INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Université d'Ottawa · University of Ottawa

FINITE WORDLENGTH EFFECTS IN DISCRETE TIME WAVELET TRANSFORM

by

Kenny Chan, B.A.Sc., P.Eng.

A thesis submitted to the
School of Graduate Studies and Research
in Partial fulfillment of the requirements for the degree of

Master of Applied Science
in Electrical Engineering

School of Information Technology and Engineering
University of Ottawa

July, 1998

ABSTRACT

This thesis investigates the finite wordlength effects in one-dimensional discrete-time wavelet transform (DTWT). A MATLAB DTWT model is written based on an efficient algorithm suggested by the Multiresolution Analysis (MRA). This model allows users to investigate the overall non-ideal effects of a MRA-based DTWT system by specifying round-off method for computation, the number of quantization bits employed in representing filter coefficient and internal computational results. Further, the possibility of discarding the high frequency portion of a signal in DTWT signal reconstruction phase (Selective Subband Reconstruction) is shown practical with the presence of finite wordlength effects. The feasibility of using either direct structure filter or lattice structure filter to realize a MRA-based DTWT system is also considered in this thesis. Simulation results show that the lattice structure filters have superior magnitude and phase responses in most scenarios under investigation to its direct structure-based counterparts. Moreover, lattice structure filter demands relatively less computation than direct structure filter in the same filter length and DTWT configuration. In conclusion, the lattice structure filter is a better alternative than the direct structure filter for MRA-based DTWT system implementation.

ACKNOWLEDGEMENT

I would like to express my gratitude to my supervisor, Prof. Tet Yeap for his outstanding guidance, advice and support throughout this work. I am also deeply indebted for my managers, Mr. Waichi Lo and Mr. Luc Lanoue, for allowing me to have flexible work hours while I was working at Nortel from September 1994 to March 1997. Without their understanding and assistance, it would be impossible to complete my degree requirements on a part-time basis and to work at Nortel at the same time.

The financial support of Nortel via employee tuition reimbursement and of Prof. Tet Yeap are gratefully acknowledged. I would also like to express my thanks to Mr. Albert Lee and Mr. Erick Wong, P.Eng., my University of Toronto MBA classmates, for their reviews and comment regarding this thesis. Special thanks are extended to many of my friends and colleagues for offering enjoyable discussions, encouragement and humorous discourse throughout this project. Last, but definitely not least, my warm thanks go to my family members for their unlimited support and many others who helped me to achieve my goal.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

The history of wavelets can be dated back to the discovery of Harr series in 1910. However, wavelet theory began to receive a lot of attention after Grossmann and Morlet [1] published the concept of wavelet transformations based on their work in seismic data analysis in 1984. The result triggered subsequent works on the mathematical foundation of wavelet theory. One of the important contributions was the introduction of compactly supported orthonormal wavelets by Ingrid Daubechies. Her paper [2] contains the first families of orthonormal wavelet bases that have practical implication in a remarkably wide variety of fields. Further development of such bases in the context of multiresolution signal analysis was formalized by Meyer [3] and Mallat [4]. Until now, wavelets remain to be an active research area and it is rapidly finding application in many disciplines. Just to name a few:

- Turbulent blood flow analysis for coronary artery disease detection; speech extraction from background noise in digital hearing aids; noise reduction and bringing out important diagnostic features in mammograms [5].

- Fractal modulation [6]; and discrete wavelet multitone (DWMT) technique [7] in asymmetric digital subscriber line (ADSL) system for high-speed voice/data transmission over unreliable communication channels.

- Multigrid methods in the solution of differential equation [8]; Mathematical tools for statistics [9].

- Pyramidal image representation in image processing and computer vision [10].

- Wavelet-based image and video compression schemes usually introduce less "artifacts" with higher compression ratio when comparing to JPEG and MPEG algorithms which are based on discrete cosine transform (DCT) [11].

1.1 What are Wavelets?

Rapid and continuous growth in wavelet has made the task of finding a unanimously accepted wavelet definition almost impossible. In view of this, we delineate wavelets from their physical properties. In general, a wavelet is an oscillating waveform that persists for only one or few cycles. Space locality of a wavelet function implies that the majority of the energy is restricted to a finite interval. A wavelet function which is zero outside the finite interval is known as "compactly supported". An example of Daubechies D_{10} wavelet is shown in Figure 1-1, along with its frequency responses in Figure 1-. Wavelets come in different sizes and shapes. Haar wavelets, Shannon wavelets, Daubechies wavelets and Butterworth wavelets are examples [12] in the growing wavelets collection.

Wavelets come in various forms: (**Classical wavelets**) using dyadic translates and dilates of only one prototype function for wavelet construction [2]; (**Wavelet Packets**) using more complex transform to yield wavelet functions with better frequency localization [13]; (**Multiwavelets**) applying more than one prototype function to construct wavelet in order to give results that classical wavelets are incapable of [14]; (**Second Generation Wavelets**) constructing wavelets with ideas other than translation and dilation of prototype function (e.g., The Lifting Scheme [15]). These items are by no mean an exhaustive and complete listing of all available wavelet formats. In fact, they are included here to illustrate the vast

diversity in current research activities. From now on, all discussions are confined to classical wavelets unless specified otherwise.

