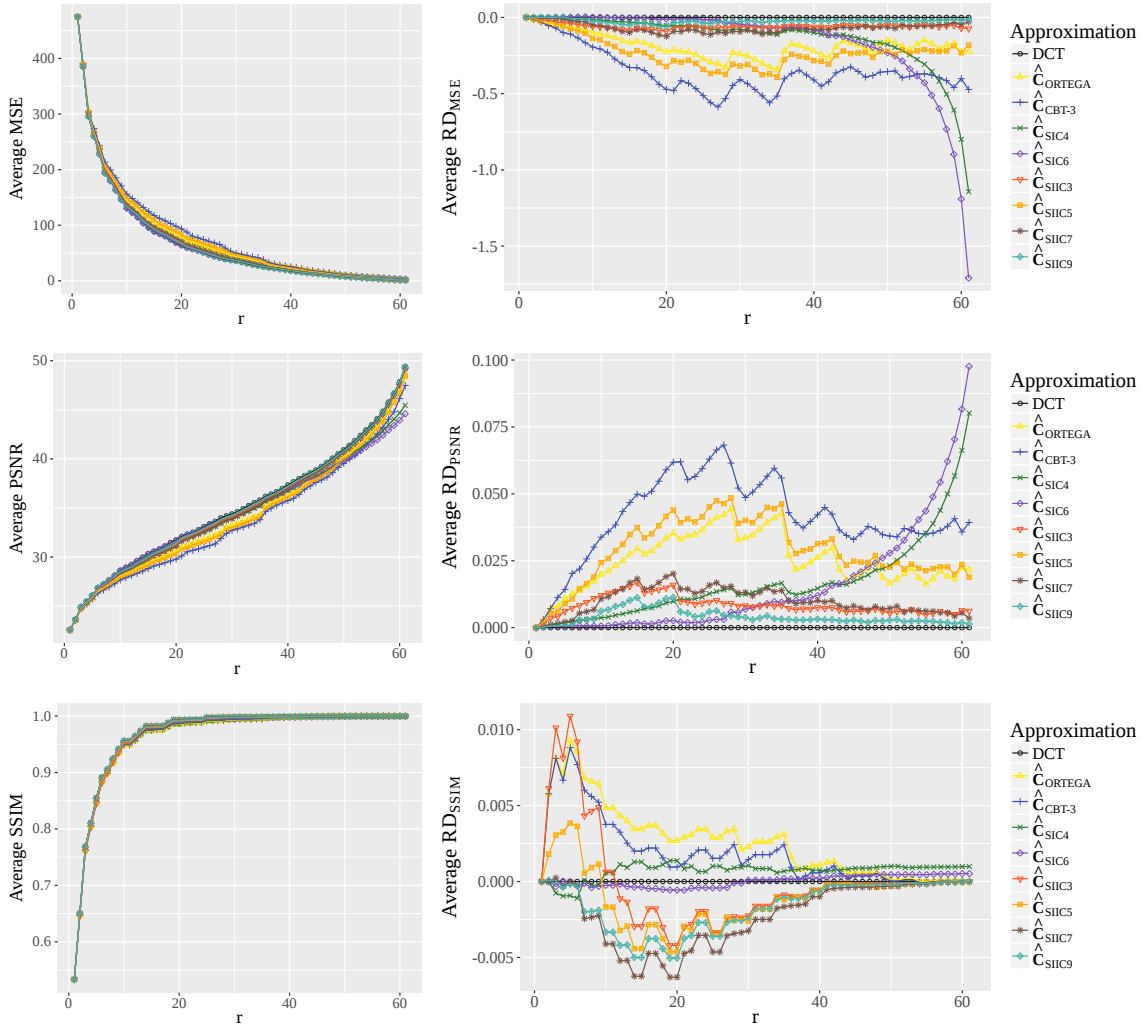


Figure 7.3: *Average curves for the MSE, PSNR and SSIM.*

From the plots in Figure 7.3, we can observe that for the MSE and PSNR: (i) although some approximations present results very close to the DCT, none of them overcomes the DCT; (ii) the proposed approximations $\widehat{\mathbf{C}}_{\text{SIC4}}$ and $\widehat{\mathbf{C}}_{\text{SIC6}}$ show the closest results to the DCT for small values of r. However, as r grows, their curves tend to distance themselves from the DCT; (iii) Approximations $\widehat{\mathbf{C}}_{\text{SIIC3}}$, $\widehat{\mathbf{C}}_{\text{SIIC7}}$ and $\widehat{\mathbf{C}}_{\text{SIIC9}}$ are consistently closer to the DCT than the approximations in literature considered in the plots, $\widehat{\mathbf{C}}_{\text{ORTEGA}}$ and $\widehat{\mathbf{C}}_{\text{CBT–3}}$, and, for larger values of r, they show the closest results to the DCT.

