CERTIFICATION OF APPROVAL

WAVELET-DCT BASED IMAGE CODER

FOR VIDEO CODING APPLICATIONS

by

Chai Beng Seow

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Approved:

[illegible]

Assoc. Prof. Dr. Varun Jeoti
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

June 2007

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

[illegible]

Chai Beng Seow

ABSTRACT

This project is about the implementation of Wavelet-DCT intra-frame coder for video coding applications. Wavelet-DCT is a novel algorithm that uses Forward Discrete Wavelet Transform (DWT) to compute DCT. It is proved that the algorithm has better compression performance for difference images compared to conventional DCT. This is possible since the algorithm allows discarding insignificant DWT coefficients or more popularly known thresholding the DWT coefficients while computing the DCT.

In video coder applications, wavelet-DCT is capable to achieve greater compression. This project is a feasibility study on the performance of Wavelet-DCT in video coder applications. A SIMULINK model for conventional intra-frame coder is developed and tested, with very significant data bit reduction achieved. Then, the conventional DCT block has been replaced with a Wavelet-DCT block. In the study, on one hand, experiment is conducted on difference image for conventional intra-frame coder; on the other, the same difference image with Wavelet-DCT based intra-frame coder. The thresholding algorithm is used to remove some of the insignificant DWT coefficients from the difference image. The main objective is to achieve a better compression capability for difference image within video coding applications. The project's experimental results supports our claim that implementation of Wavelet-DCT in intra-frame coder within a video coding application could improve the system's performance with a greater compression ratio at the same Mean Squared Error.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Dr. Varun Jeoti, for introducing the world of video coder, the theory of DCT and wavelets. This project would not be successful without his guidance, encouragement and his enormous help during the final year of my undergraduate study.

I have spent my past eighteen years for my education, from elementary school, high school, to polytechnic, and university. Many people have played significant roles in my education at every stage and I feel lucky that there is always some one who took special interest in me, nurtured me, and guided me along the way. Hence, this dissertation is in part dedicated to them.

Most importantly, I would like to thank my parents who love me, teach me to be diligent, responsible and wise. Their sacrifice and support throughout my life have helped me enormously in this project. Besides, I would like to thank my brothers, sister and my love, Hui Yin for their support and love.

Last but not least, I would like extend my appreciation to all my friends in UTP, who support me morally throughout the project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CR	:	Compression Ratio
DTI	:	Continuous Tone Image
DCT	:	Discrete Cosine Transform
DFT	:	Discrete Fourier Transform
DIT	:	Decimation in Time
DTI	:	Discontinuous Tone Image
DSP	:	Digital Signal Processing
DWT	:	Discrete Wavelet Transform
FFT	:	Fast Fourier Transform
FPGA	:	Field Programmable Gate Arrays
IDCT	:	Inverse Discrete Cosine Transform
JPEG	:	Joint Photographic Experts Group
MJ2K	:	Motion JPEG 2000
MPEG	:	Moving Picture Experts Group
MSE	:	Mean Squared Error
PSNR	:	Peak Signal-to-Noise-Ratio
QT	:	Quantization Table
VHDL	:	Very High Speed Hardware Description Language

CHAPTER 1
INTRODUCTION

A video coder is a device or software module that compresses and decompresses digital video. Compression can be both lossy and lossless. Variety of video coder technologies emerges and flourishes, with most of them lossy. Discrete Cosine Transform (DCT), the most commonly used transform in video coder, is a transform with near-optimal energy compaction property. Wavelet-DCT is an algorithm that utilizes Discrete Wavelet Transform (DWT) as a tool to compute DCT. Under normal circumstances where no approximations are made, the algorithm performs exactly like the Fast DCT algorithm. The approximation part comes in when intermediate coefficients are dropped in order to improve compression. This capability is not available in other classical Fast DCT because they do not use wavelet transform. The approximation capability is integrated into the DCT hence no additional resources are required. Therefore, this algorithm has the potential to improve compression without loosing computational speed.

An overview of general video coder system is introduced in Chapter 1. This chapter also discusses about the development of Wavelet-DCT and problem statement, which emphasizes the essentiality of a video coder with greater compression capability deployable over wide range of applications. Lastly, this chapter discusses the specific project objectives and scope.

1.1 Background of Study

A typical video coder application is illustrated by Figure 1. Digital video source obtained by a video camera is encoded (compressed) with an encoder, before it is stored or being transmitted over a communication network. The communication network could be both wired or wireless, with certain network load limitation. Then, the decoder at the receiver part will decode (decompress) the received data, before it is displayed at the receiver side. The encoder and decoder are the fundamental blocks of a video coder application, essential for video conferencing or multimedia streaming applications. The interest behind is to reduce the network load size while maintaining reasonably good video quality. Video coder compresses digital video by a drawback in between the video quality (bit rate) and also the computational complexity (coding algorithms) and robustness to data losses and errors [1].

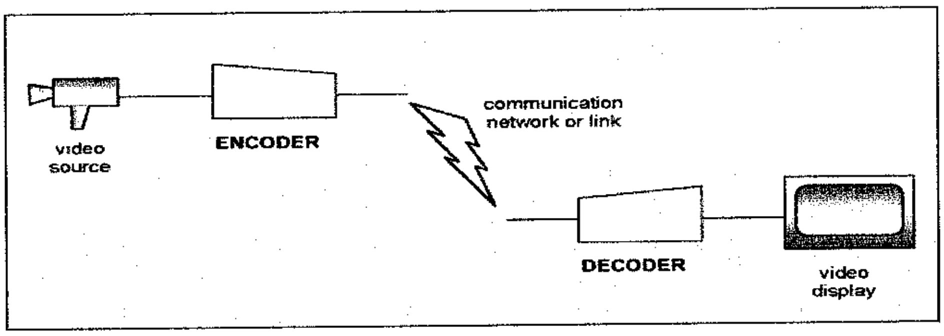


Figure 1 : Video coder for video streaming over communication network

DCT could be considered the most important computational tool in video and image coder applications. However, JPEG 2000 has introduced a state-of-the-art compression technique based on wavelet technology, which is proven to have superior compression performance and some additional features over baseline JPEG, which uses DCT [2], [3]. Nevertheless, JPEG 2000 is designed for a natural image, where the colour value changes continuously, and there is no strong edges. With an inter-frame coder, the inputs of the intra-frame coder within a video coder are mostly difference images, which are obtained by subtracting current frame of image with the previous frame. Subtraction is performed to minimize the data size to compress. Difference images are discontinuous tone images; which causes wavelet transform based compression technique not deployable for video coder applications. Eventually, DCT is still the favoured transform technique for video coder applications.

1.1.1 Discrete Cosine Transform (DCT)

Fourier Transform could decompose a signal into its component frequencies and amplitude. Like Fourier Transform, Discrete Cosine Transform (DCT) is a unique technique that has near-optimal energy compaction property [4]. With DCT, a natural image's energy could be compacted to perform compression, by dropping the low energy part. Although DCT involves many complex multiplications and additions, it can be simplified to computation efficiency comparable to Fast Fourier Transform (FFT) [4]. These advantages make DCT extensively used in image compression.

1.1.2 Fast Discrete Cosine Transform (FDCT)

Since the advent of Very Large Scale Integration (VLSI) technology that enables cheap fabrication of FDCT hardware, many video and still image coder applications have utilized the power of DCT for energy compaction. Although FDCT has been deployed extensively, some features could not be provided by classical DCT. One of the disadvantages is that it is not suitable for Discontinuous Tone Images (DTI), where difference images are DTI.

1.1.3 Wavelet Transform

The wavelet theory is a new mathematical tool that has rapidly grown into a large research area in the intersection of mathematics, electrical engineering and statistics. The wavelet transform is used in multi-resolution analysis which allows one to zoom in on local signal behaviour to analyze signal details and zoom out to get a global view of the signal. It is able to represent a signal in both time and frequency domain, which has its energy concentrated in time. In other words, low and high frequency components of a signal at a particular time get separated and concentrated. The future of signal processing appears to be gearing towards the direction of wavelets. [5] has shown that wavelet basis is the optimal basis for data compression, noise reduction and statistical estimation.

1.1.4 Discrete Wavelet Transform (DWT)

The Discrete Wavelet Transform (DWT) is the sampled version of wavelet transform. It is gaining popularity at a fast pace in the field of digital signal processing recently because of its easy implementation methods of which the most intuitive of all is the filter bank method. What is amazing for this method is that the calculations involving values of discrete wavelet functions are nowhere used. Hence, although wavelets may be complicated, by making use of the simple relationship between wavelet domain coefficient of the previous and present levels, DWT can be simple to implement. Implementation of DWT is further discussed in Chapter 2.

1.1.5 Wavelet-Discrete Cosine Transform (Wavelet-DCT)

[6] suggested Wavelet-DCT, an algorithm that uses DWT to calculate DCT as a new method of performing Fast DCT. The reason is because wavelet transform created a sparse matrix for the input data which can be exploited for the purpose of thresholding. Under normal circumstances where no data is being discarded, performing Wavelet-DCT would result in exact same results as the conventional DCT algorithm. What other algorithm could not achieve is the capability for performing thresholding for intermediate DWT coefficients.

The algorithm also has better compression performance while retaining the same video quality. Approximation is done cleverly to discard intermediate coefficients. In other words, the thresholding does not cause significant distortion to the signal but removes unimportant DWT coefficients of the data to give better compression ratio.

Although [6] showed the mathematical proof and ideas to apply Wavelet-DCT, no feasibility study has been performed on signal processing applications. Besides, difference image compression still utilizes conventional DCT for video coder applications. Therefore, the motivation of the project is to study the feasibility for implementation of Wavelet-DCT in compressing difference images, hoping that it could gain the benefits of both algorithms, DWT and DCT.

1.2 Problem Statement

Typical video coder consists of an intra-frame and inter-frame coder. Inter-frame coder (temporal coder) compares the current frame of image with the previous frame, and performs subtraction to obtain the difference image before being compressed by intra-frame coder. Therefore, the source to intra-frame coder is basically difference images. Most video coder nowadays utilizes DCT to compact image energy in the design of the intra-frame coder. Nevertheless, Motion JPEG 2000 (MJ2K), which is video adaptation of JPEG2000 standard, does not have an inter-frame coder in its architecture and each frame is an independent entity. This individual frame is encoded by either a lossy or lossless variant of JPEG 2000 intra-frame coder. However, the standard is still not well-deployed, since MJ2K integrated circuits for hardware is still unavailable [7].

Though DCT is extensively used in intra-frame coder design, it is proven that DCT does not efficiently compact energy of discontinuous tone images [8]. Therefore, it is feasible to improve the algorithm.

Wavelet based image compression is not something new in the field image processing. However, most studies conducted are only in the wavelet domain alone, not integrating with DCT computation. Since wavelets are proven to perform tremendous compression capability for still images, the implementation of Wavelet-DCT should produce positive results, while gaining the benefits of both DWT and DCT.

1.3 Project Objectives

The interest of this project is to study on the current compression technique and improve the compression performance of the intra-frame coder in compressing difference images. The objectives are:

- To develop a SIMULINK based simulation model for an intra-frame coder of a video coder.
- To develop Wavelet-DCT block using MATLAB Version 7.0 and to integrate it into the intra-frame coder model
- To simulate, analyze and study on the performance of DCT on different types of image.
- To improve the compression performance for DTI with Wavelet-DCT and deliver the performance study results for both of the algorithms.

1.4 Project Scope

- The intra-frame coder developed is a replica model from a typical video coder.
- The intra-frame coder simulation model shall be developed using SIMULINK.
- The developed Wavelet-DCT block is a self-coded embedded MATLAB function, which could be integrated into SIMULINK environment.
- The codes for Run-Length and Huffman Coder are borrowed from [12] and integrated into the SIMULINK model.
- The input to the intra-frame coder could be difference images or natural images.
- The project does not cover full video coder development, but only the intra-frame coder.
- The project is a "proof-of-concept" for Wavelet-DCT implementation onto an intra-frame coder.
- The DWT performed is only a one level forward DWT and limited to Haar Wavelet Transform.
- To deliver performance study results for both of the algorithms.

1.5 Chapter Organization

Chapter 1 of this interim report introduces about the project title and its problem statements, objectives and scope of studies. Chapter 2 gives an overview about video coder and then reviews the literature and theoretical details of an intra-frame coder. The sections covered are DCT, 2-D DCT, integer DCT, Quantization and also Entropy Coding. Then, the DWT and Wavelet-DWT are also discussed in Chapter 2. The methodologies used to develop the simulation model in SIMULINK are presented in Chapter 3. It also shows the integration of embedded MATLAB codes into the model. The next chapter is the results and discussion, which discusses about the output obtained from the simulation model. Some graphical results for compression ratio against mean-square-error are also shown and discussed, both for DCT and Wavelet-DCT. The final chapter concludes the entire document with some future recommendations.

CHAPTER 2
LITERATURE RIVIEW

Theories and literature of the project is introduced and reviewed in this chapter. It starts with brief introduction of general video coder architecture. Then, it discusses about an intra-frame coder, which is the simulation model built for this project. All the compression techniques within an intra-frame coder are also discussed, such as DCT, Quantization and also Entropy Coding. Then, DFT and classical FFT are reviewed to relate it to classical Fast DCT. Finally, the novel Wavelet-DCT and its approximation of DCT are introduced. The thresholding concept for the Wavelet-DCT based coder is explained as well.

2.1 General Video Coder Architecture

Figure 2 shows the simplified version of a general video coder. A typical video coder consists of three main blocks, as listed below:

- Inter-Frame Coder
- Intra-Frame Coder
- Run-Length and Huffman Coder

An inter-frame coder typically consists of motion estimator and motion compensator, which are not discussed in detail here. These blocks compare current frame of image with previous fame, and subtract the current frame to the previous frame to obtain the difference between them. The motivation of the inter-frame coder is to reduce size of individual frame input to the intra-frame coder. With 38 frames per second in video source (MPEG), the difference between two frames is very small, and the resultant pixels values are mostly zeroes.