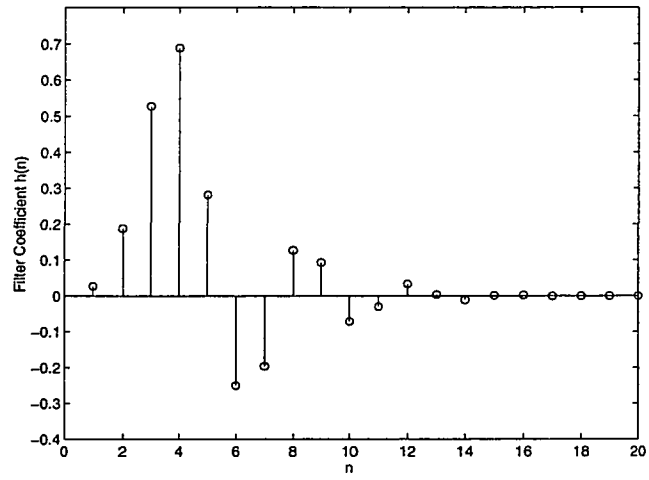


Figure 1-1 A compactly supported orthonormal wavelet: Daubachies D_{10}

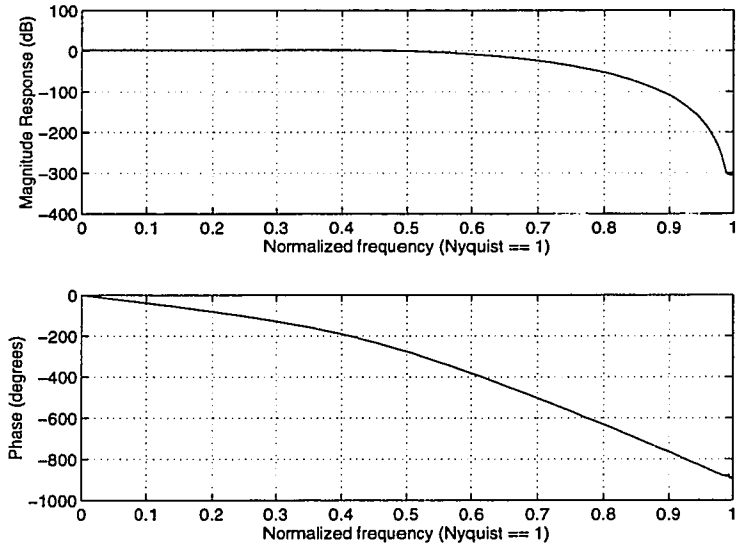


Figure 1- 2 Magnitude and Phase responses for Daubachies D_{10}

1.1.1 Wavelet Transform

Similar to the concept of Fourier transform, wavelets can be used as the building block for general functions. In Fourier transform, a signal is represented as a superposition of sinusoids with different frequencies. The Fourier coefficients record the contribution of the sinusoids at various frequencies. In wavelet transform, a signal is represented as a sum of wavelets with different locations (positions) and scales (durations). For classical wavelets, only one prototype (mother) wavelet is used in the transformation. The different locations and scales are obtained by expansion, contraction, and shifting of the prototype wavelet. The wavelet coefficients measure the contribution of the wavelets at various locations and scales.

As explained in Appendix A, the wavelet transform constitutes as natural a tool for the manipulation of transient signals (e.g. non-stationary signals) as the Fourier transform does for translation invariant signals (e.g. stationary and periodic signals).

1.1.2 Wavelets in Digital Signal Processing

One of the greatest contributions of wavelet theory is the introduction of a common framework to relate various established methodologies in numerous fields. In the DSP domain, the Mallat Pyramid algorithm [4] that calculates fast discrete wavelet transform can be viewed as a special case of subband coding. Subband coding is a digital DSP application originally developed for data compression. By using filter banks, a digital signal is separated into numerous frequency bands which are approximated by filter coefficients. The developments of subband coding and filter banks have come from Smith and Barnwell [16], Vaidyanathan [17] and Vetterli [18]. From an implementation viewpoint, quadrature mirror-image filter (QMF) banks [19, 20] and cosine-

modulated filter banks [21-23] have received considerable interest because of their optimum performances in near-perfect signal reconstruction as well as aliasing error, amplitude distortion and phase distortion reduction. Detailed connections among wavelets, subband coding and filter banks are explored in [11, 24, 25].

Although both wavelet-based signal analysis and subband coding share a common algorithm and underlying theory, their design requirements are quite different. For wavelet analysis, emphasis is put on the choice of wavelets' mathematical properties such as smoothness and localization. These properties have direct impact in interpreting the resulting wavelet coefficients. Above all, wavelet-based signal analysis can help us to explore a wide range of signals or images that other Fourier-based analysis schemes do not perform well (Refer to A.2). For subband coding, emphasis is focused in the choice of filter family that ensures high compression ratio, fast computation speed and good signal reconstruction capability.

1.1.3 Hardware Implementation of Discrete Wavelet Transform (DWT)

Recently there has been a flurry of activities around the world focusing on the development of VLSI and chip implementation of discrete wavelet transform (DWT). Several VLSI architectures, which are based on the Mallat Pyramid algorithm, have been proposed thus far (Refer to 3.1.1). However, the majority of proposals only focuses on theoretical issues such as computation complexity and achievable speed, and little attention is paid to the wordlength requirements that are at the heart of an economical implementation. DWT system designed under idealizing assumptions (e.g., floating-point arithmetic) may yield undesired performance when implemented in fixed-point arithmetic if finite wordlength effects are not compensated. In the worst case, these non-ideal effects can even

bring a system into an unstable state. Therefore, they must be considered carefully during the design and implementation phase.

1.2 Project Goals

Most often, it is not economical to model in detail or to calculate analytically the non-ideal effects of a system. For a complex system like wavelet transform, this project employs computer simulation to study the overall non-ideal effects of Discrete Time Wavelet Transform (DTWT). The primary objective of this project is to develop a versatile MATLAB model of DTWT. The model should facilitate researchers to prototype DTWT based on the Multi-Resolution Analysis (MRA) (to be reviewed in 2.3), allowing researchers to quickly specify the configuration of a DTWT system with the flexibility to apply the number of wordlength in the representation of filter coefficients and internal results. The model also allows users to adopt different rounding methods (e.g., no quantization, rounding or truncation) in order to simulate the non-ideal effects.

In an analog to digital conversion for input signal, the signal can only be implemented with finite wordlength. This quantized signal introduces noise into the system. In addition, the quantization effect due to finite wordlength also happens to the filter coefficients. Another non-ideal effect will also affect the character of the system: arithmetic operations involving finite wordlength numbers usually yield results of larger wordlength. The process to reduce the wordlength is a nonlinear operation. The resulting nonlinear system might behave completely different from the original linear design. It is obvious that using longer wordlength for data and filter coefficients can reduce the quantization errors. Therefore, it is a vital objective of the design process to determine the optimum register wordlength for both sufficient performance and acceptable error for DTWT VLSI architecture.