On other hand, for the SSIM, several approximations presented results better than the

DCT. In particular, all the approximations proposed in the work show in the plots of Figure 7.3 have results better than the DCT for, at least, 7 values of r ($\widehat{\mathbf{C}}_{\text{SIC4}}$), and, at most, 59 values of r ($\widehat{\mathbf{C}}_{\text{SIIC7}}$ and $\widehat{\mathbf{C}}_{\text{SIIC9}}$). However, $\widehat{\mathbf{C}}_{\text{ORTEGA}}$ and $\widehat{\mathbf{C}}_{\text{CBT-3}}$, only showed results better than the DCT for one and four values of r, respectively.

Chapter 8

Fast algorithm and hardware

Comparing the computational cost of its direct implementation, performance measures and results in the image compression experiments, we selected one of the new approximations to further analyze. The chosen approximation was $\mathbf{T}_{\text{SIIC3}}$. It is an orthogonal matrix, and it is among the less complex approximations proposed. Also, it overcomes all the approximations in literature in the performance measures and overcomes the DCT in terms of SSIM in the image compression experiments for several values of r.

8.1 Fast Algorithm

The direct implementation of $\mathbf{T}_{\text{SIIC3}}$ requires 48 additions and 24 bit-shifting operations. However, such computational cost can be significantly reduced by means of sparse matrix factorization [6]. In fact, considering butterfly-based structures as commonly found in decimation-in-frequency algorithms, such as [68], [104], [105], we could derive the following factorization for $\mathbf{T}_{\text{SIIC3}}$:

$$\mathbf{T}_{\text{SIIC3}} = \mathbf{D} \cdot \mathbf{A}_4 \cdot \mathbf{A}_3 \cdot \mathbf{A}_2 \cdot \mathbf{A}_1,$$

where:

$$\mathbf{A}_1 = \begin{bmatrix} 1 & & & & & & & 1 \\ & 1 & & & & & 1 & \\ & & 1 & & & 1 & & \\ & & & 1 & 1 & & & \\ & & & 1 & -1 & & & \\ & & 1 & & & -1 & & \\ & 1 & & & & & -1 & \\ 1 & & & & & & & -1 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} 1 & & & 1 & & & & \\ & & 1 & 1 & & & & \\ & -1 & 1 & & & & & \\ -1 & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix},$$

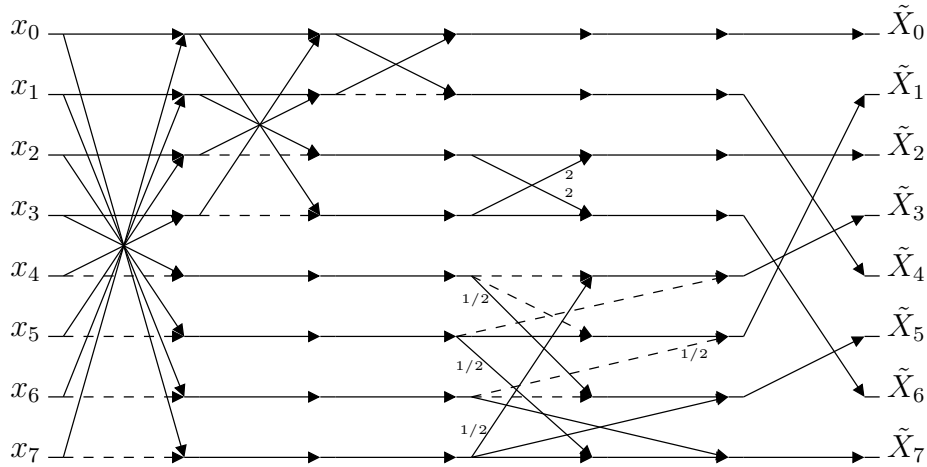


Figure 8.1: *Signal flow graph of the proposed transform, relating the input data x_n, $n = 0, 1, \ldots, 7$, to its correspondent coefficients $\tilde{X}_k$, $k = 0, 1, \ldots, 7$, where $\tilde{\mathbf{X}} = \mathbf{x} \cdot \mathbf{T}_1$. of $\mathbf{T}_1$. Dashed arrows representing multiplication by -1.*

$$\mathbf{A}_3 = \begin{bmatrix} 1 & 1 & & & & & & \\ 1 & -1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix}, \quad \mathbf{A}_4 = \begin{bmatrix} 1 & & & & & & & \\ & & & & \frac{1}{2} & 1 & 1 & \\ & 1 & 2 & & & & & \\ & & & -1 & -1 & & & \frac{1}{2} \\ 1 & & & & & & & \\ & & & \frac{1}{2} & & & -1 & 1 \\ & 2 & 1 & & & & & \\ & & & -1 & 1 & -\frac{1}{2} & & \end{bmatrix},$$

and $\mathbf{D} = \mathrm{diag}(1, 2, 1, 2, 1, 2, 1, 2)$. Figure 8.1 shows the signal flow graph related to the above factorization. The computational cost of this algorithm is only 24 additions and six multiplications by two. The multiplications by two are extremely simple to be performed, requiring only bit-shifting operations [4]. The fast algorithm proposed requires 50% less additions and 75% less bit-shifting operations when compared to the direct implementation. The computational costs of the considered methods are shown in Table 8.1.