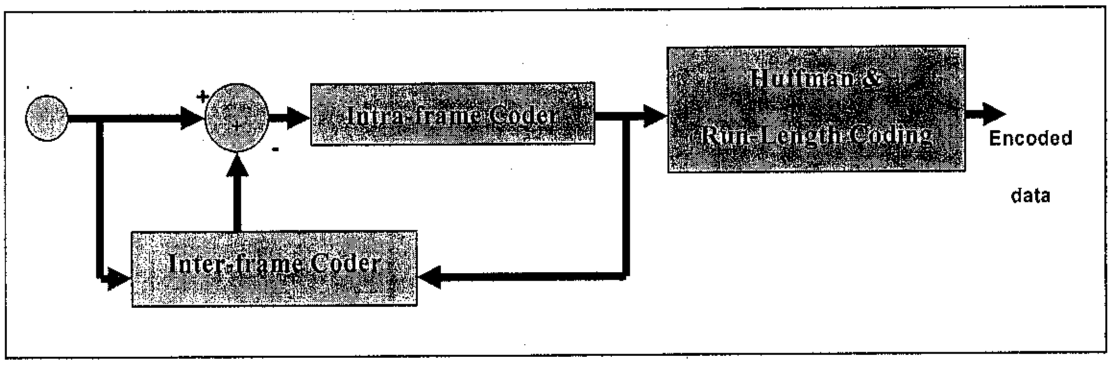


Figure 2 : A simplified version of a typical video coder model

With difference image as the input to the intra-frame coder, the net input size becomes smaller, compared to input size of the original image frame. Within the intra-frame coder, it performs DCT to compact the energy of difference image. However, it is known that DCT is not an efficient technique to compress difference image which is basically DTI [8]. Nevertheless, there is no better algorithm discovered to replace DCT till now, and DCT is still used extensively for video coder applications in current technology.

2.2 Intra-Frame Coder

Intra-frame coder is similar to the still image coder used for still image (static or non-moving image) compression. An intra-frame coder performing DCT operations will break the input images into blocks of 8x8, 16x16 or 16x8 sizes, followed by quantization of the DCT coefficients and entropy coding of the results. Figure 3 shows the intra-frame encoder model, with all important elements listed below, which will be further discussed in the coming sub-sections.

i. Discrete Cosine Transform (DCT) and inverse DCT
ii. Quantization and De-quantization
iii. Entropy Coder – Encoder and Decoder.

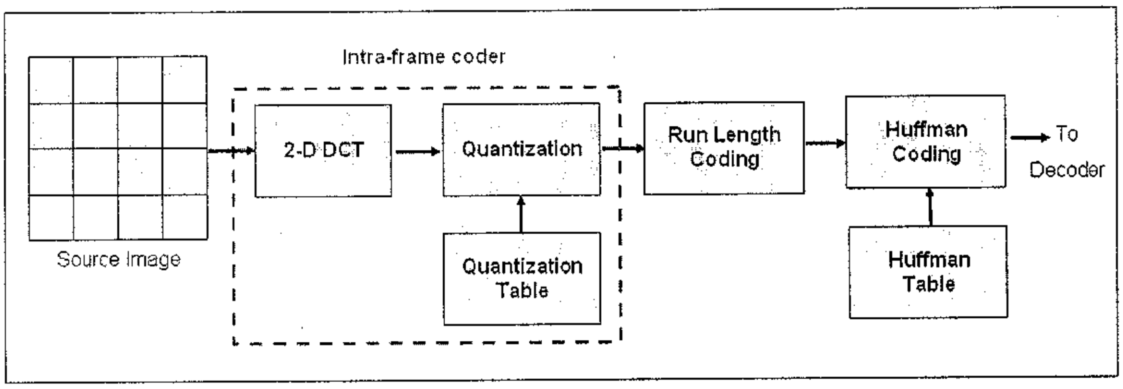


Figure 3 : Intra-frame encoder model

2.2.1 *Discrete Cosine Transform (DCT)*

DCT is very similar to DFT, with several advantages over DFT. The motivations of DCT are: (a) the DCT coefficients are real-valued; (b) the DCT has near-optimal property for energy compaction; and (c) the DCT can be computed efficiently using fast algorithms that are computationally comparable to the FFT [4]. These advantages make DCT extensively used in image compression.

To obtain the DCT a length-N sequence $x(n)$, the discrete signal must be firstly made symmetric by the following mapping:

$$x(n) = \begin{cases} x(n), & 0 \le n \le N-1 \\ x(2N-1-n), & N \le n \le 2N-1 \end{cases} \tag{2.1}$$

This is to eliminate any discontinuities while integrating the segments of signal. Then, the DCT of $x(n)$ can be mathematically defined as:

$$C_X(k) = \alpha(k) \sum_{n=0}^{N-1} x(n) \cos\frac{(2n+1)k\pi}{2N} \tag{2.2}$$

$$where \qquad \alpha(0) = \frac{1}{\sqrt{N}}, \quad \alpha(k) = \sqrt{\frac{2}{N}}, \quad 1 \le k \le N-1 \tag{2.3}$$

The inverse DCT is shown below:

$$x(n) = \sum_{k=0}^{N-1} \alpha(k) C_x(k) \cos\frac{(2n+1)k\pi}{2N} \tag{2.4}$$

The DCT are with several important properties, as listed below:

Property 1: The DCT is real and orthonormal.

Property 2: The DCT is a fast transform. The DCT of an N-vector can be computed in $O(N \log_2 N)$ operations.

Property 3: The DCT has excellent energy compaction for highly correlated data. [4]

The DCT transforms a signal or image from the spatial domain to the frequency domain. Normally, the input image is divided in block of 8x8 pixels and then transformed by using DCT. However, DCT is a 1-D transformation algorithm which is not applicable in image processing. Thus, 2-D DCT since image is 2-D in nature.

2.2.2 2-D DCT

The 2-D DCT of a $N \times N$ sequence $x(n_1, n_2)$ is defined by:

$$C_x(k_1,k_2) = \alpha(k_1)\alpha(k_2) \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} x(n_1,n_2) \cos\frac{(2n_1+1)\pi k_1}{2N} \cos\frac{(2n_2+1)\pi k_2}{2N} \tag{2.5}$$

$$\textit{where} \quad \alpha(k_1) \textit{ and } \alpha(k_2) \textit{ are defined as in } (2.3)$$

The 2-D DCT is a separable transform that can be performed in terms of 1-D DCT using the row-column decomposition method. The inverse 2-D DCT is defined by:

$$x(n_1,n_2) = \sum_{k_1=0}^{N-1} \sum_{k_2=0}^{N-1} \alpha(k_1)\alpha(k_2) C_x(k_1,k_2) \cos\frac{(2n_1+1)\pi k_1}{2N} \cos\frac{(2n_2+1)\pi k_2}{2N} \tag{2.6}$$

An example application of 2-D DCT in image compression is shown in Figure 4 Figure 4 , where the 64 (or 256, dependent on the block sizes) pixels of a block of image data x(i,j), are transformed to its corresponding 8x8 DCT coefficients, X(i,j).

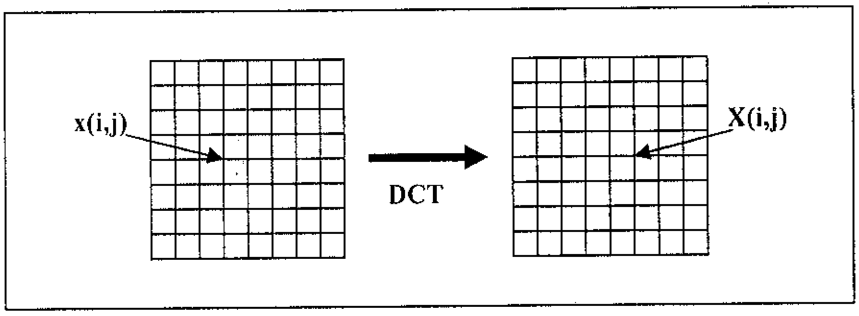


Figure 4 : 8x8 pixels transformed into 8x8 DCT coefficients redrawn from [17]

These DCT coefficients represent the values of the DCT basis functions as illustrated in Figure 5 .

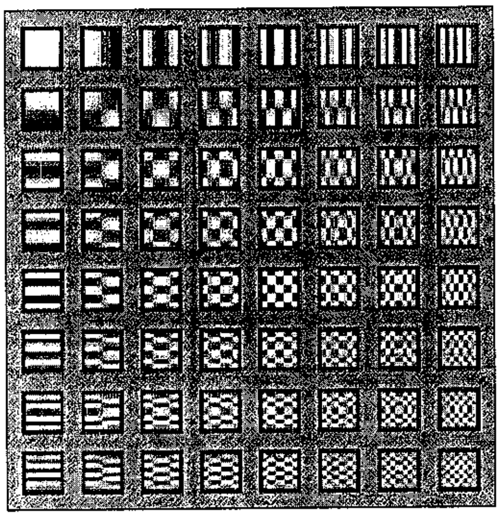


Figure 5 : The DCT basic functions, borrowed from [17]

The top left portions of the functions shown are with the lowest frequencies while the bottom right portions are with the highest frequencies. As the energy tends to concentrate at the basic functions with lower frequencies, the coefficients will have greater value on the top left portion, as shown in Appendix A.

2.2.3 Quantization

After performing the DCT, each of the 64 (or 256 for 16x16 block size) DCT coefficients is quantized in conjunction with a carefully designed 64-element Quantization Table. There are different Quantization Tables available for Continuous Tone Images (CTI) and Discontinuous Tone Images (DTI). DTI has a uniform step size of 16 while CTI quantization table does not introduce uniform step size, but variable step size for different coefficients at different space location. Please refer to [9] for the CTI Quantization Table. The quantized coefficients are then rounded to integer values, to ensure efficient implementation on hardware, with only fixed-point computations. From [9], the coefficients values shown in Appendix A are quantized to the quantized values shown in Appendix B.

Note that there are lots of zeros but only few non-zero concentrated at the left top corner. At the decoder, the quantized values are multiplied by the corresponding QT elements to recover the original dequantized values. However, quantization is a lossy algorithm where the quantized values can not actually be dequantized back to its original unquantized values (or can not be fully recovered), but much closed to its original values. This introduces lost of information in quantization.

2.2.4 Entropy Coder

After quantization, all of the quantized coefficients are ordered into the "zigzag" sequence as shown in Figure 6 . This ordering helps to facilitate entropy encoding by placing low-frequency non-zero coefficients before high-frequency coefficients. For natural images which are basically CTI, its reordered coefficients are preceded by non-zeroes values, followed by mostly all zeroes in the tail of the sequence.

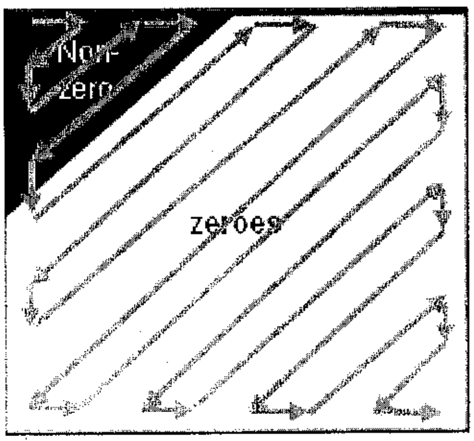


Figure 6 : Zigzag sequence

Upon reordering the coefficients, the coder achieves additional compression losslessly by encoding the quantized DCT coefficients based on their statistical characteristics. The baseline JPEG standard uses both Run-Length and Huffman coding.

2.2.5 Run-Length Coding (RLC)

RLC is a very simple method for compression of sequential data. RLC is an average compression method that works by counting the number of adjacent pixels with same gray level values (This count is called run length). Run length coding is easily implemented, either in software or in hardware. It is fast and very well verifiable, but its compression ability is very limited.

2.2.6 Huffman Coding

Huffman Coding exploits the input probabilities for the image and it is a lossless compression technique. It is based on the fact that in an input stream certain tokens occur more often than others. Typical images give compression ratio in the range of 1:1.2 to 1:2.5. The followings steps are used to perform Huffman Coding, where an example to illustration the steps is available in Appendix F.

1. Find the gray level probabilities for the image by finding the histogram.
2. Order the input probabilities (histogram magnitudes) from smallest to largest.
3. Combine the smallest two by addition until only two values remain.
4. Go to step 2, until only two probabilities are left.

5. By working backward along the tree, generate code alternating assignment

2.3 Reviewing DFT and FFT

The sequence of N complex numbers $x_0,\ldots,x_{N-1}$ is transformed into the sequence of N complex numbers $X_0,\ldots,X_{N-1}$ by the DFT according to the formula:

$$X(k) = \sum_{n=0}^{N-1} x(n)W_N^{kn}, \qquad 0 \le k \le N-1 \tag{2.7}$$

$$W_{N=e}^{-j2\pi/N}$$

DFT can be efficiently calculated using different FFT algorithms but only Radix-2 Decimation in Time (DIT) variant of the FFT is discussed here. It can be shown mathematically or graphically.

2.3.1 Mathematical Representation

The N point DCT is broken into two N/2 point DFT by specifying the input signal into odd and even numbered samples to get:

$$X(k) = \frac{1}{N}\sum_{m=0}^{N/2-1} x(2n)W_N^{2mk} + \frac{1}{N}\sum_{m=0}^{N/2-1} x(2n+1)W_N^{2m+1} \tag{2.8}$$

Assume that the input signal has been divided by 1/N (normalized), this yields:

$$x_1(m) = EVEN\ SAMPLES = x(2n)$$
$$x_2(m) = ODD\ SAMPLES = x(2n+1)$$

By substitution into equation (X)

$$X(k) = \sum_{m=0}^{N/2-1} x_1(m)W_{N/2}^{mk} + \sum_{m=0}^{N/2-1} x_2(m)W_{N/2}^{(2m+1)k} \tag{2.9}$$

The odd twiddle factor is still different from the even part. To simply the equation, the DFT definition $W_N^{2mk} = W_{N/2}^{2mk}$ is used. Thus, by manipulation and substitution:

$$X(k) = \sum_{m=0}^{N/2-1} x_1(m)W_{N/2}^{mk} + \sum_{m=0}^{N/2-1} x_2(m)W_{N/2}^{mk} \tag{2.10}$$

Now, there are two N/2 point DFT which take less time to work out than one N point

DFT. This process of decimation each DFT Is continued until we reach a series of two point DFT.

2.3.2 *Graphical Representation*

Figure 7 shows the block diagram of the last stage of length-8 radix-2 DIT FFT. The input data are separated into even and odd groups before going through a length-4 DFT block. The even-odd separation mechanism is shown in Appendix C where the binary bits are reshuffled.