DTWT can be implemented using cascade and series of digital filters (refer to discussion in section 2.4.1). The configuration and definition of specific filter characteristics is given in Chapter 2. Since digital filter is the building block of DTWT, the study of non-ideal effects of two possible digital filter implementation, namely the direct structure and the lattice structure is important. In this project, the non-ideal effects due to quantization effect in filter coefficient and internal computation under various wordlengths and rounding methods are simulated. Armed with the above information, system designers can better understand the non-ideal effects of the MRA-based DTWT. ASIC designers can use the simulation results and the MATLAB DTWT model developed in this project to help choosing the optimum hardware at various parts of the system without sacrificing overall performance, while at the same time reducing the size, cost and power consumption of their resulting products.

1.3 Organization of the Thesis

Chapter 2 gives a brief outline of the wavelet theory with emphasis on MRA-based discrete wavelet transform implementation. The first part of Chapter 3 is devoted to a literature research on previous work done on VLSI implementation of discrete wavelet transform and finite wordlength analysis. The rest of the chapter focuses on defining implementation criteria for our DTWT MATLAB model and the simulation results for the direct structure filter. In Chapter 4, the lattice filter coefficients for Daubechies wavelet family are derived. Then, the simulations performed on the direct structure are repeated for the latter structure, and the results are compared and analyzed. Finally, Chapter 5 presents the conclusions and suggests future directions to pursue.

In Appendix A, a review on traditional transformations is given and the shortcomings of short-time Fourier transform are also mentioned in order to

contrast the advantages of discrete wavelet transform in certain tasks. All MATLAB functions and programs developed for this project are listed in Appendix B. Finally, simulation data for both direct and lattice digital filter implementations covered in Chapters 3 and 4 are tabulated in Appendix C.

Chapter 2

THE WAVELET TRANSFORM

Wavelet transform (WT) can be categorized into two basic forms. They are namely the continuous wavelet transform (CWT) and discrete wavelet transform (DWT). The former operates on continuously indexed functions $f(t), t \in \mathbf{R}$ whereas the latter operates on sequences $f[n], n \in \mathbf{Z}$. The objectives of this chapter are to introduce both CWT and DWT, and their relationships with various signal processing concepts, such as Multiresolution Analysis (MRA) and filter banks. Finally, this chapter provides a brief introduction on multidimensional wavelet transform.

The research into WT was largely motivated by the drawbacks of traditional transforms such as Fourier transform. Analogous with Fourier transform, WT is a linear transform that converts input signal into different frequency components. However, one of the most distinguishing features of WT is that it possesses better time/frequency localization properties which cannot be achieved by any Fourier-based transforms. An epitome of Fourier-based transforms, including discrete Fourier transform, discrete cosine transform and Windowed Fourier transforms, can be found in A.2. As we will show in this chapter and Appendix A, WT not only offers an alternative to visualize or process signals, but also provides an extra degree of freedom for system design and implementation.

2.1 Continuous Wavelet Transform

The CWT maps a continuously single-indexed function into another function that is continuously indexed by two variables: scale and location. The CWT is defined as [11, 17, 26]:

$$W_{a,b}(f) = \langle \psi_{a,b}(t), f(t) \rangle = |a|^{-1/2} \int_{-\infty}^{\infty} f(t) \psi\left(\frac{t-b}{a}\right) dt; \quad a,b \in \mathbf{R}\ (a \neq 0), \quad (2.1.1)$$

where the function ψ is commonly referred as the mother wavelet. The wavelets family $\psi_{a,b}(t)$ in equation (2.1.1) is the dilated and translated version of the mother wavelet ψ. Equation (2.1.1) measures the similarity between a signal $f(t)$ and the shifts/scales of the mother wavelet. In other words, $W_{a,b}(f)$ represents the correlation of $f(t)$ with ψ scaled by a and shifted by b. The factor $|a|^{-1/2}$ is used to conserve the norm. In addition, the integral in equation (2.1.1) suggests another way of interpreting the CWT. That is, the integral can be viewed as the output of a bandpass filter of impulse response $|a|^{-1/2} \psi(\frac{-t}{a})$, at location b given the input signal $f(t)$. This observation will become evident when we consider the implementation of wavelet transform in later sections.

WT is not limited to one set of basis function like short-time Fourier transform (STFT) which utilizes just the sine or cosine functions. In fact, the inclusion of time and scale parameters in wavelet transform provides families of basis that generalize and enrich existing methods[1] for partitioning of a signal's energy content. Due to the tradeoff nature of these two parameters, the frequency localization afforded by WT can be increased with increasing scale at the expense of time localization or vice versa. As illustrated in Figure 2-1, the wavelet window

[1] STFT or Gabor-based analysis has only one variable - time.

$\psi_{a,b}(t)$ is long at low frequencies (i.e., large a) that lead to high frequency resolution (but low time resolution). Consequently, the overall or global behavior of the signal is detected. Conversely, the window $\psi_{a,b}(t)$ is short at high frequencies (i.e., small a) that lead to high spatial resolution (but low frequency resolution). Hence, the detailed or non-stationary behavior of the signal can be captured.

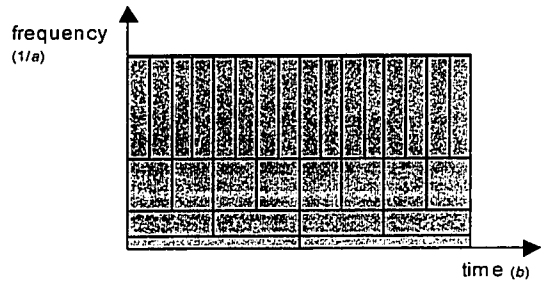


Figure 2-1 Time-frequency tilings for wavelet transform. Notice that the time-frequency plane does not have the uniform width as in the case for STFT (Please refer to Figure A-1).