In general terms, DCT approximations exhibit a trade-off between computational cost and transform performance [86], i.e., less complex matrices effect poor spectral approximations [4]. Departing from this general behavior, the proposed transformation $\mathbf{T}_{\mathrm{SIIC3}}$ has (i) excelling performance measures and (ii) lower or similar arithmetic cost when compared to competing methods, as shown in Table 8.1.

8.2 FPGA Implementation

The proposed design along with $\mathbf{T}_{\mathrm{ORTEGA}}$ and $\mathbf{T}_{\mathrm{CBT-3}}$ were implemented on an FPGA chip using the Xilinx ML605 board. Considering hardware co-simulation the FPGA realization was tested with 100,000 random 8-point input test vectors. The test vectors were generated from

Table 8.1: *Computational cost comparison*

Method	Multiplications	Additions	Bit-shifts
DCT [59]	11	29	0
IDCT (HEVC) [98]	0	50	30
$\mathbf{T}_{\text{SIIC3}}$ (proposed)	0	24	6
$\widehat{\mathbf{C}}_{\text{ORTEGA}}$ [79]	0	24	2
$\mathbf{T}_{\text{SDCT}}$ [78]	0	24	0
$\mathbf{T}_{\text{RDCT}}$ [48]	0	22	0
$\mathbf{T}_{\text{MRDCT}}$ [87]	0	14	0
$\mathbf{T}_{\text{BAS-2008a}}$ [80]	0	18	2
$\mathbf{T}_{\text{BAS-2008b}}$ [81]	0	21	0
$\mathbf{T}_{\text{BAS-2009}}$ [82]	0	18	0
$\mathbf{T}_{\text{BAS-2011}}$ [84]	0	16	0
$\mathbf{T}_{\text{BAS-2013}}$ [85]	0	24	0
$\mathbf{T}_{\text{CBT–1}}$ [49]	0	22	4
$\mathbf{T}_{\text{CBT–2}}$ [49]	0	22	6
$\mathbf{T}_{\text{CBT–3}}$ [49]	0	24	0
$\mathbf{T}_{\text{CBT–4}}$ [49]	0	24	4
$\mathbf{T}_{\text{CBT–5}}$ [49]	0	24	6
$\mathbf{T}_{\text{CBT–6}}$ [49]	0	18	0
$\mathbf{T}_{\text{CBT–7}}$ [49]	0	28	12

within the MATLAB environment and, using JTAG based hardware co-simulation, routed to the physical FPGA device where each algorithm was realized in the reconfigurable logic fabric. Then the computational results obtained from the FPGA algorithm implementations were routed back to MATLAB memory space. The diagrams for the designs can be seen in Figure 8.2.

The metrics employed to evaluate the FPGA implementations were: configurable logic blocks (CLB), flip-flop (FF) count, and critical path delay (T_{cpd}), in ns. The maximum operating frequency was determined by the critical path delay as $F_{\max} = (T_{\text{cpd}})^{-1}$, in MHz. Values were obtained from the Xilinx FPGA synthesis and place-route tools by accessing the `xflow.results` report file. Using the Xilinx XPower Analyzer, we estimated the static (Q_p in W) and dynamic power (D_p in mW/MHz) consumption. In addition, we calculated area-time (AT) and area-time-square (AT^2) figures of merit, where area is measured as the CLBs and time as the critical path delay. The values of those metrics for each design are shown in Table 8.2.

The design linked to the proposed design approximation $\mathbf{T}_{\text{SIIC3}}$ possesses the smallest T_{cpd} among the considered methods. Such critical path delay allows for operations at a 8.55% and 19.96% higher frequency than the designs associated to $\mathbf{T}_{\text{ORTEGA}}$ and $\mathbf{T}_{\text{CBT–3}}$, respectively.