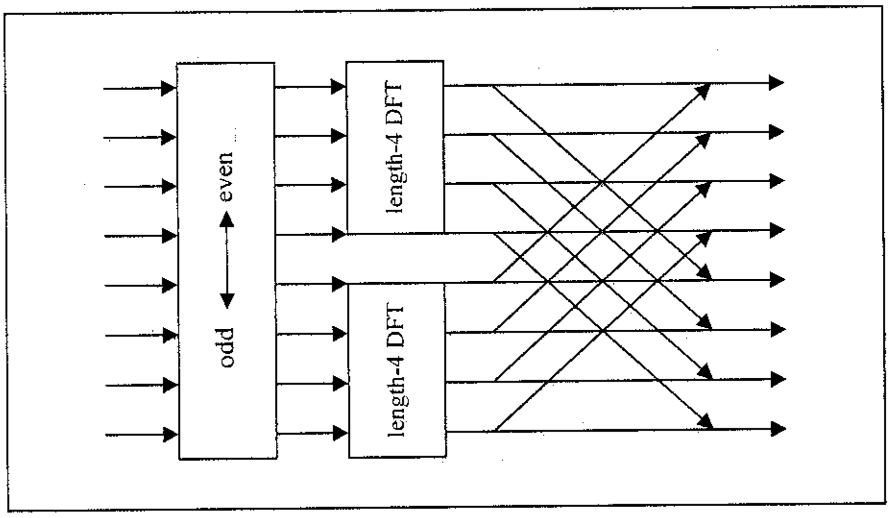


Figure 7 : Block diagram illustrating Decimation-In-Time (DIT) FFT, reproduced from [10]

Finally, the butterfly operation is used to combine the short length DFT into longer length DFT. Figure 8 shows the details of the butterfly operations. One complex multiplication involving the twiddle factor W_N^r and two complex additions (A, B) is performed for every butterfly operation.

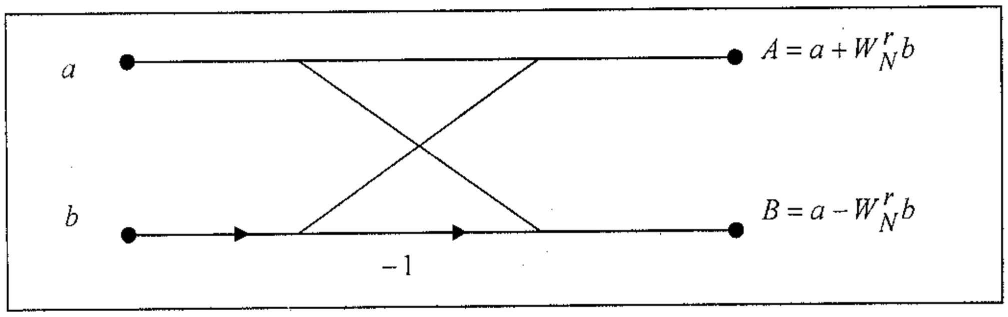


Figure 8 : Butterfly operations in Radix-2 DIT-FFT, reproduced from [10]

Equation (2.11) gives an insight from the matrix point of view. F_N is an NxN DFT matrix (i.e. $F_N(m,n) = e^{-j2\pi mn/N}$) where $m,n \in \{0,1,\ldots,N-1\}$ are row and column indices respectively. S_N is the separation matrix used for the even-odd separation. $I_{N/2}$ is the N/2 point identity matrices and $T_{N/2}$ is the respective twiddle factor matrices that do the butterfly operation. $F_{N/2}$ is the N/2 point complex exponentials matrix that will convolve smaller blocks of the input sample, X.

$$F_N = \begin{bmatrix} I_{N/2} & T_{N/2} \\ I_{N/2} & -T_{N/2} \end{bmatrix} \begin{bmatrix} F_{N/2} & 0 \\ 0 & F_{N/2} \end{bmatrix} S_N X \qquad (2.11)$$

2.4 DWT-based DCT and Its Approximation

The structure of DWT based DCT is very similar to the classical FFT, except that the bit-reversal and odd-even index separation process is now replaced by computationally intensive forward DWT. Figure 9 shows the block diagram of the last stage of the length-8 Approximate DCT. This figure is an equivalent comparison to Figure 7 .

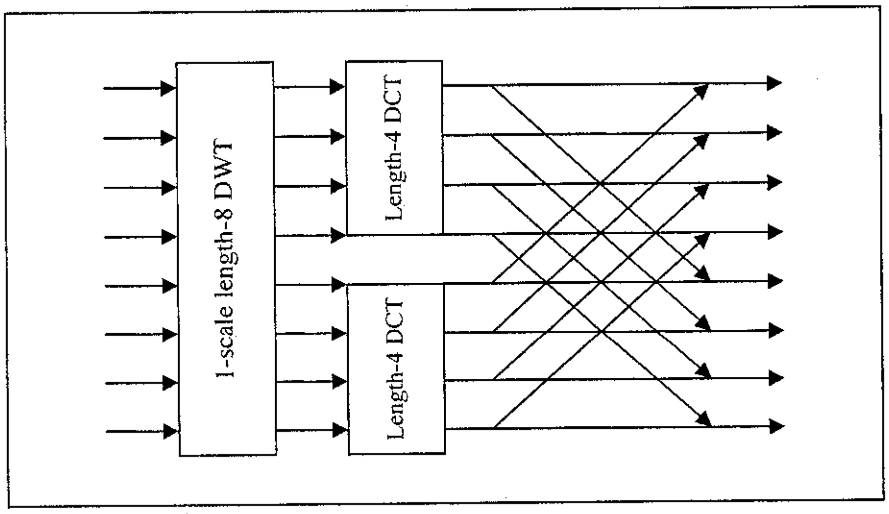


Figure 9 : Block diagram illustrating Wavelet-DCT

The data is separated into detail and approximation part via one scale wavelet transform before going through a length-4 DCT block. After that, the results of the DCT are combined using an equivalent butterfly operation to form back the length-8 DCT. The detail of the equivalent butterfly operation is shown in Figure 10 .

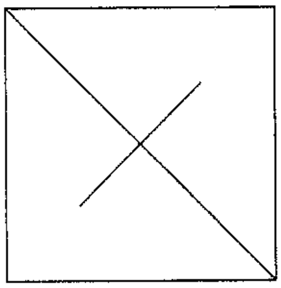


Figure 10 : Equivalent cross-diagonal butterfly operation matrix BN, reproduced from [6]

[6] suggested Equation (2.12) as an implementation of the Wavelet-DCT. Equation (2.12) shows the matrix form of the Wavelet-DCT.

$$C_k = B_N \begin{bmatrix} C_{N/2} & 0 \\ 0 & C_{N/2} \end{bmatrix} \begin{bmatrix} I_{N/2} & 0 \\ 0 & (-1)^m I_{N/2} \end{bmatrix} H_N \tag{2.12}$$

$$where\ B_N = \begin{bmatrix} \cos\left(\frac{k\pi}{2N}\right) & P\sin\left(\frac{k\pi}{2N}\right) \\ -P\cos\left(\frac{k\pi}{2N}\right) & \sin\left(\frac{k\pi}{2N}\right) \end{bmatrix}$$

$$n, k \in \{0,1,\ldots,N/2-1\}$$

where B_N is the $N \times N$ butterfly operation, I_N is the $N \times N$ identity matrix, and H_N is the $N \times N$ one level Haar wavelet matrix

The DCT using DWT has a slightly higher order of computational complexity as compared to DIT-FFT. It becomes an approximation when pruning or thresholding is performed on the intermediate wavelet coefficients to speed up computation.

2.4.1 Implementation of DWT

Three common methods are identified for forward DWT operation:

- Filter banks
- Lifting scheme
- Lattice filters

Filter bank method is used in this project because it the most intuitive of all three methods. It follows the definition that the next level wavelet and scaling coefficients are obtained by a convolution with **h**$_0$ and **h**$_1$. The highest level scaling coefficient is taken to be the sampled points of the input data. Thus, the DWT is obtained by filtering with impulse response **h**$_0$ and **h**$_1$. The entire DWT operation then looks very similar to Quadrature Mirror Filtering as shown in Figure 11 . The attractive part is that **h**$_0$ and **h**$_1$ are also governed by a simple relationship.

$$h_1(n) = (-n)^n h(N-1-n) \tag{2.13}$$

Amazingly, calculations involving the values of the discrete wavelet functions are no where used. Although wavelets ay be complicated two-parameter functions, the simple relationship between previous and present levels wavelet domain coefficients make the actual implementation of DWT fairly simple.

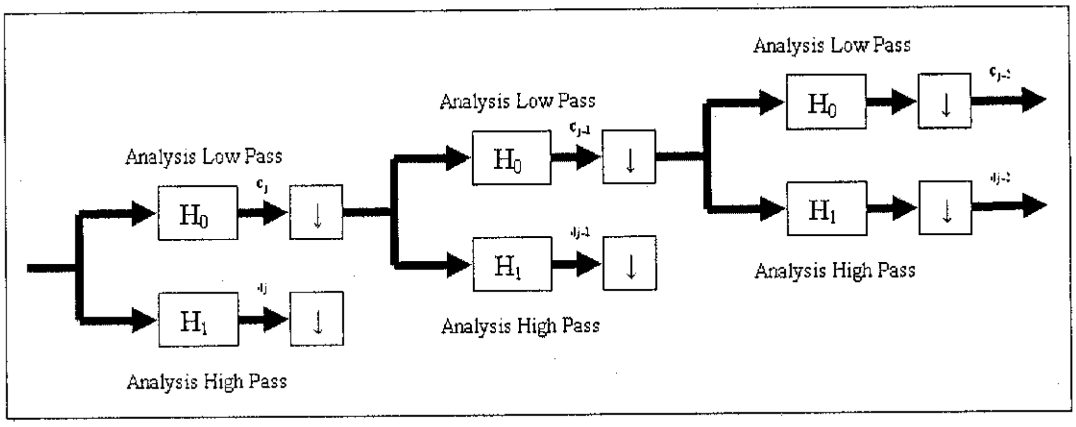


Figure 11 : Implementation of 3 level forward DWT, reproduced from [10]

Referring to Figure 11 , only the filter **h**$_0$ and the scaling function are required to completely specify the DWT relations. This corresponds to simply setting the limits of our implementation which is the number of forward transforms and the relationship complexity between two levels of coefficients. The wavelet theory maybe more complex than Fourier theory but the implementation of DWT is much simpler than implementing DFT.

2.5 Performance Parameters for Systems

There are several important parameters used throughout this project as means of quantizing improvement for system performance. The parameters are Bit rate (Bit per

Pixel or BPP), Compression Ratio (CR), Mean Squared Error (MSE) and Peak Signal-to-Noise-Ratio (PSNR). BPP is average number of bits per each pixel in the encoded image, while CR is the compression ratio obtained from the encoding process, both used to measure system's compression capability. BPP and CR can be mathematically represented as:

$$BPP = \frac{N_C}{N_O}(8\, bits) \tag{2.14}$$

$$CR = \frac{N_O}{N_C} \tag{2.15}$$

Where N_C is the number of pixels in the compressed image while N_O is the number of pixels in the original image.

MSE is the square of the error of the output, which makes the output to differ from the quantity estimated. PSNR is the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. Both of these parameters are used as a measure of quality of reconstruction in image compression, and are mathematically represented as:

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\|I(i,j) - K(i,j)\|^2 \tag{2.16}$$

$$PSNR = 10 \bullet \log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20 \bullet \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \tag{2.17}$$

Where m is the image vertical length and n is its horizontal length. I is the original image while K is the decoded image. MAX is the maximum pixel value, which is 255, since the pixel value is represented by 8-bit, from 0 to 255.

CHAPTER 3
METHODOLOGIES/PROJECT WORK

Chapter 3 describes the implementation strategy, problems encountered in the design and the solution to those problems. It starts by justifying the correct engineering tools used in this project, which then discusses the use of the tools and the methodologies to develop the intra-frame coder model. Development of every part of the model is also presented. Then, methods used to generate input to the model and the simulation procedures are described. Finally, the design of the Wavelet-DCT algorithm, its coding and its integration into simulation model are explained. Appendix D contains the MATLAB code for implementation of the Wavelet-DCT.

3.1 Project Preparation/Familiarization Stage

The project begins by establishing proper project objectives, which are feasible, and realizable. This has taken time limitation and facility availability into account. The initial interest of this project is to study the current video compression technique and to further improve its compression capability, if feasible. Being realistic, the genuine idea is to model the intra-frame coder (or still image coder) of a video coder. Therefore, the project background is studied to review on the literature details, such as DCT, quantization, zigzag scanning and Entropy coding technique, which are important sub-systems of the intra-frame coder. Upon learning these well-established theoretical concepts, a suitable modeling language is identified, which is SIMULINK. SIMULINK is a very powerful and dynamic engineering tool, usable for sophisticated and complex mathematical and system modeling. It offers wide selection of blocksets, especially for Digital Signals Processing (DSP) applications. It is also very flexible, with capability to integrate with embedded MATLAB code, which is self-defined. SIMULINK offers solutions that enable engineers to reduce significant time to model or develop systems for the computing, digital signal processing and many other applications.

3.2 Intra-frame Coder Development by SIMULINK

The SIMULINK is equipped with Video and Image Processing Blockset for video coder development, such as built-in 2-D DCT block, 2-D IDCT block and many other vector manipulation (reshape, elements re-arrangement) blocks. All these blocks are fundamental building blocks for intra-frame coder model. The intra-frame coder is built and simulated as illustrated in Figure 12 .