2.1.1 The Inverse Continuous Wavelet Transform

In this section, the invertibility of the CWT is investigated. Although further restrictions can be imposed on the mother wavelet ψ to ensure functionality such as orthogonality, the only requirement for the transform to be invertible on its range is that [11, 24, 26, 27]

$$C_\psi = \int_{-\infty}^{\infty} \frac{|\Psi(\omega)|^2}{|\omega|} d\omega < \infty, \qquad (2.1.1.1)$$

where $\Psi(\omega)$ is the Fourier transform of the mother wavelet $\psi(t)$. If the *admissibility* condition specified in (2.1.1.1) is satisfied, the following inversion formula holds on the range of the transform

$$f(t) = \frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} W_{a,b}(f) \psi_{a,b}(t) \frac{dadb}{a^2}. \qquad (2.1.1.2)$$

In practice, we are interested in a mother wavelet that decays sufficiently fast (i.e., compactly supported). In that case, the admissibility condition reduces to the requirement that $\Psi(0) = 0$ [26]. Equation (2.1.1.1) also implies that $|\Psi(\omega)| = 0$ as $\omega \to \pm\infty$. Since the Fourier transform Ψ decays at high frequencies and remains zero at origin, the mother wavelet exhibits *bandpass* behavior.

Because the scaling and time parameters in equation (2.1.1) are continuous, the CWT and its inverse shown respectively in (2.1.1) and (2.1.1.2) are indeed *oversampled.* Oversampling is a useful property in reducing the reconstruction error. It is because the reconstruction mean square error due to noise can be reduced by a factor equivalent to the oversampling ratio [28, 29]. The definition for oversampling ratio will be giving in the next section after we have introduced the concept of *frame.* Nonetheless, the CWT has minor practical value in real-time applications due to its numerous (*uncountably infinite* to be exact) computational steps. In the next section, we explain that it is not necessary to perform WT on all possible values of a and b in order to achieve complete signal representation and invertibility.

2.2 Discrete Wavelet Transform and Frames of Wavelets

Even though a signal has compact support in the (a, b) plane, all values of the CWT in that region are still required to fully represent the signal. Here, the discrete wavelet transform (DWT) that maps $L^2(\mathbf{R})$ to $l^2(\mathbf{Z}^2)$ is introduced. That is, the CWT defined in (2.1.1.2) can be computed on a discrete grid of points $(a_n, b_n)_{n\in Z}$.

The scaling parameter is first discretized to be $a = a_0^m$, with $m \in Z$ and $a_0 \neq 1$. Then, b is discretized by taking integer multiples of b_0 ($b_0 > 0$). The time the parameter b is chosen in such a way to cover the whole time axis $\psi(t - nb_0)$. Since the basis functions are rescaled, b must be dependent of m and is therefore defined to be $b = nb_0a_0^m$. Thus, the discretized wavelet functions is defined as

$$\psi_{m,n}(t) = a_0^{-m/2}\psi(a_0^{-m}t - nb_0), \quad (m,n) \in \mathbf{Z}^2 \tag{2.2.1}$$

where $a = a_0^m$, $b = nb_0a_0^m$, $(m,n) \in \mathbf{Z}^2$, $a_0 > 1$, *and* $b_0 > 0$. Subsequently, the DWT equation is obtained by substituting (2.2.1) back into (2.1.1):

$$W_{m,n}(f) = \langle \psi_{m,n}(t), f(t) \rangle = a_0^{-m/2} \int_{-\infty}^{\infty} f(t)\psi(a_0^{-m}t - nb_0)dt, \quad (m,n) \in \mathbf{Z}^2. \tag{2.2.2}$$

In this definition, the signal f is being scaled and translated in a discrete manner rather than a continuous one as in CWT. The discretization of (2.2.2) is illustrated in Figure 2-2.

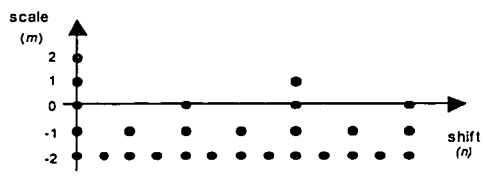


Figure 2-2 Sampling Grid for DWT (the case shown is $a_0 = 2^{1/2}, b_0 = 1$.)

Before we could show that the sequence $\langle \psi_{n,m}, f \rangle_{n,m \in Z}$ in (2.2.2) can completely characterize the signal f and that f can be reconstructed from this sequence in a numerically stable manner, we need to introduce the concept of *frame* first.

A *frame* is a family of functions $\{x_k\}$ in a Hilbert space[2] H if for all f in H there exist $A\|f\|^2 \le \sum_k |\langle x_k, f \rangle|^2 \le B\|f\|^2$ $(0 < A \le B < \infty)$. A and B are called *frame bounds*. A frame is known as a *tight* frame if A and B are equal. The oversampling ratio or redundancy ratio equals A if $\|x_k\| = 1$ in the case of tight frame. If the ratio is one (i.e., $A = B = 1$), $\{x_k\}$ constitute an orthonormal basis [28]. We now return to the particular of the wavelet function. In Appendix A.1, we discuss the dual and elementary bases. These bases coincide when they are orthonormal (i.e., $\{\varphi_i\} = \{\tilde{\varphi}_i\}$). Hence, this property offers an efficient way to recover the original signal f from $\langle \psi_{m,n}, f \rangle$. Substitute (A.1.2) or (A.1.3) into (A.1.1), we have

$$f = \sum_{m,n \in Z} \langle \varphi_{m,n}, f \rangle \tilde{\varphi}_{m,n} \,. \tag{2.2.3}$$

---

[2] A complete inner product space is called Hibert space or L_2-space. Hibert spaces contain a countable orthonormal basis if and only if it is separable. Details for separability can be referred to [30].

For an orthonormal basis, Daubechies mentioned in [28] that automatically $\tilde{\psi}_{m,n} = A^{-1}\psi_{m,n}$ and A equals 1. Then, equation (2.2.3) becomes

$$f = \sum_{m,n\in Z} \langle \psi_{m,n}, f \rangle \psi_{m,n} . \tag{2.2.4}$$

Moreover, Daubechies also showed that the inner product $\langle \psi_{m,n}, f \rangle$ for an orthonormal basis is unique. Therefore, they are adequate to uniquely represent the original signal if an orthonormal basis (mother wavelet) is utilized.

Although equation (2.2.4) is redundant in general, it is possible to remove the redundancy by adding more restrictions on the sampling parameters as well as the choice of mother wavelet. In [29] and [25], Cohen and Vetterli demonstrated that the DWT equivalent to oversampling the CWT on a very dense grid when $a_0 = 1$ and $b_0 = 0$. In fact, we are going to study a particular set of DWT parameters ($\psi_{n,m}$, a_0 and b_0) that allow critically sampling of CWT. The removal of redundancy reduces computation that eventually leads to an efficient algorithm now known as fast wavelet transform (FWT).