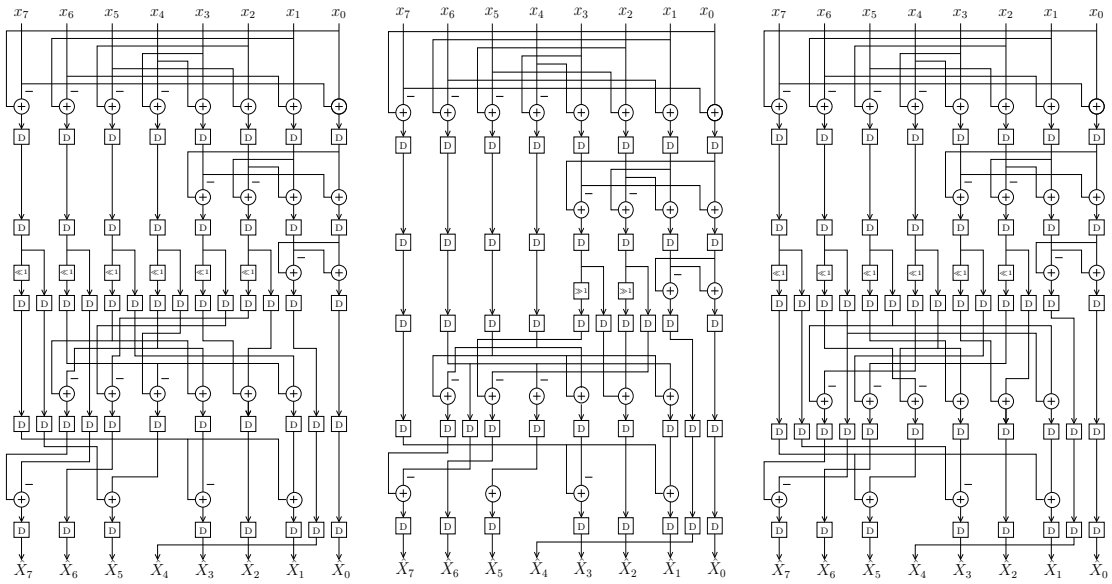


Figure 8.2: *Architectures for (a)* $\mathbf{T}_{SIIC3}$*, (b)* $\mathbf{T}_{ORTEGA}$*, and (c)* $\mathbf{T}_{CBT-3}$*.*

Table 8.2: *Hardware resource consumption and power consumption using Xilinx Virtex-6 XC6VLX240T 1FFG1156 device*

Approximation	CLB	FF	T_{cpd} (ns)	$F_{\max}$ (MHz)	D_p (mW/GHz)	Q_p (W)	AT	AT^2
$\mathbf{T}_{\text{SIIC3}}$ (proposed)	135	408	1.750	571	2.74	3.471	236	413
$\mathbf{T}_{\text{LO}}$ [79]	114	349	1.900	526	2.82	3.468	217	412
$\mathbf{T}_{6}$ [49]	125	389	2.100	476	2.57	3.460	262	551

In terms of area-time and are-time-square measures, the design linked to the approximation $\mathbf{T}_{\text{ORTEGA}}$ presents the best results, followed by the one associated to $\mathbf{T}_{\text{SIIC3}}$.

Chapter 9

Conclusion

In this work, we introduced a greedy algorithm to find low-complexity approximations for a given matrix based on angular distance. The initial version of the method had no constraints. Thus, in order to guarantee the orthogonality of the obtained approximations, we proposed a constrained to orthogonality version of the algorithm. Then, we discussed several ways to reduce the algorithm complexity by exploring the DFT, DHT and DCT structural features and defined approximations schemes. The defined approximations schemes were used to derive new approximations for the 8-point DCT. Thirteen new approximations were obtained. All of them had outstanding results in terms of performance measures and six of them had also great results in the image compression experiments. Based on the results from the evaluation steps, we took $\mathbf{T}_{\text{SIIC3}}$ and proposed a fast algorithm for it that requires only 24 additions and 6 bit-shiftings operations. The FPGA implementation of $\mathbf{T}_{\text{SIIC3}}$ was also implemented and compared to $\mathbf{T}_{\text{ORTEGA}}$ and $\mathbf{T}_{\text{CBT–3}}$. In this case, $\mathbf{T}_{\text{SIIC3}}$ also overcame the approximations in literature, being able to work at a 8.55% and 19.96% higher frequency than the designs associated to $\mathbf{T}_{\text{ORTEGA}}$ and $\mathbf{T}_{\text{CBT–3}}$, respectively.

References

[1] R. Priemer, "Introductory signal processing," 1991.

[2] J. Moura, "What is signal processing? [president's message]," *IEEE Signal Processing Magazine*, vol. 26, no. 6, pp. 6–6, Nov 2009.

[3] I. T. on Signal Processing. [Online]. Available: http://ieeexplore.ieee.org/xpl/aboutJournal.jsp?punumber=78

[4] V. Britanak, P. Yip, and K. R. Rao, *Discrete Cosine and Sine Transforms.* Academic Press, 2007.