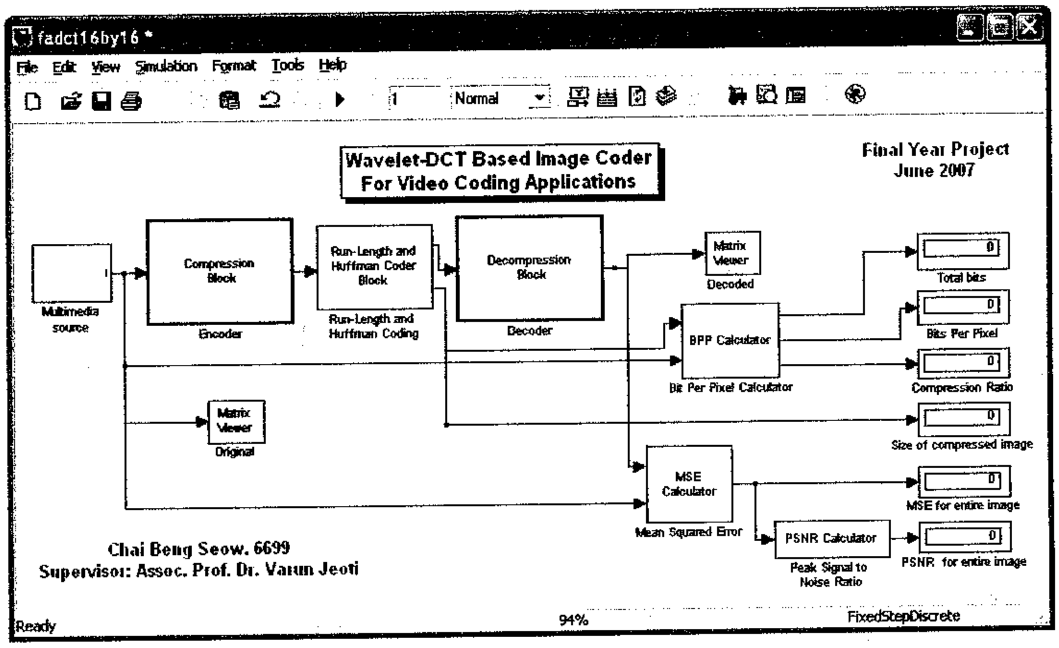


Figure 12 : Intra-frame coder

The intra-frame coder consists of several sub-systems, such as encoder, Run-Length and Huffman encoder and decoder. The multimedia source is the input block for test images or video source. The matrix viewer "Original" is used to visualize the input multimedia file while the decoded output is displayed on the matrix viewer named "Decoded".

The test image is compressed by the encoder, which is followed by the Run-Length coder and Huffman coder. This is the point where maximal compression is achieved and several readings for the encoded (compressed) image are measured, which are total bits, BPP, CR, size of the compressed image, MSE and PSNR.

Then, the encoded image data is decoded by the inverse part of the Huffman and Run-Length Coder, before being transmitted to the decoder. The decoder de-compresses the image into a decompressed form of image, which consists of some level of MSE, due to the lossy compression.

3.2.1 The Encoder

Figure 13 shows the internal blocks of the encoder (compression block), which includes 2-D DCT, quantization and zigzag scanning.

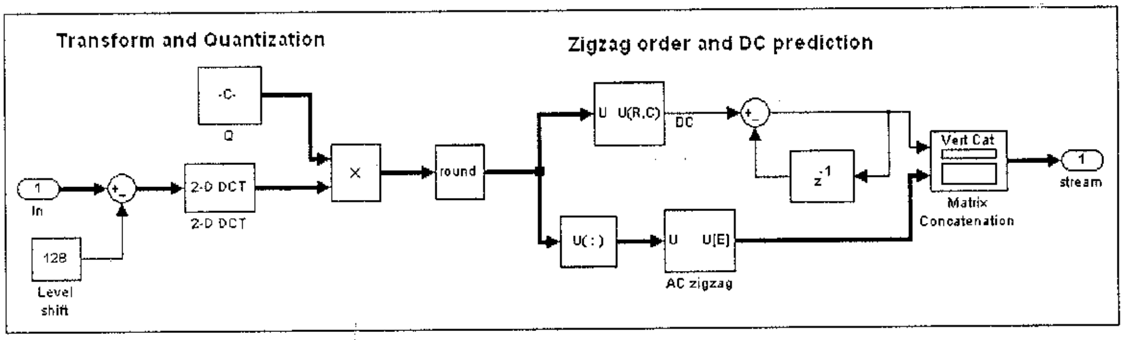


Figure 13 : The encoder of the intra-frame coder

As discussed in subsection 2.2.2 , 2-D DCT is an essential part intra-frame coder. SIMULINK offers a built-in 2-D DCT block, which is then used in the simulation model to transform the original 8x8 blocks from image data into its corresponding sets of 8x8 DCT coefficients, as shown in Figure 13 . Prior to performing the operation, the input data's pixel value is firstly deducted with a value of 128, to reduce the bit per pixel value of the original image from 8 bpp (ranging from 0 to 255) to 7 bpp (ranging from 0 to (255 – 128)). This will improve the system's computation efficiency. Then, the 2D-DCT block is used in the model after the BPP reduction.

The DCT coefficients are then quantized by using the standard quantization coefficients obtained from [9]. A vector of quantization coefficients (8x8) is divided from the 8x8 DCT coefficients to perform quantization. Two different quantization tables are used for CTI and DTI, as discussed in subsection 2.2.3 . Please refer to [9] for the said CTI quantization table. After that, the quantized values are rounded, to ease computations with only integers.

Then, to perform zigzag re-ordering, a vector selector block is used to nonlinearly select quantized coefficients. This is a re-ordering of the vector elements, where the elements will be finally concatenated with the DC value (most top left DCT coefficient).

3.2.2 Run-Length and Huffman Coder

Run-Length and Huffman Coder of the intra-frame coder are implemented by using an embedded-MATLAB function block, which allows user to integrate MATLAB function into SIMULINK. The "Embedded MATLAB Editor" allows user to call MATLAB code and customize the required function not available in SIMULINK. The initial attempt was to use the available MATLAB function for Huffman coding to perform the required task, which include "huffmanenco" and "huffmandeco". However, the use of these functions required the generation of a dictionary from a source with known probability model. This means that the dictionary is generated while reading and analyzing the vector of data, which is not realizable in real-time. Since Entropy coding is not the focus of this project, the code is borrowed from [12]. Initially, the code from [12] could not work, as it only processes data in array format, instead of vector. Therefore, the code is updated accordingly, so that it recognizes vector data.

3.2.3 The Counterpart of Encoder: Decoder

The decoder is the inverse part of the encoder. It performs the inversed operations of the encoder in inverse ordering, starting with de-zigzag scanning, dequantization and finally 2-D IDCT. The decoder is illustrated in Figure 14 .

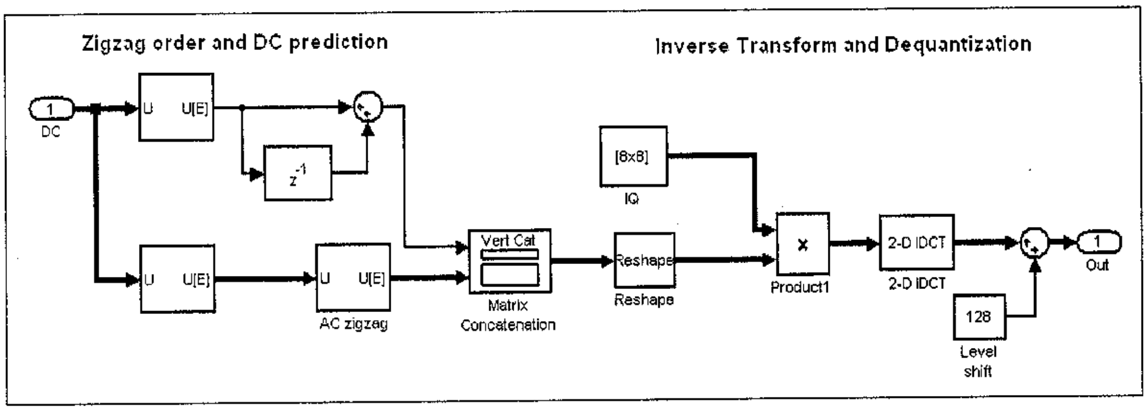


Figure 14 : The decoder of the intra-frame coder

3.2.4 *System's Performance Measurement Blocks*

SIMULINK blocks are used to construct the PSNR calculator, MSE calculator and also the BPP calculator to perform the mathematical computations for the required readings. Figure 15 shows the MSE Calculator built with SIMULINK block, an example of the measurement blocks that performs the equation (2.16) discussed in section 2.5 .

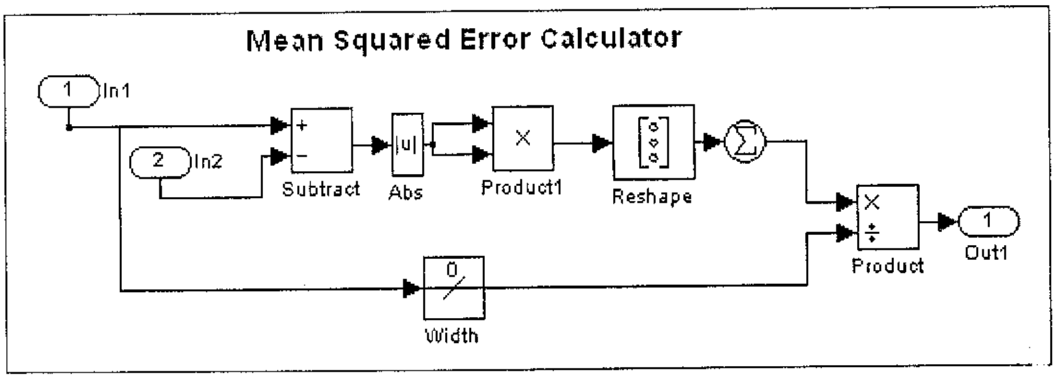


Figure 15 : Mean Squared Error calculator of the intra-frame coder

3.2.5 *Simulation Procedures*

Multimedia file, such as image or video source are inputted into in the system to perform intra-frame compression. The MSE calculator will subtract the pixel values of the input file from the output files. This implies that the total number of pixels within the input and output files must be identical. If input pixels fail to find their corresponding (same coordinate) pixels in the output file, subtraction could not be performed and system error occurs. Therefore, it is important to ensure that input file is dividable by 16x16 or 8x8 block sizes. Then, the system is run to provide the decoded image and the necessary readings at the output.

3.3 Wavelet-DCT Based Intra-frame Coder Development by SIMULINK

The Wavelet-DCT block is programmed in MATLAB code, where the MATLAB code is documented in Appendix D. Then, the Wavelet-DCT block is integrated into the encoder part, as shown in Figure 16 by using embedded-MATLAB function block.

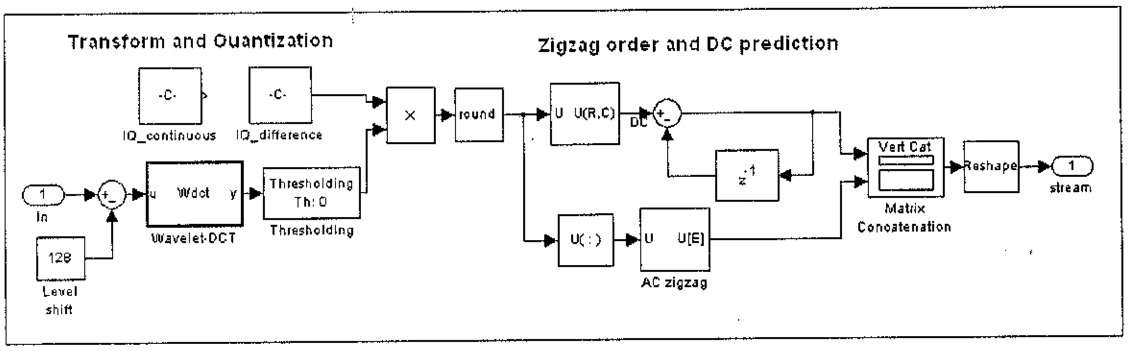


Figure 16 : The encoder for the Wavelet-DCT based intra-frame coder

Other than the Wavelet-DCT block, an additional DTI Quantization table is added to the encoder and decoder too. This allows the user to selectively choose in between the two quantization tables, dependent on the input image type. A "thresholding" block is also included to the system to adjust the system's compression ratio.

3.3.1 Wavelet-DCT Code Development

Code for implementation of Fast Approximate Fourier Transform (FAFT) by [11] is referenced in designing and coding the Wavelet-DCT code. The Wavelet-DCT is based on Equation (2.12), where H_N, the Haar Wavelet of length-8 is obtained by using the Forward DWT function "forwardDWTN.m" from [11],[18]. A half size DCT (dctmtx(4)) is coded for $C_{N/2}$ by borrowing code from "dctmtx.m" from MATLAB. Then the identity matrixes $I_{N/2}$ and $(-1)^m I_{N/2}$ are easily coded with "eye(4)" functions.. Finally, the modified equivalent butterfly function B_N is based on the matrix in Equation (2.12).

B_N function initially caused problem, which is finally solved by transposing the other matrixes of Equation (2.12) to obtain the B_N. This is done by expecting the output to be equivalent to the dctmtx(8) function, if an identity matrix input is used. Then, transposing all the other matrixes and multiplying them with the dctmtx(8), as shown by Equation (3.1) will give the B_N. The output shows the expected equivalent cross-diagonal butterfly function as illustrated in Figure 10 .

$$B_N = C_k \begin{bmatrix} C_{N/2} & 0 \\ 0 & C_{N/2} \end{bmatrix}' \begin{bmatrix} I_{N/2} & 0 \\ 0 & (-1)^m I_{N/2} \end{bmatrix}' H_N' \qquad (3.1)$$

Where the symbol (') denotes transpose of matrix

After obtaining the B_N matrix, the output matrix coefficients for Wavelet-DCT are identical with the DCT matrix, if thresholding is not introduced. The functionality of the Wavelet-DCT in comparison to the conventional DCT is firstly analyzed in MATLAB before porting the code into SIMULINK environment.

The Wavelet-DCT is then integrated into the intra-frame coder. This is done with the use of embedded-MATLAB function block.

3.3.2 Data Generation

Multimedia file like natural still image is a type of CTI. These conventional images are the input to the system directly. However, since the system only can process images with size dividable by 8x8 or 16x16 blocks, the input images must be properly cropped before being used. This can be achieved by any photo-editing software applications, or MATLAB itself.

For difference images, which are basically DTI, the image is obtained by subtracting the subsequent frame to the reference frame. The two subsequent frames from a video source are obtained by using MATLAB code, as documented in Appendix E. Then, the current frame is subtracted from previous frame, as illustrated in Figure 17 .