Since our major concern is the digital implementation of DWT, focus will be placed on the discrete time wavelet transform (DTWT) which is the equivalent to the DWT for sequences. In short, DTWT and DWT are algorithmically equivalent.

2.3 Multiresolution Analysis

We now take a different angle to view wavelet transform through the lens of Multiresolution Analysis (MRA) that provides another piece of the puzzle towards practical implementation of FWT. Mallat introduced MRA in [4], an image decomposition concept that represents the image as a hierarchy of

resolution levels. In general, the original state of an input image is considered to be at its highest resolution. To obtain a lower approximation of the image, it is necessary to filter the image with a lowpass filter. If a halfband filter is used in the lowpass operation, the resultant approximation will have half of the original bandwidth. Thus, in accordance with Nyquist's principle, the sampling rate can now be reduced by a factor of two without inducing aliasing.

2.3.1 MRA Overview

One of the fundamental assumptions of MRA is that there exists a set of nested signal subspaces of the signal space $L^2(\mathbf{R})$ with various resolutions:

$$V_{-\infty} \subset \ldots \subset V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \ldots \subset V_{\infty} = L^2(\mathbf{R}). \tag{2.3.1.1}$$

Each space V_i has a different basis that provides a different time resolution. The time resolution becomes coarser in the space V_i as index i decreases. In addition, MRA also assumes that there exist a scaling function $\phi(t)$ which constitutes an orthonormal basis of the space V_0

$$\phi_k(t) = \phi(t-k), \quad k \in \mathbf{Z}, \tag{2.3.1.2}$$

$$V_0 = span_k\{\phi_k(t)\}. \tag{2.3.1.3}$$

Another MRA fundamental assumption, the *twoscale* property, describes the relationship between two successive approximations of a signal or image as

$$f(t) \in V_i \Leftrightarrow f(2t) \in V_{i+1}. \tag{2.3.1.4}$$

Using equations from (2.3.1.2) to (2.3.1.4), we can derive an orthonormal basis $\phi_{i,k}$ for all the spaces V_i as

$$\phi_{i,k}(t) = 2^{i/2}\phi(2^i t - k), \quad i,k \in \mathbf{Z}. \tag{2.3.1.5}$$

Due to the fact that the resolution in the subspaces increases in powers of two, the scaling functions are *dyadic*. Moreover, since $\phi \in V_0 \quad (V_0 \subset V_1)$ and $2^{1/2}\phi(2t-n)$ span V_1, ϕ can be represented as a linear combination of $\phi(2t-n)$:

$$\phi(t) = \sum c(n)\phi(2t-n), \quad \text{with coefficients } c(n)_{n\in\mathbf{Z}}. \tag{2.3.1.6}$$

Notice that equation (2.3.1.5) resembles the discretized wavelet functions in (2.2.1). Hence, we can apply all the results we derived earlier. The basis in (2.3.1.5) is in fact compressed by a factor of two along the time axis every time the index i increases by one. The implication of equations (2.3.1.1) and (2.3.1.5) is that the frequency content of a coarser approximation is smaller than that of a finer level approximation. Thus, an efficient representation of a signal can be obtained by removing redundancy between any two successive signal approximations (i.e., $V_i \subset V_{i+1}$). For any successive approximations, a subspace W_i can be represented as a direct sum

$$V_{i+1} = V_i \oplus W_i, \quad i \in \mathbf{Z}. \tag{2.3.1.7}$$

W_i is said to be the orthogonal complement of V_i and is spanned by the same orthonormal basis in (2.3.1.5) [26]. From a signal processing point of view, a coarser image V_i is obtained by removing the high frequency content W_i from a finer image V_{i+1}. The decomposition of the signal space $L^2(\mathbf{R})$ in (2.3.1.1) can be rewritten in terms of the orthogonal complement $W_{i\in\mathbf{Z}}$:

$$V_i \oplus W_i \oplus W_{i+1} \oplus \ldots \oplus W_{-2} \oplus W_{-1} \oplus W_0 \oplus W_1 \oplus \ldots = L^2(\mathbf{R}). \quad (2.3.1.8)$$

The index i denotes the depth of decomposition, and any signal $f \in L^2(\mathbf{R})$ can be decomposed into a sum of subband signals in terms of scaling function ϕ and mother wavelet ψ [24] as follows

$$f(t) = \sum_m \alpha_i(m)\phi_{i,m}(t) + \sum_{j=i}^{\infty}\sum_m \beta_j(m)\psi_{j,m}(t). \quad (2.3.1.9)$$

The first summation in (2.3.19) represents $f(t) \in V_i$ whereas the second double summation represents $f(t) \in W_j$. In most applications, decomposition of signal f up to infinity level is not economically feasible. In practice, the signals are usually projected into a finite number of subspaces. In case of DWT, signals are element of a proper subspace in $L^2(\mathbf{R})$. Because $f(t) \in V_0 \subset L^2(\mathbf{R})$, the finite level decomposition of V_0 can be expressed analogous to (2.3.1.1) as

$$V_0 = V_i \oplus W_i \oplus W_{i+1} \oplus W_{i+2} \oplus \ldots \oplus W_{-1} \oplus W_0 . \quad (2.3.1.10)$$

Thus, (2.3.1.9) can be reduced to

$$f(t) = \sum_m \alpha_i(m)\phi_{i,m}(t) + \sum_m \beta_i(m)\psi_{i,m}(t). \quad (2.3.1.11)$$

In this section, we have bridged two concepts, namely the MRA in signal processing and the DWT in mathematics. This connection not only provides new perspectives and information, and the seeds for new developments, but also allows both concepts to utilize tools which already exists in each of these fields. As a fruitful result of this mutual relationship, we are going to explore the *filter banks* implementation of DTWT.