[5] R. J. Cintra, "An integer approximation method for discrete sinusoidal transforms," *Journal of Circuits, Systems, and Signal Processing*, vol. 30, no. 6, pp. 1481–1501, December 2011.

[6] R. E. Blahut, *Fast algorithms for signal processing.* Cambridge University Press, 2010.

[7] G. Strang, "Wavelets," *American Scientist*, vol. 82, no. 3, pp. 250–255, 1994.

[8] H. Helms, "Fast fourier transform method of computing difference equations and simulating filters," *IEEE Transactions on Audio and Electroacoustics*, vol. 15, no. 2, pp. 85–90, 1967.

[9] B. S. Reddy and B. N. Chatterji, "An FFT-based technique for translation, rotation, and scale-invariant image registration," *IEEE Tansactions on Image Processing*, vol. 5, no. 8, pp. 1266–1271, 1996.

[10] R. C. Gonzalez and R. E. Woods, *Digital image processing.* Upper Saddle River, NJ: Prentice Hall, 2012.

[11] L. C. Godara, "Application of the fast fourier transform to broadband beamforming," *The Journal of the Acoustical Society of America*, vol. 98, no. 1, pp. 230–240, 1995.

[12] S. R. Seydnejad and S. Akhzari, "Performance evaluation of pre-and post-fft beamforming methods in pilot-assisted simo-ofdm systems," *Telecommunication Systems*, vol. 61, no. 3, pp. 471–487, 2016.

[13] C.-H. Cheng, L. L. Liou, J. B. Tsui, and D. M. Lin, "Chirp signal detection using fft peak frequency difference [correspondence]," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 3, pp. 1449–1453, 2016.

[14] S. Saponara and B. Neri, "Radar sensor signal acquisition and multidimensional fft processing for surveillance applications in transport systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 4, pp. 604–615, 2017.

[15] D. H. Klatt and L. C. Klatt, "Analysis, synthesis, and perception of voice quality variations among female and male talkers," *the Journal of the Acoustical Society of America*, vol. 87, no. 2, pp. 820–857, 1990.

[16] S. M. Ransom, S. S. Eikenberry, and J. Middleditch, "Fourier techniques for very long astrophysical time-series analysis," *The Astronomical Journal*, vol. 124, no. 3, p. 1788, 2002.

[17] K. S. Perera, M. Hahmann, W. Lehner, T. B. Pedersen, and C. Thomsen, "Modeling large time series for efficient approximate query processing," in *International Conference on Database Systems for Advanced Applications*. Springer, 2015, pp. 190–204.

[18] M.-Y. Chen and B.-T. Chen, "Online fuzzy time series analysis based on entropy discretization and a fast fourier transform," *Applied Soft Computing*, vol. 14, pp. 156–166, 2014.

[19] S. M. Kay, *Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory*, ser. Prentie Hall Signal Processing Series, A. V. Oppenheim, Ed. Upper Saddle River, NJ: Prentice Hall, 1993, vol. 1.

[20] I. Bártfai, "An illustration of harmonic regression based on the results of the fast fourier transformation," *Yugoslav journal of operations research*, vol. 12, no. 2, 2016.

[21] F. Fitzke, B. Masters, R. Buckley, and L. Speedwell, "Fourier transform analysis of human corneal endothelial specular photomicrographs," *Experimental eye research*, vol. 65, no. 2, pp. 205–214, 1997.

[22] R. N. Bracewell, "Discrete hartley transform," *JOSA*, vol. 73, no. 12, pp. 1832–1835, 1983.

[23] M. T. H. (auth.), *Multiplicative Complexity, Convolution, and the DFT*, 1st ed., ser. Signal Processing and Digital Filtering. Springer-Verlag New York, 1988.

[24] R. N. Bracewell and R. N. Bracewell, *The Fourier transform and its applications*. McGraw-Hill New York, 1986, vol. 31999.

[25] C.-C. Tseng and S.-L. Lee, "Digital image sharpening using riesz fractional order derivative and discrete hartley transform," in *Circuits and Systems (APCCAS), 2014 IEEE Asia Pacific Conference on*. IEEE, 2014, pp. 483–486.

[26] H. Kasban, "A spiral based image watermarking scheme using karhunen–loeve and discrete hartley transforms," *Multidimensional Systems and Signal Processing*, vol. 28, no. 2, pp. 573–595, 2017.

[27] P. Duhamel and M. Vetterli, "Improved fourier and hartley transform algorithms: Application to cyclic convolution of real data," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 6, pp. 818–824, 1987.

[28] S.-C. Pei and S.-B. Jaw, "Computation of discrete hilbert transform through fast hartley transform," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 9, pp. 1251–1252, 1989.