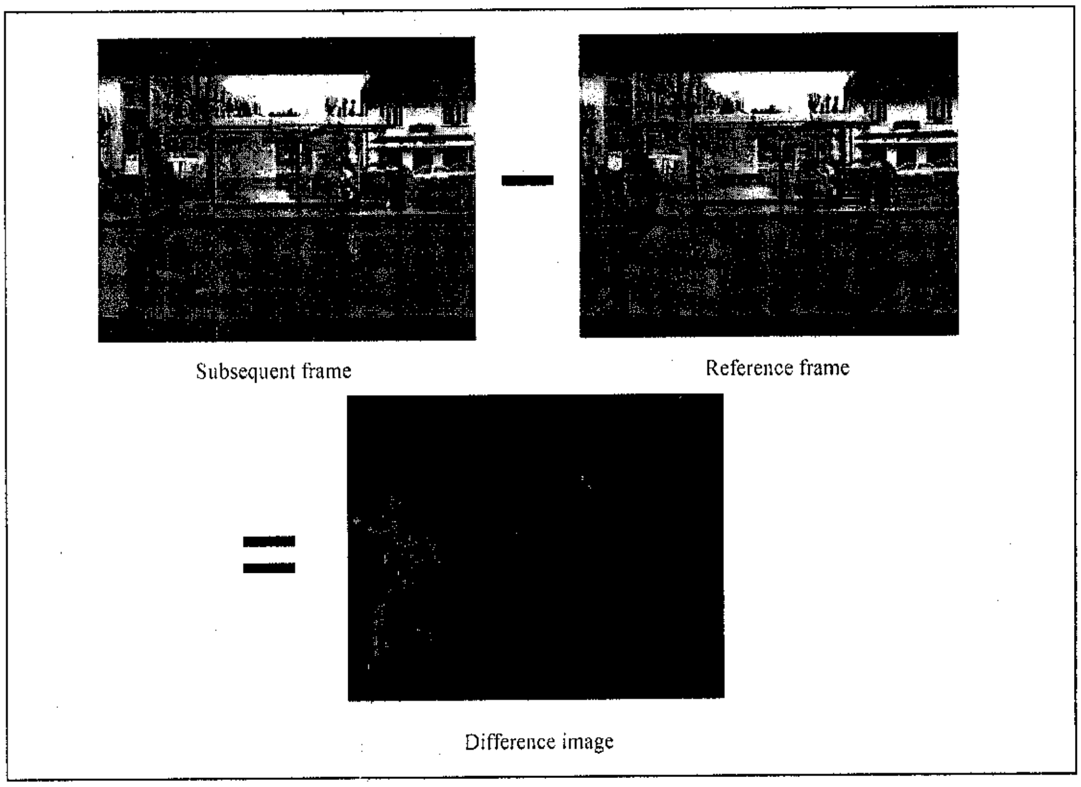


Figure 17 : Obtaining difference image from 2 subsequent frames.

Other difference images can be obtained with same technique. The difference image is used as DTI source for the intra-frame coder, while CTI source is based on natural images.

3.3.3 Simulation Procedure for Wavelet-DCT based Intra-frame Coder

To test the Wavelet-DCT code's functionality, an identity matrix of 8x8 and 16x16 are used as the input. The output is equivalent with pure 8x8 and 16x16 DCT matrixes obtained from the dctmtx(8) and dctmtx(16) respectively.

Then, for the Wavelet-DCT based Intra-frame Coder (after the code integration), the DTI and CTI images are inputted as source. Then the simulation procedures are the same as in subsection 3.2.5 .

3.3.4 Thresholding

The idea of thresholding in this project is to remove the unimportant DWT coefficients, within the Wavelet-DCT algorithm. Thresholding occurs after the DWT operation, where the image energy gets compacted to their frequency bands after the DWT operation. The frequency band compaction is very image dependent. The experiment shows that DWT is able to compact the pixel values' energy well. The dominant frequency band will have very large value, with very small values (lower than value of '1') for non-dominant frequency bands within an image. Therefore, these non-significant values could be removed since they are unimportant.

Thresholding is done by choosing an appropriate threshold level where fixed-value thresholding scheme is applied. A small value of '1' or '2' is used as the threshold level, since any coefficients beyond these values are considered important and should be retained. This threshold level could be varied with the mean to remove or preserve more coefficients. If conventional DCT operation is desired, then the Threshold value must be set to '0'.

3.3.5 Data Collection

The simulation is performed to record the results of the compression performance of the conventional intra-frame coder and the Wavelet-DCT based intra-frame coder. By using the Wavelet-DCT based intra-frame coder, the conventional intra-frame coder is achieved by setting the threshold level to '0'. Then, different threshold levels are applied to compute the corresponding compression ratios and MSE for Wavelet-DCT based intra-frame coder. The readings are then recorded and tabulated using tables.

Several different types of commonly used test images (CTI) as well as DTI obtained from subsection 3.3.2 are utilized for this simulation. The comparison of compression performance for DTI for both coders: conventional intra-frame coder and Wavelet-DCT based intra-frame coder are presented using graphs. The compression ratio versus MSE graph is useful indication of the relationship between the two variables. The graph is plotted with CR and MSE obtained for various DTI images.

CHAPTER 4
RESULTS AND DISCUSSION

Chapter 4 shows the outcome of the simulation conducted according to the methodologies discussed in Chapter 3. It is split into two parts, for CTI and DTI. It shows the simulation output (decoded image, compression ratio, BPP, MSE) and discusses about the coders' compression performance, for both 16x16 and 8x8 block intra-frame coder. The discussion is based on the figures displayed.

4.1 Results and Discussion for Continuous Tone Images

The intra-frame coders of 16x16 and 8x8 blocks are firstly tested with several standard test images from [13]. A standard test image is used to test image processing and image compression algorithms across different institutions [13]. This standardizes all results and makes comparison of results, both visually and quantitatively feasible among different labs. These images are mostly natural and typical images that image processing technique needs to deal with.

However, there are also different types of test images which yield different compression performance. Two test images (256x256, 8 bpp) with different spatial and frequencies characteristics are selected for this simulation: Baboon, and Lena.

4.1.1 Image with Great Details and Low Predictability: Baboon

Test image Baboon has lot of details, where it contains component in high frequencies area with low predictability. This means that Baboon has low redundant image, which is difficult for compression. In other words, compression ratio achievable with Baboon is low. The simulations results for Baboon test image, including its decoded image, bit rate (bpp), compression ratio, MSE, PSNR are shown in this sub-section. Figure 18   shows the original Baboon and the encoded image for Baboon using 16x16 block intra-frame coder.

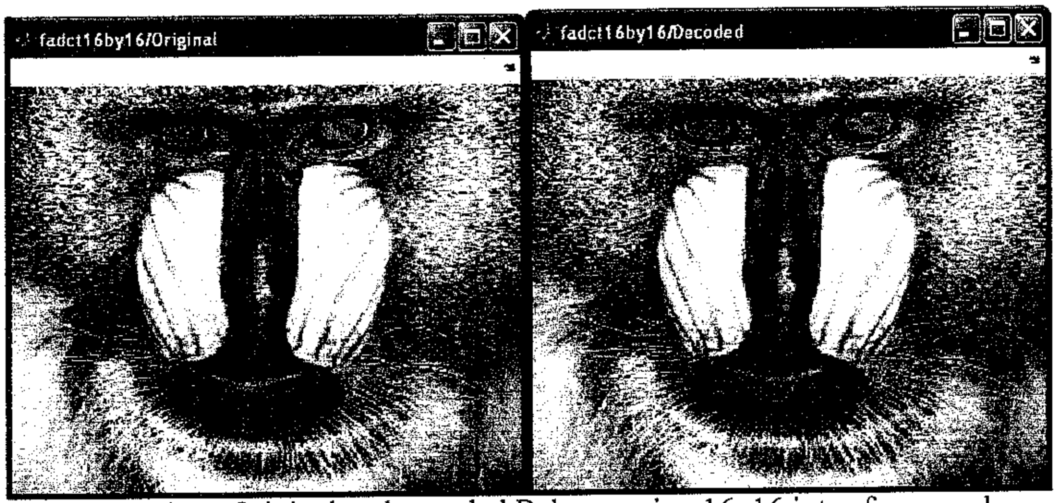


Figure 18   : Original and encoded Baboon using 16x16 intra-frame coder.

Other than the decoded image, the simulation model also shows the performance readings for the Baboon, as shown in Figure 19   . These readings are then tabulated in Table 1   .

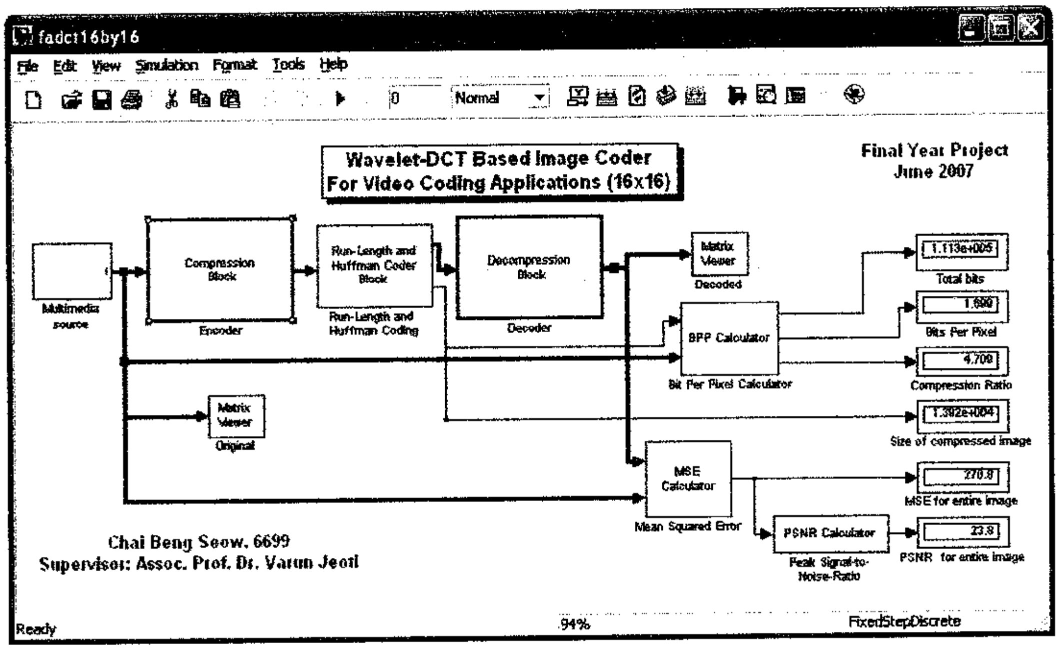


Figure 19   : Data rate, compression ratio, PSNR and MSE of Baboon using 16x16 intra-frame coder.

Figure 18 shows that the decoded Baboon is almost identical with the input original Baboon. Visually, difference of the two images is too small where it is undistinguishable. With this quality of decoded Baboon, the system provides compression ratio of 4.709 or a bit rate of 1.699 bbp. The compression ratio is based on input image of 8bpp. The MSE is low at 270.8, which proves that reasonably good quality is remained. The low compression ratio here proves that Baboon test image is with high detail and low predictability. The same experiment then performed by using 8x8 block intra-frame coder, where the decoded output is shown in Figure 20 .

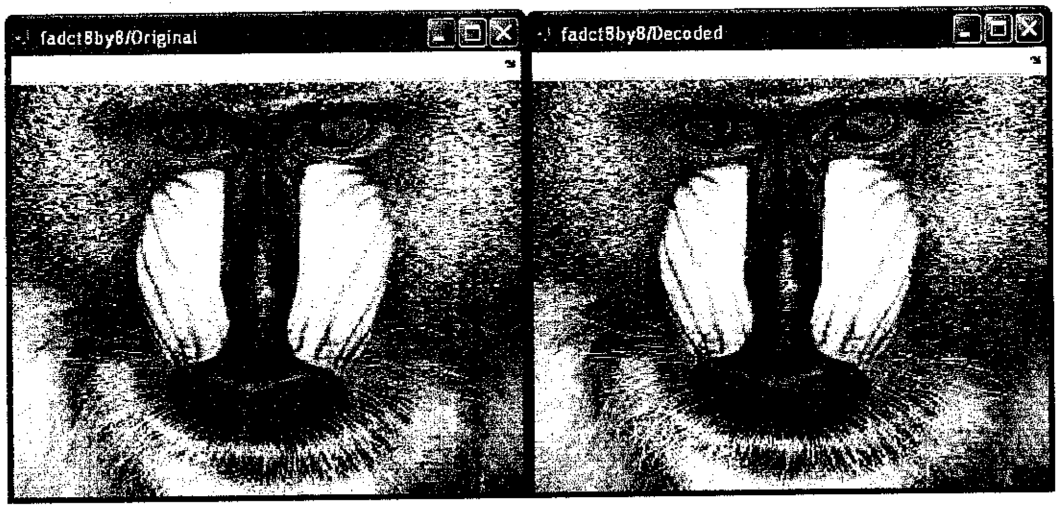


Figure 20 : Original and encoded Baboon using 8x8 intra-frame coder.

Again, visually the decoded image is not differentiable from the input image for the 8x8 block. However, it is in fact quantitatively very different from the 16x16 block, as summarized in Table 1 .

Table 1 : Compression performance for Baboon

Block size	Bit rate (bpp)	Compression ratio	MSE	PSNR
16x16	1.699	4.709	270.8	23.8
8x8	1.865	4.289	195	25.23

Table 1 shows that 8x8 block intra-frame coder compresses the Baboon at lower ratio, at 4.289 times while it's corresponding bit rate is 1.865 bpp. However, the reduction in compression ratio improves the image quality, where the MSE decreases to 195 and the PSNR is slightly higher at 25.23. There is no significant performance superiority in this case, since it is very difficult to judge as the increment in compression ratio is basically to introduce more error to the decoded image. However, it proves that the simulation model works well for the Baboon test image.

4.1.2 Image with Average Details and Higher Predictability: Lena

The experiment is continued with Lena, which is one of the most popular test images [13]. Lena is a natural and typical image with average details and higher predictability compared to Baboon. This means that Lena should yield a higher compression ratio compared to Baboon. Figure 21 shows the original Lena and the encoded Lena using 16x16 block intra-frame coder.