2.4 Fast Wavelet Transform

There is a strong tie between filter bank and dyadic wavelets. The development of filter bank started in the late 70's after Croisier, Esteban and Galand [19] first proposed the quadrature mirror filter (QMF). Further work and important insights contributed to this field from individuals, such as Smith and Barnwell [16], Vetterli [31] and Vaidyanathan [20], have been led to the study of perfect reconstruction filter banks. The MRA framework used in the analysis of wavelet/dyadic decompositions automatically associates a discrete-time perfect reconstruction filter bank to any wavelet decomposition. Daubechies [2] and Mallat [32] have shown that filter bank can also be used to generate wavelet bases as well. The main goal of this section is to introduce a computationally efficient filter bank structure for DTWT implementation. The algorithms to allow fast computation of DWT are collectively known as FWT.

2.4.1 Filter Bank Implementation of FWT

An analysis filter bank consists of lowpass and highpass filters operating together to separate a signal into frequency bands. Separately, these filters are usually not invertible but a stable solution for invertibility can be formed by combining them together. Moreover, the subband signals can be recombined into the original signal by a matched synthesis filter bank. This process is known as *subband coding* in DSP domain. Since the subband signals may be more efficiently compressed or transmitted than their original format, subband coding has been used in both audio and image applications.

Recall the MRA decomposition equation, the coefficients $\alpha_l(n)$ and $\beta_l(n)$ in (2.3.1.11) for a lower resolution level can be directly obtained from its finer resolution coefficients $\alpha_{l+1}(n)$ without resorting to $L^2(\mathbf{R})$ inner products. To

develop such a fast algorithm, we begin by expressing $\phi_{i,m}(t)$ of V_i in terms of $\phi_{i+1,m}(t)$ of V_{i+1} using (2.3.1.5) and (2.3.1.6):

$$\begin{aligned}\phi_{i,m}(t) &= 2^{1/2}\phi(2^i t - m) \\ &= 2^{1/2}\sum_v c(v)\phi(2^{i+1}t - 2m - v).\end{aligned} \tag{2.4.1.1}$$

Substituting $g(k) = 2^{-1/2}c(k), \quad k \in \mathbf{Z}$ and $2m + v = n$ into (2.4.1.1), we have

$$\begin{aligned}\phi_{i,m}(t) &= \sum_n g(n-2m)2^{(i+1)/2}\phi(2^{i+1}t - n) \\ &= \sum_n g(n-2m)\phi_{i+1,n}(t)\end{aligned} \tag{2.4.1.2}$$

In a similar fashion, wavelet basis $\psi_{i,m}(t)$ for space W_i is expressed as follows

$$\psi_{i,m}(t) = \sum_n h(n-2m)\phi_{i+1,n}(t). \tag{2.4.1.3}$$

Since the coefficients $\alpha_i(m)$ can be written as scalar products of signal f and $\phi_{i,m}(t)$ (Refer to A.1.3), $\alpha_i(m)$ can be expressed in terms of $\alpha_{i+1}(m)$ using the results of (2.4.1.2).

$$\begin{aligned}\alpha_i(m) &= \langle \phi_{i,m}(t), f(t) \rangle \\ &= \sum_n g(n-2m)\langle \phi_{i+1,n}(t), f(t) \rangle \\ &= \sum_n g(n-2m)\alpha_{i+1}(n) \\ &= g(-n) * \alpha_{i+1}(n) \mid_{n=2m.}\end{aligned} \tag{2.4.1.4}$$

Following a similar manner, the coefficients can be expressed in terms of $\beta_i(m)$:

$$\beta_i(m) = h(-n) * \alpha_{i+1}(n) \mid_{n=2m}. \tag{2.4.1.5}$$

Equations (2.4.1.4) and (2.4.1.5) imply that coefficients $\alpha_i(m)$ and $\beta_i(m)$ can be computed by performing a convolution of $\alpha_{i+1}(m)$ with series $g(-n)$ and $h(-n)$ respectively. The outputs are subsequently downsampled by a factor of two. These operations are depicted as an analysis filter bank in Figure 2-3 (3 decomposition levels configuration). In addition, the corresponding synthesis filter bank is shown for completeness. A synthesis filter bank is the dual inverse of its analysis filter bank counterpart.

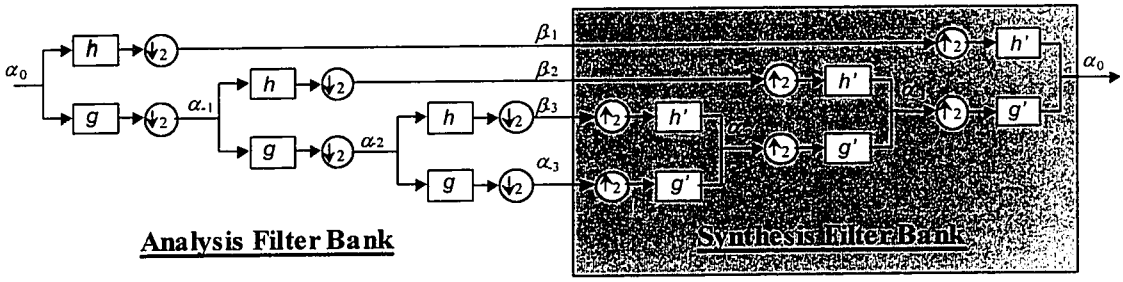


Figure 2-3 Octave filter bank: wavelet coefficients expansion using three-level analysis filter bank; reconstruction of original signal using a corresponding three-level synthesis filter bank.

In the analysis filter bank, the signal α_0 is decomposed into a high frequency component β_{-1} and a low frequency component α_{-1} via a pair of complementary digital filters h (highpass) and g (lowpass). These operations are reapplied to the lowpass subband component for further decomposition. Hence, the lowpass component is filtered by the same pair of filters, and the results are subsequently downsampled to generate two additional subband components. Identical to the operations in previous stage, decomposition creates an upper band (component

β_2) and an lower band (component α_2) of the input (component α_1). The overall decomposition scheme forms a dyadic tree-structure, which offers a practical way to implement FWT. The structure corresponds to an octave filter bank in DSP terminology.

Due to the downsampling (upsampling) operations in the analysis (synthesis) filter bank, the length of the input sequence is halved (doubled) in every decomposition (reconstruction) level. To illustrate the downsampling effect on the signal itself, the spectra for filtered signals after lowpass (X) and highpass (Y) operations and their corresponding spectra after downsampling are shown below. Notice that the frequency axis has been scaled with respect to the input sampling rate and the magnitude is halved after the downsampling. On the contrary, the effect of the upsampling operation is simply the reverse of the described downsampling effect.