[29] H. Jleed and M. Bouchard, "Acoustic environment classification using discrete hartley transform features," in *Electrical and Computer Engineering (CCECE), 2017 IEEE 30th Canadian Conference on*. IEEE, 2017, pp. 1–4.

[30] K. Shruthi, B. Shashank, Y. S. Saketh, H. Prasantha, and S. Sandya, "Comparison analysis of a biomedical image for compression using various transform coding techniques," in *Advanced Computing (IACC), 2016 IEEE 6th International Conference on*. IEEE, 2016, pp. 297–303.

[31] G. Heydt, K. Olejniczak, R. Sparks, and E. Viscito, "Application of the hartley transform for the analysis of the propagation of nonsinusoidal waveforms in power systems," *IEEE transactions on power delivery*, vol. 6, no. 4, pp. 1862–1868, 1991.

[32] X. Zhang, H. Wu, and Y. Ma, "A new auto-focus measure based on medium frequency discrete cosine transform filtering and discrete cosine transform," *Applied and Computational Harmonic Analysis*, vol. 40, no. 2, pp. 430–437, 2016.

[33] L. Cao, L. Jin, H. Tao, G. Li, Z. Zhuang, and Y. Zhang, "Multi-focus image fusion based on spatial frequency in discrete cosine transform domain," *IEEE signal processing letters*, vol. 22, no. 2, pp. 220–224, 2015.

[34] R. A. Kozhemiakin, V. V. Lukin, B. Vozel, and K. Chehdi, "Filtering of dual-polarization radar images based on discrete cosine transform," in *Radar Symposium (IRS), 2014 15th International*. IEEE, 2014, pp. 1–4.

[35] S. S. Nassar, N. M. Ayad, H. M. Kelash, H. S. El-Sayed, M. A. El-Bendary, F. E. A. El-Samie, and O. S. Faragallah, "Efficient audio integrity verification algorithm using discrete cosine transform," *International Journal of Speech Technology*, vol. 19, no. 1, pp. 1–8, 2016.

[36] B. Ram, "Digital image watermarking technique using discrete wavelet transform and discrete cosine transform," *International journal of Advancements in Research & technology*, vol. 2, no. 4, pp. 19–27, 2013.

[37] M. Lei, Y. Yang, X. Liu, M. Cheng, and R. Wang, "Audio zero-watermark scheme based on discrete cosine transform-discrete wavelet transform-singular value decomposition," *China Communications*, vol. 13, no. 7, pp. 117–121, 2016.

[38] Z. Fan, J. Jiang, S. Weng, Z. He, and Z. Liu, "Human gait recognition based on discrete cosine transform and linear discriminant analysis," in *Signal Processing, Communications and Computing (ICSPCC), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–6.

[39] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. xviii–xxxiv, February 1992.

[40] D. J. Le Gall, "The MPEG video compression algorithm," *Signal Processing: Image Communication*, vol. 4, no. 2, pp. 129–140, 1992.

[41] International Telecommunication Union, "ITU-T recommendation H.261 version 1: Video codec for audiovisual services at $p \times 64$ kbits," ITU-T, Tech. Rep., 1990.

[42] ——, "ITU-T recommendation H.263 version 1: Video coding for low bit rate communication," ITU-T, Tech. Rep., 1995.

[43] A. Luthra, G. J. Sullivan, and T. Wiegand, "Introduction to the special issue on the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 557–559, Jul. 2003.

[44] M. T. Pourazad, C. Doutre, M. Azimi, and P. Nasiopoulos, "HEVC: The new gold standard for video compression: How does HEVC compare with H.264/AVC?" *IEEE Consumer Electronics Magazine*, vol. 1, no. 3, pp. 36–46, Jul. 2012.

[45] U. Sadhvi Potluri, A. Madanayake, R. J. Cintra, F. M. Bayer, S. Kulasekera, and A. Edirisuriya, "Improved 8-point approximate DCT for image and video compression requiring only 14 additions," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 61, no. 6, pp. 1727–1740, 2014.

[46] V. A. Coutinho, R. J. Cintra, F. M. Bayer, S. Kulasekera, and A. Madanayake, "A multiplierless pruned DCT-like transformation for image and video compression that requires ten additions only," *Journal of Real-Time Image Processing*, pp. 1–9, 2015.

[47] F. M. Bayer, R. J. Cintra, A. Edirisuriya, and A. Madanayake, "A digital hardware fast algorithm and FPGA-based prototype for a novel 16-point approximate DCT for image compression applications," *Measurement Science and Technology*, vol. 23, no. 8, p. 114010, November 2012.

[48] F. M. Bayer and R. J. Cintra, "Image compression via a fast DCT approximation," *IEEE Latin America Transactions*, vol. 8, no. 6, pp. 708–713, Dec. 2010.