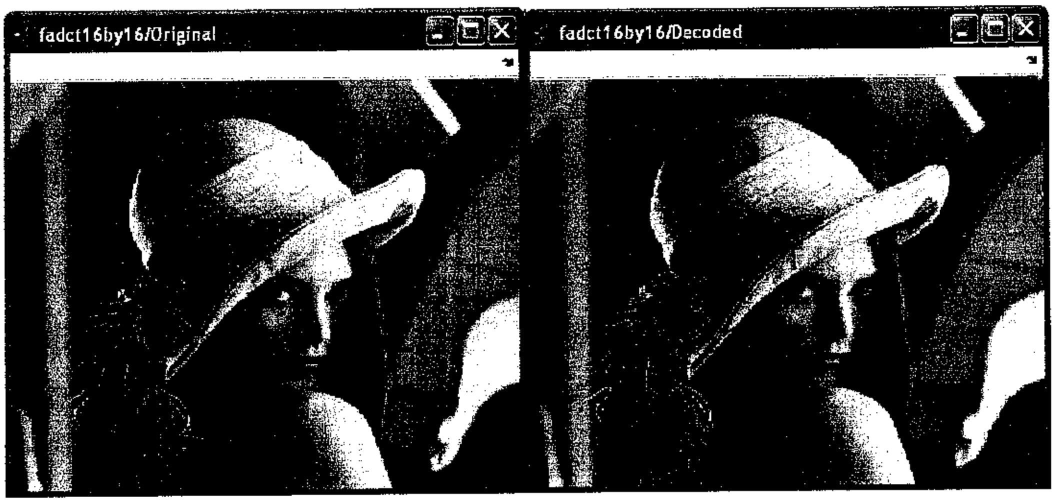


Figure 21 : Original and encoded Lena using 16x16 intra-frame coder.

The compression performance readings for Lena using 16x16 intra-frame coder is summarized in Table 2 , with comparison with the performance of 8x8 intra-frame coder. The decoded Lena is seen to have little artifacts but the quality is still very good compared with the input Lena.

Then, Lena is compressed using 8x8 intra-frame coder, to examine the performance difference. Figure 22 shows the original and encoded Lena using 8x8 intra-frame coder.

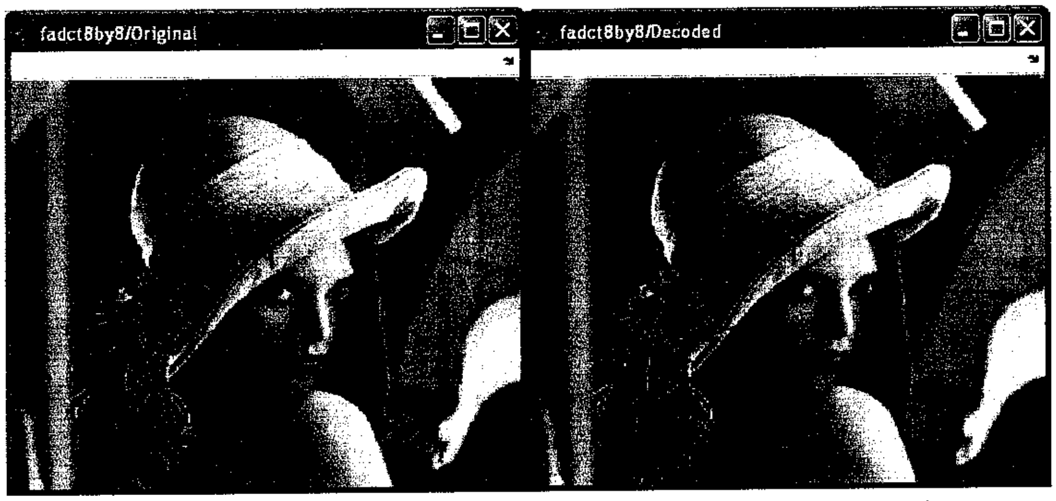


Figure 22 : Original and encoded Lena using 8x8 intra-frame coder.

The encoded Lena using shown in Figure 22 still retains very good quality. Visually, there is no difference for decoded Lena for 16x16 block and 8x8 block intra-frame coder. However, its quantitative outputs are recorded and summarized in Table 2 , to show the actual performance difference of the two coders.

Table 2 : Compression performance for Lena

Block size	Bit rate (bpp)	Compression ratio	MSE	PSNR
16x16	0.9761	8.196	77.47	29.24
8x8	1.145	6.986	51.72	30.99

Table 2 shows that 8x8 intra-frame coder has a higher compression ratio for Lena at 6.986 and bit rate of 1.145 bpp while the compression ration for 16x16 intra-frame coder is 8.196 with bit rate equals to 0.9761 bpp. However, the additional compression is achieved by increment of MSE, where 16x16 coder provides MSE of 77.47 with PSNR 29.24, while 8x8 coder has a MSE of 51.72 and PSNR of 30.99. Therefore, performance superiority is unjustified again, which is also not the interest of this project.

Comparing the performance of the two different images, it is realized that the Baboon with greater details has lower compression ratio compared to the Lena, with lower details. Thus, it shows that compression ratio is yielded higher with image of lower details, which is normally associated with dominant colors or patents particularly.

Difference image acquired from subsection 3.3.2 is one of the examples of this low details image.

4.2 Results and Discussion for Discontinuous Tone Images

The intra-frame coders of 16x16 and 8x8 blocks are then tested with difference image of DTI nature. Difference images have very little details, since most of the contents are black or very low values, such as '0' or '1'. This makes difference images to yield very high compression ratio via the intra-frame coder. To perform compression on difference image, a video file is used to obtain its video frames. Then, the video frame no. 351 is subtracted from its frame no. 350, where the methodology is shown in subsection 3.3.2 . Then, the difference image is firstly tested with 16x16 intra-frame coder, where the decoded image for this difference image of frame 351-350 is shown in Figure 23 .

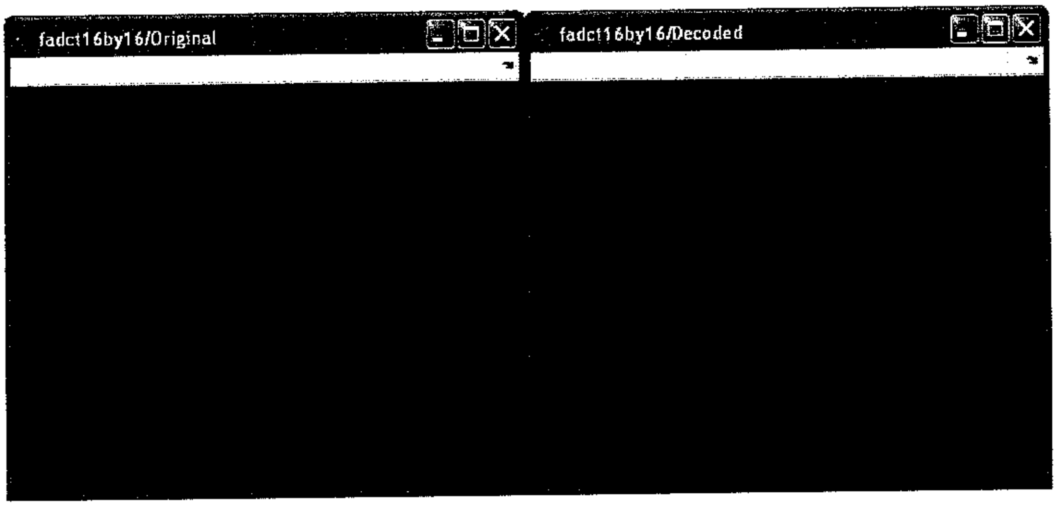


Figure 23 : Original and encoded difference image of frame 351-350 using 16x16 intra-frame coder.

Then, the same difference image is simulated with 8x8 intra-frame coder, where the decoded image for this difference image of frame 351-350 is shown in Figure 24 .

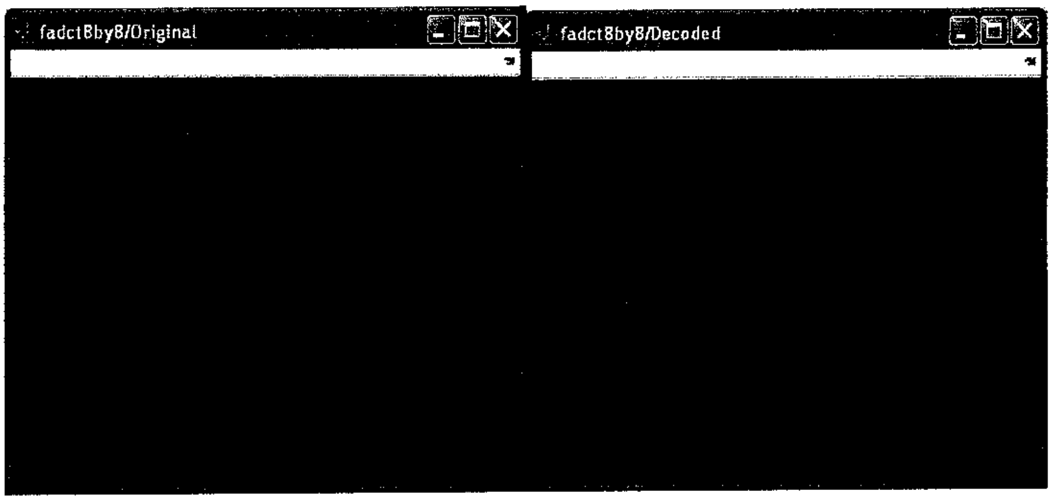


Figure 24 : Original and encoded difference image of frame 351-350 using 8x8 intra-frame coder.

Obviously, there is not much that could be discussed from the visual inspection of the decoded difference image from the two coders. Therefore, comparison of the quantitative results is necessary to evaluate on the performance, which is summarized in Table 3 .

Table 3 : Compression performance for difference image of frame 351-350

Block size	Bit rate (bpp)	Compression ratio	MSE	PSNR
16x16	0.1693	47.25	3.832	42.3
8x8	0.2771	28.87	3.42	42.79

Table 3 shows that difference image is able to achieve very significantly higher compression ratio compared to the natural CTI images. With 16x16 block size, it is able to achieve up to compression ratio of 47.25 and bit rate of 0.1693 bpp, while with 8x8 block size, the compression ratio is lower at 28.87 with bit rate of 0. 2771. This shows that compression ratio for difference image with low details could be compressed at higher ratio with 16x16 intra-frame coder, for this particular case. In addition, it is the same as for the CTI, where 16x16 normally gives higher compression ratio with higher MSE. The reason behind this phenomenon is that 16x16 block has longer run-length of zeroes compared to 8x8. The longer run-length of zeroes is applicable for difference images only since most of the pixel values of difference images are zeroes.

However, this phenomenon is not always true and it is very dependent on the input image. Some difference image might be compressed better by the 8x8 block size also. Generally, Table 2 just proves that it is not necessary that 8x8 will always give higher compression ratio as compared to 16x16 block size. Since the interest of this project is to perform Wavelet-DCT, which performs DWT algorithm internally, it is better to choose 16x16 block size. This is because that DWT can compact the frequency band better with longer size vector, where it treats the 16x16 block as 16 vectors of length equals to 16.

4.3 Results and Discussion for Discontinuous Tone Images with Wavelet-DCT based Intra-frame Coder

This section discusses about the performance of the Wavelet-DCT based intra-frame coder for difference images (DTI), as compared with the conventional intra-frame coder. Both of the coders are based on 16x16 block size. Several difference images are obtained from a video file, where f_n is the current frame while f_{n-1} the previous frame. There is no particular reason why the video file is used, since the interest is to obtain the difference images within its video frames, but not the video file. Since difference images mostly consists of pixel values '0', any other video file will be able to produce difference images of DTI nature. In this case, f_n can be frame 351 of the video file while f_{n-1} will be frame 351-1 = frame 350.

Table 4 summarizes the quantitative results of inter-frame compression performed with 16x16 block, for several arbitrary chosen difference images from the video file. The table compares the performance of DCT with Wavelet-DCT, in term of the bit rate, compression ratio, MSE and PSNR. Conventional DCT based intra-frame coder is performed with threshold value set to '0' while Wavelet-DCT based intra-frame coder is achieved by setting threshold value to '1' or '2'. This threshold level is manually determined with aim to improve the compression performance for difference images. This is done by trial and error methods.

Table 4 : Performance Comparison for DCT and Wavelet-DCT on several arbitrary difference images with 16x16 block size.

Difference Image $(f_n - f_{n-1})$	DCT				Wavelet-DCT			
	BR	CR	MSE	PSNR	BR	CR	MSE	PSNR
$(f_{301} - f_{300})$	0.1556	51.42	1.83	45.51	0.1552	51.56	1.824	45.52
$(f_{251} - f_{250})$	0.0957	83.57	2.698	43.82	0.0942	84.94	2.692	43.83
$(f_{351} - f_{350})$	0.1693	47.25	3.832	42.3	0.1658	48.25	3.842	42.29
$(f_{551} - f_{550})$	0.2637	30.34	3.879	42.24	0.262	30.54	3.885	42.24
$(f_{261} - f_{260})$	0.2454	32.6	4.575	41.53	0.2447	32.69	4.581	41.52
$(f_{401} - f_{400})$	0.3853	20.76	6.118	40.26	0.3789	21.11	6.109	40.27

Where BR is Bit Rate, CR is Compression Ratio, MSE is Mean Squared Error and PSNR is Peak Signal to Noise Ratio.

From Table 4 it is proved that Wavelet-DCT based intra-frame coder has high possibility to compress the difference images at higher compression ratio. Table 4 shows that all compression ratios for Wavelet-DCT are always higher than the conventional DCT. Most importantly, the additional compression ratio is achieved not at the tradeoff of MSE. It is shown that some of the difference images show improvement in the MSE when compression ratio increases.

This achievement is a tremendous one since there are improvements in both factors. Furthermore, difference image itself yields very high compression ratio, where additional compression rate is normally very difficult to achieve. Even though the compression ratio improves at very low margin, this is actually very significant for difference image compression, which it typically consists of most zeroes within its original image.

Figure 25 illustrates the comparison of bit rate performance for the two algorithms for difference images. Though the two lines are much closed to each other, the pink line (Wavelet-DCT) is always smaller in its bit rate as compared to blue line (DCT).