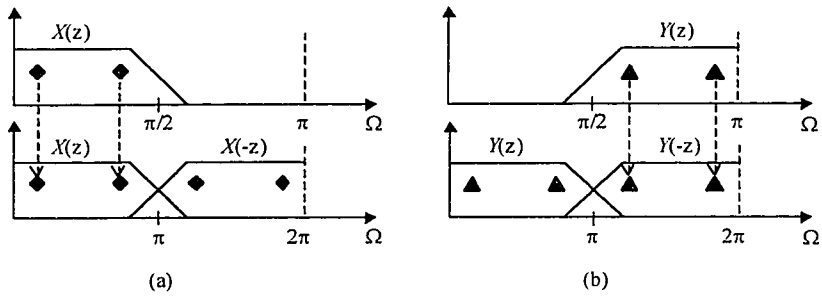


Figure 2-4 (a) Spectra of the lowpass filtered signal (top) and its downsampled signal (bottom); (b) Spectra of the highpass filtered signal (top) and its downsampled signal (bottom)

Using the same naming convention in our three-level octave filter bank example, the length of sequence in various stages can be summary as follows:

Sequences	α_0	α_{-1}, β_{-1}	α_{-2}, β_{-2}	α_{-3}, β_{-3}	α_{-j}, β_{-j}, $(0 \le j \le \infty)$
Length	N	$\frac{N}{2}$	$\frac{N}{4}$	$\frac{N}{8}$	$\frac{N}{2^j}$

Table 2-1 Sequence length for MRA-based DTWT implementation

One important factor that determines the usefulness of the above DTWT implementation is its computational complexity. Filtering is carried out by means of convolution, which consists of multiplication and addition operations. For any one-dimensional signal, assuming the first level decomposition requires C operations, only half of the operations $C/2$ are needed in the next immediate level due to downsampling. Hence, this poses an upper bound for the total number of filter operations C_{Total} for K decomposition stage as follows:

$$C_{Total} = C + \frac{C}{2} + \frac{C}{4} + \frac{C}{8} + \cdots + \frac{C}{2^{K-1}} = 2C\left(1 - \frac{1}{2^K}\right) \le 2C . \qquad (2.4.1.6)$$

Nonetheless, the filter delay D grows with K as demonstrated in equation (2.4.1.7). Since each subsequent decomposition stage runs at half the sampling rate as its previous one, the processing time doubles for each extra decomposition level. This limitation restricts the number of decomposition stage in real-time applications.

$$D_{Total} = D + 2D + 4D + 8D + \cdots 2^{K-1} D = (2^K - 1)D . \qquad (2.4.1.7)$$

Up to this moment, we show the possibility of using filter bank to implement FWT. Next, we state the filter conditions to ensure a stable FWT implementation from equations (2.4.1.4) and (2.4.1.5).

2.4.2 Perfect Reconstruction Orthogonal Filter Bank

When considering the application of wavelet decomposition of a signal at certain resolution levels, basis functions of compact support in time are necessitated by the requirement of finite computation time. Therefore, it is appropriate to limit our discussions to the case of compactly supported wavelets and scaling functions. In that case, only a finite number of g_k (follows definition in equation (2.4.1.2.) in the lowpass filter may be non-zero. In [33, 34], Lawton established that ϕ completely determines $\{g_k\}$ and vice versa. He also showed that the wavelet functions $\{\psi_{n,m}(t)\}_{(n,m)\in\mathbf{Z}^2}$ have to be a tight frame (i.e., orthogonal) of $L^2(\mathbf{R})$ in order to give a unique MRA.

Alias cancellation and distortion removal are two extra filter design conditions that are essential to guarantee perfect reconstruction of a signal [11, 12]. In addition, the orthonormality mentioned before requires the synthesis filters to be the time-reversed versions of their corresponding analysis filters. Moreover, to ensure filter stability and to reduce design and implementation complexity, all digital filters in the filter bank are assumed to be causal finite impulse response (FIR) filters[3] with real-valued filter coefficients. In [35-37], wavelets generalization and subband coding implementations using infinite impulse response (IIR) filters have been mentioned. All of these conditions, in turn, impose four filter requirements [27] for the analysis (h and g) and synthesis (h' and g') filter banks shown in Figure 2-3. They are:

[3] IIR filter banks have good frequency selectivity and low computation complexity when comparing to FIR filter banks. However, the orthogonal requirement in DWT requires all analysis filter banks to be causal and all synthesis filter banks to be anticausal or vice versa. Because of the fact that anticausal IIR filters cannot be implemented unless their impulse responses are truncated in general, IIR become less attractive to be used in realization.

$$\begin{aligned} h'(n) &= (-1)^n g'(p-1-n) \\ h(n) &= (-1)^{n+1} g'(n) \\ h(n) &= h'(p-1-n) \\ g(n) &= g'(p-1-n) \end{aligned} \qquad (2.4.2.1)$$

The above relationships imply that we can design a prototype filter, such as the lowpass analysis filter $g(n)$, and then use this prototype to obtain the rest of the other filters (e.g., g', h and h') according to (2.4.2.1).

Since the fundamental principle behind MRA algorithm relies on successive approximation, an additional constraint is imposed on $g(n)$ so as to ensure a stable solution. After substituting $g(k) = 2^{-1/2} c(k), \quad k \in \mathbf{Z}$ into (2.3.1.6), performing a Fourier transform, and carrying out the recursion for i times upon the results, we can obtain the following limit as i approaching infinity [17]:

$$\Phi(jw) = \prod_{k=0}^{\infty} 2^{-1/2} G\left(e^{\frac{jw}{2^{k+1}}}\right), \qquad (2.4.2.2)$$

where Φ and G are the Fourier transform of scaling function ϕ and the lowpass prototype filter g respectively. The limit in (2.4.2.2) exists if G is *regular* which lead to a wavelet function with some degree of smoothness (i.e., differentiable) via iteration [24]. *Regularity* is vital because smooth approximations or functions are desired. This regularity condition also applies to the mother wavelet ψ.