[49] R. J. Cintra, F. M. Bayer, and C. J. Tablada, "Low-complexity 8-point DCT approximations based on integer functions," *Signal Processing*, 2014.

[50] G. A. F. Seber, *A Matrix Handbook for Statisticians*, ser. Wiley Series in Probability and Mathematical Statistics. Wiley, 2008.

[51] A. D. Poularikas, *Transforms and applications handbook*. CRC press, 2010.

[52] W. Ray and R. Driver, "Further decomposition of the Karhunen-LoÃ¨ve series representation of a stationary random process," *IEEE Transactions on Information Theory.*, vol. 16, no. 6, pp. 663–668, November 1970.

[53] G. Strang, *Linear Algebra and Its Applications*. Brooks Cole, Feb. 1988.

[54] Y. Arai, T. Agui, and M. Nakajima, "A fast DCT-SQ scheme for images," *Transactions of the IEICE*, vol. E-71, no. 11, pp. 1095–1097, Nov. 1988.

[55] R. C. Gonzalez and R. E. Woods, *Digital image processing*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

[56] W. Yuan, P. Hao, and C. Xu, "Matrix factorization for fast DCT algorithms," in *IEEE International Conference on Acoustic, Speech, Signal Processing (ICASSP)*, vol. 3, May 2006, pp. 948–951.

[57] W. H. Chen, C. Smith, and S. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Transactions on Communications*, vol. 25, no. 9, pp. 1004–1009, September 1977.

[58] E. Feig and S. Winograd, "Fast algorithms for the discrete cosine transform," *IEEE Transactions on Signal Processing*, vol. 40, no. 9, pp. 2174–2193, September 1992.

[59] C. Loeffler, A. Ligtenberg, and G. Moschytz, "Practical fast 1D DCT algorithms with 11 multiplications," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, May 1989, pp. 988–991.

[60] J. Liang and T. D. Tran, "Fast multiplierless approximation of the DCT with the lifting scheme," *IEEE Transactions on Signal Processing*, vol. 49, pp. 3032–3044, December 2001.

[61] M. Masera, M. Martina, and G. Masera, "Odd type DCT/DST for video coding: Relationships and low-complexity implementations," 2017.

[62] Z. Wang, "Fast algorithms for the discrete W transform and for the discrete Fourier transform," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-32, pp. 803–816, Aug. 1984.

[63] N. Suehiro and M. Hatori, "Fast algorithms for the DFT and other sinusoidal transforms," *IEEE Transactions on Acoustic, Signal, and Speech Processing*, vol. 34, no. 6, pp. 642–644, June 1986.

[64] Z. Wang, "Combined DCT and companding for PAPR reduction in OFDM signals." *J. Signal and Information Processing*, vol. 2, no. 2, pp. 100–104, 2011.

[65] B. G. Lee, "A new algorithm for computing the discrete cosine transform," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-32, pp. 1243–1245, Dec. 1984.

[66] F. S. Snigdha, D. Sengupta, J. Hu, and S. S. Sapatnekar, "Optimal design of JPEG hardware under the approximate computing paradigm," in *Proceedings of the 53rd Annual Design Automation Conference*. ACM, 2016, p. 106.

[67] T. Suzuki and M. Ikehara, "Integer DCT based on direct-lifting of DCT-IDCT for lossless-to-lossy image coding," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2958–2965, Nov. 2010.

[68] H. S. Hou, "A fast recursive algorithm for computing the discrete cosine transform," *IEEE Transactions on Acoustic, Signal, and Speech Processing*, vol. 6, no. 10, pp. 1455–1461, October 1987.

[69] C.-K. Fong and W.-K. Cham, "LLM integer cosine transform and its fast algorithm," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 6, pp. 844–854, 2012.

[70] K. Choi, S. Lee, and E. S. Jang, "Zero coefficient-aware IDCT algorithm for fast video decoding," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, 2010.

[71] M. Masera, M. Martina, and G. Masera, "Adaptive approximated dct architectures for hevc," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 12, pp. 2714–2725, 2017.

[72] J.-S. Park, W.-J. Nam, S.-M. Han, and S.-S. Lee, "2-D large inverse transform (16× 16, 32× 32) for HEVC (high efficiency video coding)," *JSTS: Journal of Semiconductor Technology and Science*, vol. 12, no. 2, pp. 203–211, 2012.

[73] M. Vetterli and H. Nussbaumer, "Simple FFT and DCT algorithms with reduced number of operations," *Signal Processing*, vol. 6, pp. 267–278, Aug. 1984.

[74] X. Xu, J. Li, X. Huang, M. Dalla Mura, and A. Plaza, "Multiple morphological component analysis based decomposition for remote sensing image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 5, pp. 3083–3102, 2016.