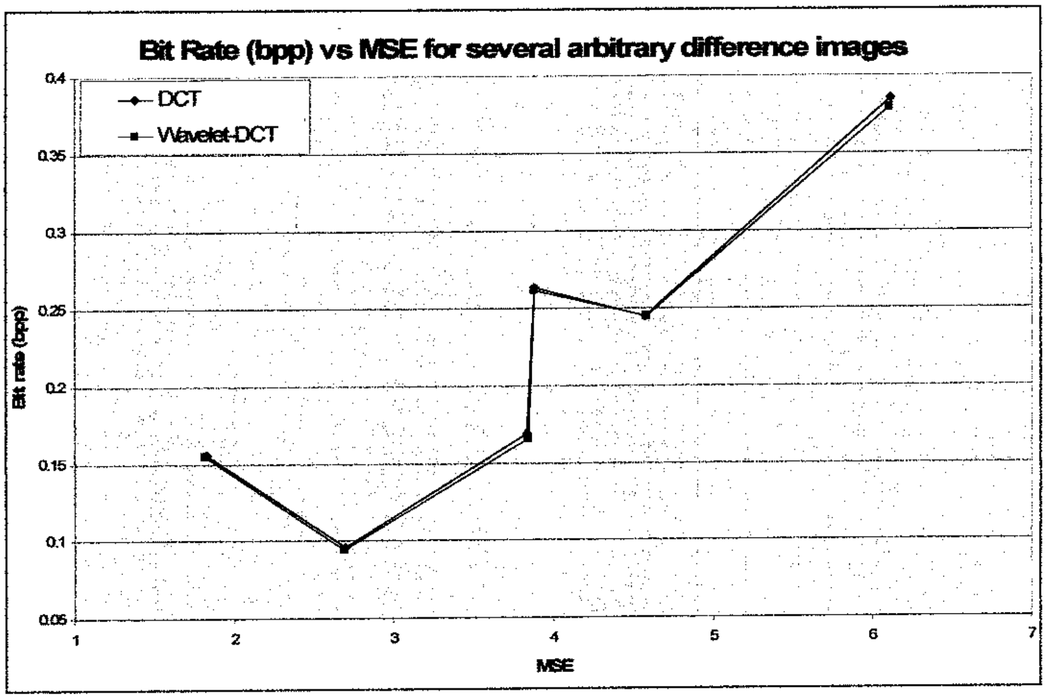


Figure 25 : Bit rate (bpp) versus MSE for several arbitrary difference images

Figure 26 illustrates the comparison of compression ratio for conventional DCT and Wavelet-DCT. The compression performances are much closed, but the pink line (Wavelet-DCT) is always superior in performance compared to blue line (DCT).

From the two graphs, it can be concluded that Wavelet-DCT has better performance over DCT in intra-frame coder to compress difference images, if an appropriate threshold level is applied to remove insignificant DWT coefficients. It is also discussed in section 2.1 , that most of the input of a video coding applications are difference images, where subtractions are performed on subsequent frames against its reference frames. Therefore, Wavelet-DCT is able to be utilized in such video coding applications.

Currently, DCT is still well-deployed in video coding applications for the intra-frame coder. With replacement of DCT by Wavelet-DCT, compression performance of the video coding applications will therefore improve, which directly contributes to data rate reduction for video streaming applications, decreases the network load and video file storage reduction.

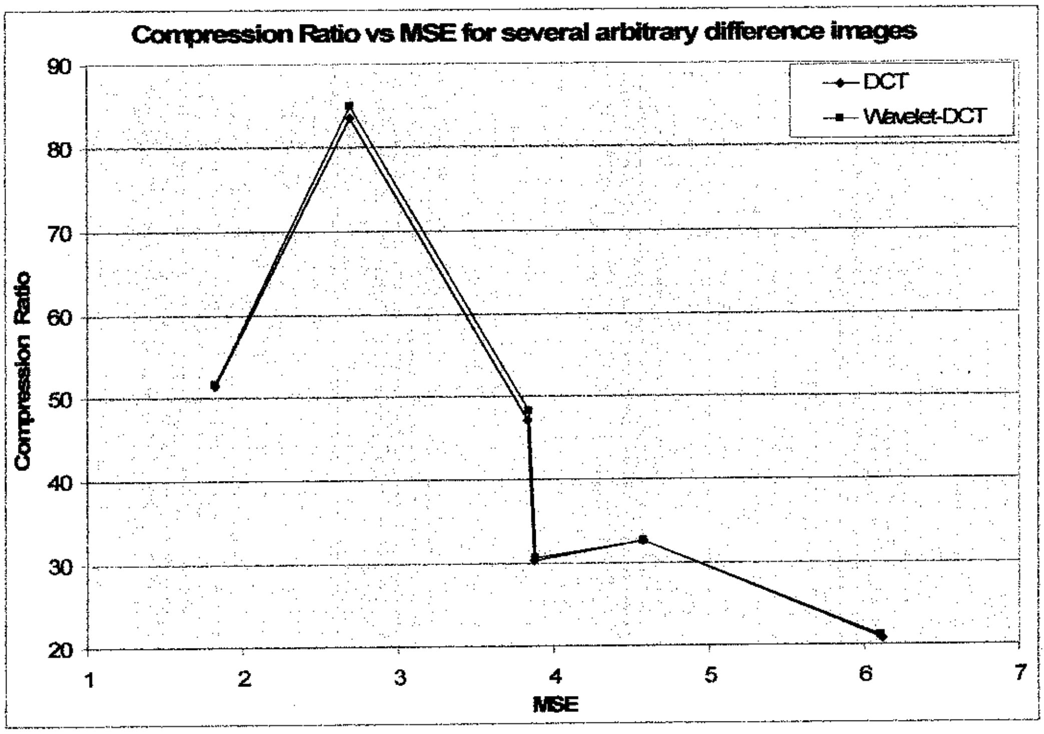


Figure 26 : Compression ratio versus MSE for several arbitrary difference images

In conclusion, the implementation of Wavelet-DCT based intra-frame coder for video coding application is considered successful. The Wavelet-DCT based intra-frame coder is able to achieve small additional compression with proper thresholding level. The algortihm might also reduce the MSE of the decoded image.

CHAPTER 5
CONCLUSION AND RECOMMENDATIONS

Chapter 5, the last chapter of the report is divided into two parts, which are conclusion and recommendations. The conclusion is a review of the project's objective followed by a brief discussion on the methodology of the simulation and the results that was achieved. The recommendations are suggestions for anyone who is interested to undertake in-depth study in this area of work. The idea given is for further development of the project.

5.1 Conclusion

A simulation model for an intra-frame coder has been developed by using SIMULINK. The intra-frame coders could be used for 8x8 and 16x16 block processing. All the internal blocks of the intra-frame coder are built on SIMULINK, except for the Entropy Coder, which has been developed using MATLAB.

Different types of images are used as test images for the simulation model, where the result shows that the intra-frame coder can achieve only low compression ratio for images with lots of details. However, it could compress images with fewer details at a higher compression ratio. For simulation with difference images, it shows that the compression ratio is very high, proving why inter-frame coder is used in a video coder application. The difference image is acquired by extracting several frames from a video file, and then performs subtraction on the frames, from its reference frame.

Then, a novel Wavelet-DCT algorithm using wavelet transform is proposed for implementation in the video coding applications, to replace DCT. The Wavelet-DCT code is developed using MATLAB, where its functionality has been tested and verified prior to its integration into the intra-frame coder.

The algorithm can perform thresholding to remove insignificant DWT coefficients, which improves the compression performance of an intra-frame coder, particularly for difference images. With an appropriate threshold value, bit rate of the encoded difference image can be reduced to increase the compression ratio.

With the use of Wavelet-DCT, the SIMULINK simulation result shows that there is compression performance improvement for difference image. Therefore, the Wavelet-DCT is useful for video coding applications, where its intra-frame coder's inputs are mostly difference images.

5.2 Recommendations

Due to time constraints and limited resources, the study on the implementation of Wavelet-DCT in the video coding applications is not entirely comprehensive. Therefore, the following suggestions are recommended for further development:

5.2.1 Wavelet Types

The project is conducted using only Haar Wavelet for the intra-frame coder applications. Daubechies-2 or other wavelets should be tried on the intra-frame coder. However, with other wavelet, the system will become more complicated and the computation complexity will increase. A study on the use of appropriate wavelets should be undertaken to identify the potential of this algorithm, without introducing excessive computational complexity.

5.2.2 Thresholding Scheme

The thresholding scheme used for the intra-frame coder is fixed level thresholding, which is a crude form hard thresholding. Other thresholding schemes should be explored like soft-thresholding suggested by [14] and wavelet thresholding [15]. A study should be undertaken to select the most suitable thresholding scheme.

5.2.3 Fast Wavelet Transform

Newer method which is more efficient in performing wavelet transform should be implemented. This efficient wavelet transform is also known as Fast Wavelet Transform, which is introduced by [16] As the name implies, it is faster because it relies on polynomial additions rather than multiplications. It can also be adapted to produce integer wavelet transform coefficients for the purpose of fixed-point processing. This could be a new area of research field to explore.

5.2.4 Full Video Coding Applications Test

The study has only constructed an intra-frame coder, which is not comprehensive for video coding application test. Therefore, it is suggested that the intra-frame coder should be integrated into a video coding application model, to fully test on its functionality and performance. The video coder model should be developed using SIMULINK, so that the existing work could be integrated easily.

5.2.5 Quantization Table

This study introduces a fixed-level quantization table for DTI images. This fixed-level quantization table is recommended for DCT coefficients of DTI images. However, it is suggested to replace the fixed-level quantization table with an adaptive quantization scheme, where appropriate quantization coefficients could be identified for the DTI images using Wavelet-DCT.

5.2.6 Intellectual Property (IP)

It is also recommended that the optimized Wavelet-DCT algorithm is converted into Verilog or VHDL for the purpose of fixed-point simulations. Then, Verilog or VHDL can be easily converted into FPGA for prototyping and patented as Intellectual Property (IP).

REFERENCES

[1] : H.264 - Web-based free-content multilingual encyclopedia
http://en.wikipedia.org/wiki/H.264, 12 September 2006

[2] : JPEG2000 - JPEG standard
http://www.jpeg.org/jpeg2000/, 13 March 2007

[3] : Michael D. Adams. *The JPEG-2000 Standard, Dept.* of Elec. and Comp. Engineering, University of British Columbia

[4] : Tamal Bose (2004) *Digital Signal and Image Processing,* John Wiley & Sons

[5] : D.L. Donoho. *Unconditional bases are optimal bases for data compression and for statistical estimation.* Applied and Computational harmonic Analysis, Vol.1, No. 1, pp 100-115, December 1993

[6] : How Sun Dee, *Wavelet-Based Block Transforms and Their Applications,* Master thesis, Electrical and Electronics Engineering Department, Universiti Sains Malaysia, Malaysia, January 2001

[7] : Fossel, S., Fottinger, G. Mohr, J. *Motion JPEG2000 for High Quality Video Systems,* Fraunhofer Soc. For Appl. Res., Fraunhofer Inst. for Integrated Circuits IIS, Erlangen, Germany, IEEE Transactions on Consumer Electronics, Nov 2003

[8] : Takeshi Mogi. *A Hybrid Compression Method based on Region Separation for Synthetic and Natural Compound Images,* W OEle ctric Co. Ltd., Hypermedia Research Center, 10/24/1999

[9] : Peter Symes (2001) *Digital Video Compression,* the McGraw-Hill Companies

[10] : Haitao Guo. *Wavelet for Approximate Fourier Transform and data Compression, Ph.D.* Thesis, Rice University, Houston Texas, May 2007

[11] : Andre Ooi Ken Lee, *Implementation of Fast Approximate Fourier Transform and Its Application,* Bachelor Thesis, Electrical and Electronics Engineering Department, Universiti Teknologi PETRONAS, Malaysia, June 2006

[12] : Karl Skretting. *Arithmetic Coding and Huffman Coding in MATLAB,* Cybernetics Group at University of Stavanger, Last update: Oct. 11. 2005. http://www.ux.uis.no/~karlsk/proj99/, 06 April 2007

[13] : Standard Test Image - Web-based free-content multilingual encyclopedia http://en.wikipedia.org/wiki/Standard_test_image, 08 April 2007

[14] : D.L. Donoho, *De-noising by Soft-Thresholding,* IEEE Transaction on Information Theory, Vol. 41, may 1993.

[15] : S. Mallat, *A Wavelet Tour of Signal Processing,* New York Academic, 1999

[16] : I. Daubechies, W. Sweldens, *"Factoring the Wavelet Transform into Lifting Steps"*, Technical Report, Princeton and Lucent Technologies, Revised: November 1997.