[75] M. T. Heideman and C. S. Burrus, *Multiplicative complexity, convolution, and the DFT*, ser. Signal Processing and Digital Filtering. Springer-Verlag, 1988.

[76] A. V. Oppenheim, *Discrete-time signal processing.* Pearson Education India, 1999.

[77] K. J. Horadam, *Hadamard Matrices and Their Applications.* Princeton University Press, 2007.

[78] T. I. Haweel, "A new square wave transform based on the DCT," *Signal Processing*, vol. 82, pp. 2309–2319, November 2001.

[79] K. Lengwehasatit and A. Ortega, "Scalable variable complexity approximate forward DCT," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 11, pp. 1236–1248, Nov. 2004.

[80] S. Bouguezel, M. O. Ahmad, and M. N. S. Swamy, "Low-complexity 8×8 transform for image compression," *Electronics Letters*, vol. 44, no. 21, pp. 1249–1250, 2008.

[81] ——, "Low-complexity 8×8 transform for image compression," *Electronics Letters*, vol. 44, no. 21, pp. 1249–1250, Sep. 2008.

[82] ——, "A fast 8×8 transform for image compression," in *2009 International Conference on Microelectronics (ICM)*, Dec. 2009, pp. 74–77.

[83] ——, "A novel transform for image compression," in *53rd IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug. 2010, pp. 509–512.

[84] ——, "A low-complexity parametric transform for image compression," in *Proceedings of the 2011 IEEE International Symposium on Circuits and Systems*, May 2011.

[85] ——, "Binary discrete cosine and Hartley transforms," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 4, pp. 989–1002, April 2013.

[86] C. J. Tablada, F. M. Bayer, and R. J. Cintra, "A class of DCT approximations based on the Feig–Winograd algorithm," *Signal Processing*, vol. 113, pp. 38–51, 2015.

[87] F. M. Bayer and R. J. Cintra, "DCT-like transform for image compression requires 14 additions only," *Electronics Letters*, vol. 48, no. 15, pp. 919–921, Jul. 2012.

[88] U. S. Potluri, A. Madanayake, R. J. Cintra, F. M. Bayer, and N. Rajapaksha, "Multiplier-free DCT approximations for RF multi-beam digital aperture-array space imaging and

directional sensing," *Measurement Science and Technology*, vol. 23, no. 11, p. 114003, October 2012. [Online]. Available: http://stacks.iop.org/0957-0233/23/i=11/a=114003

[89] W. K. Cham, "Development of integer cosine transforms by the principle of dyadic symmetry," in *IEE Proceedings I Communications, Speech and Vision*, vol. 136, no. 4, August 1989, pp. 276–282.

[90] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction To Algorithms*. MIT Press, 2001, ch. 16.

[91] R. J. Cintra and F. M. Bayer, "A DCT approximation for image compression," *IEEE Signal Processing Letters*, vol. 18, no. 10, pp. 579–582, Oct. 2011.

[92] K. Scharnhorst, "Angles in complex vector spaces," *Acta Applicandae Mathematica*, vol. 69, no. 1, pp. 95–103, 2001.

[93] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? A new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, Jan. 2009.

[94] V. K. Goyal, "Theoretical foundations of transform coding," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 9–21, September 2001.

[95] J. Katto and Y. Yasuda, "Performance evaluation of subband coding and optimization of its filter coefficients," *Journal of Visual Communication and Image Representation*, vol. 2, no. 4, pp. 303–313, December 1991.

[96] B. Flury and W. Gautschi, "An algorithm for simultaneous orthogonal transformation of several positive definite symmetric matrices to nearly diagonal form," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 1, pp. 169–184, Jan. 1986. [Online]. Available: http://dx.doi.org/10.1137/0907013

[97] D. S. Watkins, *Fundamentals of Matrix Computations*, ser. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts. Wiley, 2004.

[98] J.-R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards - including High Efficiency Video Coding (HEVC)," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1669–1684, Dec. 2012.

[99] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. C-23, no. 1, pp. 90–93, Jan. 1974.

[100] I.-M. Pao and M.-T. Sun, "Approximation of calculations for forward discrete cosine transform," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 3, pp. 264–268, Jun. 1998.

[101] USC-SIPI Image Database. University of Southern California. [Online]. Available: http://sipi.usc.edu/database/

[102] D. Salomon, G. Motta, and D. Bryant, *Data Compression: The Complete Reference*, ser. Molecular biology intelligence unit. Springer, 2007.

[103] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[104] P. Yip and K. Rao, "The decimation-in-frequency algorithms for a family of discrete sine and cosine transforms," *Circuits, Systems and Signal Processing*, vol. 7, no. 1, pp. 3–19, 1988.

[105] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*. San Diego, CA: Academic Press, 1990.