[17] : Dave Marshall. *The Discrete Cosines Transform,* Latest Update: April 2001 http://www.cs.cf.ac.uk/Dave/Multimedia/node231.html, 15 April 2007

[18] : Tat Wei, Ho, Implementation of The Fast Approximate Fourier Transform via Discrete Wavelet Transform on the TI TMS320C5402", Digital Signal Processing Project, Universiti Teknologi PETRONAS, Tronoh Perak, Malaysia, Now 2002

APPENDICES

APPENDIX A
EXAMPLE: DCT COEFFICIENTS OF AN IMAGE

86	2	-3	6	-2	2	-2	2
-247	-4	-5	-3	0	-3	1	1
-117	-1	1	-1	-1	-1	-1	0
-40	-2	2	1	2	1	-2	0
-7	-2	-1	1	0	-1	-1	2
-6	1	0	0	0	0	-2	-1
-4	-1	-1	1	-2	-1	-1	-1
-3	-3	-1	1	0	1	1	1

Figure 27 : The DCT coefficients of an image (example from [9])

APPENDIX B

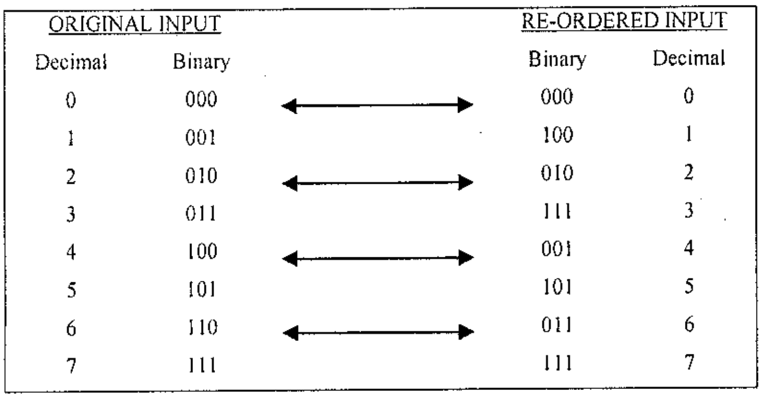
EXAMPLE: QUANTIZED DCT COEFFICIENTS OF AN IMAGE

5	2	0	0	0	0	0	0
-21	0	0	0	0	0	0	0
-8	0	0	0	0	0	0	0
-3	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Figure 28 : The quantized DCT coefficients of an image (example from [9])

APPENDIX C

EVEN-ODD SEPARATION PROCESS IN FFT



ORIGINAL INPUT			RE-ORDERED INPUT	
Decimal	Binary		Binary	Decimal
0	000	↔	000	0
1	001		100	1
2	010	↔	010	2
3	011		111	3
4	100	↔	001	4
5	101		101	5
6	110	↔	011	6
7	111		111	7

Figure 29 : Even-odd separation by shuffling input bits

APPENDIX D
MATLAB CODE FOR WAVELET-DCT

fwdDWTN.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Author Ho Tatt Wei (Rev: Chai Beng Seow
%Script to compute the Wavelet-DCT of an input image using DWT
%After the work of H.Guo (Doctoral Thesis
%J level wavelet transform using any arbitrary wavelet. (NOT wavelet
packets meaning DWT is iterated on the scaling coefficients)
%WITH PERIODIC CONVOLUTION to avoid expansiveness through FIR
subband filtering
%Weakness of PERIODIC convolution is that the signal length must be
divisible by 2^J due to downsample/upsample process
%Combine downsampling and upsampling together in DWT matrices
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function DWTcoeffsN=fwdDWTN(x,wavelet,J)
[row col] = size(x);
if row ==1 |col ==1
    if col>1
        x=x';
    end
end

[h0 h1 g0 g1]=wfilters(wavelet);
limitH = length(h0);
N = length(x);
if N < limitH
    DWTcoeffsN=x;
    return
end

if mod(N,2^J)>0
    remainder = rem(N,2^J);
    L=2^J-remainder;
    L=N+L;
    x(L)=0;
    N=length(x);
end;
%x=wshift('1',x,limitH-1);
h0mtx = convmtx(fliplr(h0),N);
ppdh0 = h0mtx(1:end,end - limitH+2:end);
h0mtx = h0mtx(1:end,1:end-limitH+1);
h0mtx(1:end,1:limitH-1) = h0mtx(1:end,1:limitH-1)+ppdh0;

h1mtx = convmtx(fliplr(h1),N);
ppdh1 = h1mtx(1:end,end - limitH+2:end);
h1mtx = h1mtx(1:end,1:end-limitH+1);
h1mtx(1:end,1:limitH-1) = h1mtx(1:end,1:limitH-1)+ppdh1;

DSMPLmtx = eye(N);
DSMPLmtx = DSMPLmtx(1:2:end, 1:end);

DWTmtx = [DSMPLmtx*h0mtx;DSMPLmtx*h1mtx];
```

```
DWTcoeffsN = DWTmtx*x;
if J>1
input = DWTcoeffsN(1:ceil(N/2));
DWTcoeffsN(1:ceil(N/2))= fwdDWTN(input,wavelet,J-1);
End

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%End of forward DWT

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

waveletDCT.m

```
function [Xk] = waveletDCT(x,wavelet,threshold)
maxlevels = 3;
N=length(x);
J=1;
[row col] = size(x);
if row==1|col ==1
    if col>1
        x=x';
    end
end

if(threshold ==1)
    for k=1:16
           if abs(cDWT(k))<1
            cDWT(k)=0;
        end
    end
end

w= cDWT(ceil(N/(2^J))+1:ceil(N/(2^(J-1))));
level=1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%New code programmed by Chai Beng Seow starts here
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


n=N;
for cnt = 1:N/2
p(cnt)=(N/2)-cnt+1;
end

col = (0:n/2-1);
   A = diag(cos(pi *col / (2*n)));
    C = -diag(sin(pi *col / (2*n)));
    C=C(p,:);
row = (n/2+1:n);
    B = diag(cos(pi *row / (2*n)));
    B = B(p,:);
    D = diag(sin(pi *row / (2*n)));
Bn(1:n,1:n)=0;
Bn(1:n/2,1:n/2)=A;
Bn(n/2+1,n/2+1)=D(end,end);
Bn(n/2+2:n,n/2+2:n) = D(1:end-1,1:end-1);
Bn(2:n/2,n/2+2:n)=B(2:end,1:end-1);
Bn(n/2+2:n,2:n/2)=C(1:end-1,2:end);

Tfactor=Bn;

%The illustration of the hardcoded modified equivalent butterfly
%coefficients
%Tfactor = [1      0       0       0       0       0       0       0;
 %        0       0.9808  0       0       0       0       0
0.1951;
  %       0       0       0.9239  0       0       0       0.3827  0;
   %      0       0       0       0.8315  0       0.5556  0       0;
    %     0       0       0       0       1       0       0       0;
     %    0       0       0       -0.5556 0       0.8315  0       0;
      %   0       0       -0.3827 0       0       0       0.9239  0;
       %  0       -0.1951 0       0       0       0       0
0.9808];
%the coeffients are obtained from the code

n=N/(2^J);
[cc,rr] = meshgrid(0:n-1);
c = sqrt(2 / n) * cos(pi * (2*cc + 1) .* rr / (2 * n));
c(1,:) = c(1,:) / sqrt(2);
W2=c;
W=W2;

for count = 1:2^J-1
    indexBeg = (count*(N/(2^J)))+1;
    indexEnd = (count+1)*N/(2^J);
    W(indexBeg:indexEnd,indexBeg:indexEnd)=W2;
end

if J>1
    J=J-1;
    n=N/(2^J);
[cc,rr] = meshgrid(0:n-1);
c = sqrt(2 / n) * cos(pi * (2*cc + 1) .* rr / (2 * n));
c(1,:) = c(1,:) / sqrt(2);
W3=c;
    for count = 1:1
        first_index = (count*(N/(2^J)));
        W(1:first_index,1:first_index)=W3;
        indexBeg = (count*(N/(2^J)))+1;
        indexEnd = (count+1)*N/(2^J);
        W(indexBeg:indexEnd,indexBeg:indexEnd)=W3;
    end
end

if J>1
    J=J-1;
    n=N/(2^J);
[cc,rr] = meshgrid(0:n-1);

c = sqrt(2 / n) * cos(pi * (2*cc + 1) .* rr / (2 * n));
c(1,:) = c(1,:) / sqrt(2);
W4=c;
    for count = 1:1
        indexBeg = (count*(N/(2^J)))+1;
        indexEnd = (count+1)*N/(2^J);
        W(indexBeg:indexEnd,indexBeg:indexEnd)=W4;
    end
end
```

```
Xk = 1;
J=level;

for count=1:J
       Xk = Tfactor;
end
Iby2=eye(N);
for a = 1:N
    if a > (N/2)&& mod(a,2)==0
        Iby2(a,a)=-1;
    end
end

Xk=Xk*W*Iby2*cDWT;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% End of WaveletDCT.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

APPENDIX E

MATLAB CODE FOR DIFFERENCE IMAGE ACQUISITION

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This MATLAB code is used to acquire difference image
%'input video' is the video file where the difference image is
obtained form
%for this particular example, frame 501 and 500 are obtained
%The two frames are then subtracted to obtain the difference image
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
frame500=aviread('input_video',500);
frame501=aviread('input_video',501);
gray500=rgb2gray(frame500.cdata);
gray501=rgb2gray(frame501.cdata);
diff=gray501-gray500;
imshow(diff);
```

APPENDIX F
AN EXAMPLE FOR HUFFMAN CODING APPLICATION

A 2-bits/pixel image is 10 rows by 10 columns.

Step 1: Find the gray level probabilities for the image by finding the histogram.
The image's histogram is given in Figure 30 . It is then converted into probabilities with g_0 = 20/100 =0.2; g_1= 30/100 =0.3; g_2= 10/100 =0.1; g_3= 40/100 =0.4

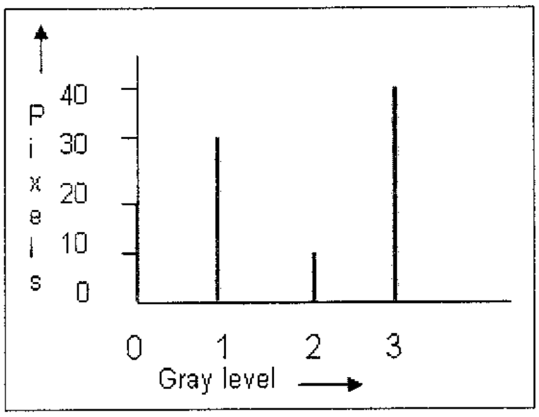


Figure 30 : Histogram of a 2-bits/pixel images

Step 2: Order the input probabilities (histogram magnitudes) from smallest to largest.

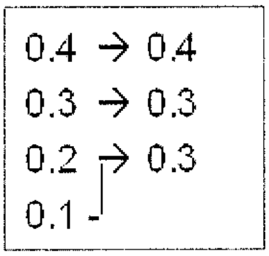
g3 = 0.4

g1 = 0.3

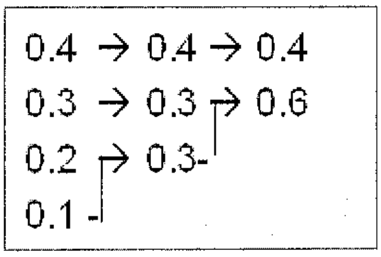
g0 = 0.2

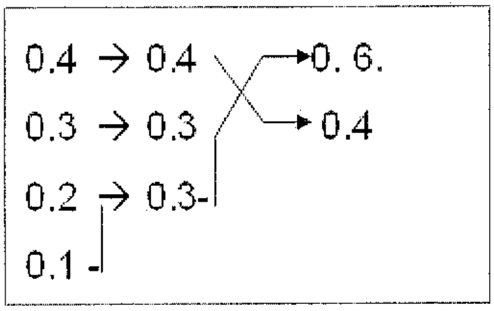
g2 =0.1

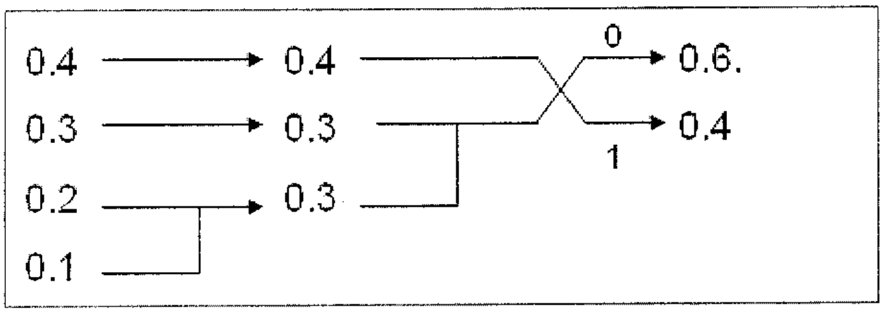
Step 3: Combine the smallest two by addition until only two values remain.

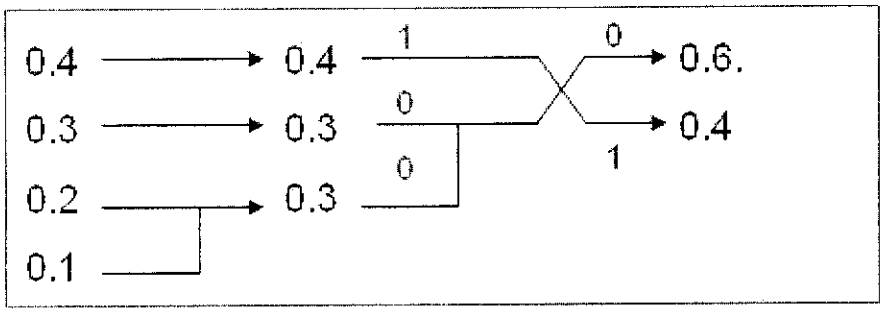


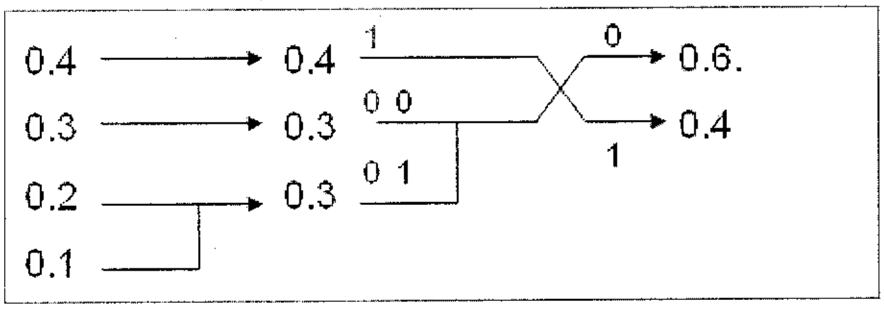
Step 4: Go to step 2, until only two probabilities are left.

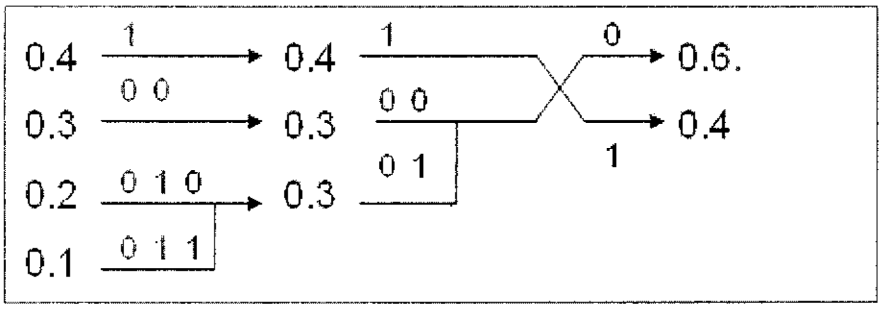












Step 5:By working backward along the tree, generate code alternating assignment of 0 and 1.

	Huffman code
g_0 =0.2 ;	010
g_1 = 0.3 ;	00
g_2 = 0.1 ;	010
g_3 =0.4 ;	1

Average Bits = 9/4 = 2.25 = 2 bits/pixel (A)

Average length in Huffman code

= Σi=0-3 li pi = 3 (0.20+ 2 (0.3) + 3 (0.1) + 1 (0.4)

= 1.9 bits/pixel (B)

Compression Ratio = A: B = 2:1.9 = 1.05