Perfect Reconstruction Orthogonal Filter banks are sometimes referred to as *paraunitary* [38]. A paraunitary system is termed a *lossless* system when it is causal and stable [39]. The frequency responses for all filters within a paraunitary filter banks are power complementary and the magnitude responses can be summarized as:

$$|H(\omega)| = |G(\pi + \omega)| = |G(\pi - \omega)|, \tag{2.4.2.3a}$$

$$|G(\omega)|^2 + |H(\omega)|^2 = 1. \tag{2.4.2.3b}$$

g is a half-band lowpass filter whereas h is a half-band highpass filter. Furthermore, the frequency responses for both filters form mirror image of each other about the line $\Omega = \pi/2$ as illustrated in Figure 2-5. Because of these characteristics, the DTWT filter pairs are often known as a Quadrature Mirror Filter (QMF) [19]. This link between DTWT and perfect reconstruction QMF was first discovered by Mallat [4]. Independently, Smith and Barnwell have called the filter with the above characteristics be Conjugate Quadrature Filter (CQF) in their engineering publications [16, 40].

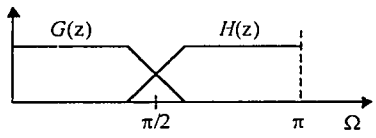


Figure 2-5 Transfer functions for QMF

In search of a compactly supported orthogonal wavelet bases that offers good time localization, regularity and smoothness, Daubechies [2] showed that it is possible to obtain a real-valued lowpass prototype filter g by placing a maximum number of zeros at $\omega = \pi$ (or $z = -1$ in the z-domain notation). The sufficient condition proposed by Daubechies for the prototype is

$$G(z) = \left(\frac{1+z^{-1}}{2}\right)^L A(z), \quad L \in \mathbf{Z}_+, \tag{2.4.2.4}$$

where L is the order of zero of $G(z)$ at $z = -1$. The solutions for (2.4.2.4) are FIR filters with L real-valued coefficients. As L increases, the degree of filter regularity as well as the smoothness of wavelet functions increases [11]. Another consequence of equation (2.4.2.4) is that $G(z)$ will have L vanishing moments (i.e., $\int t^m \psi(t)dt = 0, m = 0\ldots, L-1.$). The increase of vanishing moment implies that ψ will have more and more oscillations. To continue our search for prototype filter, we inserting (2.4.2.4) into (2.4.2.3b) and have

$$\left|\frac{1+e^{-jw}}{2}\right|^{2L} |A(\omega)|^2 + \left|\frac{1-e^{-jw}}{2}\right|^{2L} |A(\omega+\pi)|^2 = 1. \qquad (2.4.2.5)$$

Daubechies [2, 41] and other contributors [11, 12, 17, 26, 27] have shown a special halfband solution for $|A(\omega)|$ in (2.4.2.5):

$$|A(\omega)|^2 = \sum_{k=0}^{L-1} \binom{L+k-1}{k} \sin\left(\frac{\omega}{2}\right)^{2k}, \qquad (2.4.2.6)$$

The filter family that follows (2.4.2.4) and (2.4.2.6) is referred to as maxflat or maximally flat. This kind of filter has maximum flatness at $\omega = 0$ and $\omega = \pi$ and was derived by Herrmann [42] in the 70s'. For L = 2, equations (2.4.2.4) and (2.4.2.6) together gives the filter equation of $G(z)$ as follows:

$$G(z) = \frac{1}{4\sqrt{2}}\left[(1+\sqrt{3}) + (3+\sqrt{3})z^{-1} + (3-\sqrt{3})z^{-2} + (1-\sqrt{3})z^{-3}\right]. \qquad (2.4.2.7)$$

The filter solution in equation (2.4.2.7) is known as Daubechies D_2 filter (the subscript reflects the order of $G(z)$ zero at z = -1). This filter, as well as its family D_L, is compactly supported and forms a perfect reconstruction orthogonal filter bank. In addtion, the limit in (2.4.2.2) does exist for the Daubechies filter family.

For a particular choice of ϕ and ψ corresponding to each L, the regularity of Daubechies filter family increases linearly with L [2].

2.5 Multidimensional Wavelets

Up to this point, our discussions are concentrated on one-dimensional (1-D) wavelets. As mentioned in the Chapter 1, wavelets have gained popularity in areas such as image processing field. Since images are usually two-dimensional (2-D), 2-D wavelets and filter banks are required for the job. There are two types of filter available to construct multidimensional wavelets and filter banks: *separable* and *nonseparable* filters.

2.5.1 Separable Filter

Two-dimensional wavelets can be considered as tensor products of their 1-D wavelets counterpart [24, 43]. That is, the 2-D image or signal is treated as a composite of two 1-D signals. Taking a $i \times j$ image f as our example, one MRA decomposition requires two "passes" for this case. In the first pass (the row operation), i rows are processed by convoluting each of them with the lowpass (g'') and highpass (h'') filter pair separately. Then, every other column will be kept. In the second pass (the column operation), the two processed $i \times j/2$ images are fed into the same filter pairs again. This time every other row will be kept. Subsequently, the 2-D wavelet transform resulted in four subband images each with a size of $i/2 \times j/2$ (a quarter of the original image). Each subband images represents different resolution of the original image. The operations are depicted in Figure 2-6. Further MRA composition using the same filter pairs can be continued on the coarsest subband image (L,L). Again, the number of iteration depends on the application requirements. In Figure 2-6, we also show

the support region for a three level 2-D decomposition. Similar to the 1-D situation, the synthesis part can be implemented inversely.

Separable filter of size $i \times j$ offers only $i + j$ design variables whereas nonseparable filter of the same size offer $i \times j$ [43]. Hence, only rectangular divisions of the spectrum are possible. Some applications may need divisions that can better capture the image or signal energy. In that case, a nonseparable solution has to be chosen. Nonetheless, we give up ease of design for better energy separation because most of the 1-D filter bank concepts can be trivially extended to cover 2-D wavelets or higher. Recall that we have only one mother wavelet and one scaling function in the 1-D case. In the 2-D case, the number of scaling function remains at one but there will be three separable mother wavelets:

L,L $\quad \phi^{(1)}(x_1, x_2) = \phi(x_1)\phi(x_2)$ (2.5.1.1a)

L,H $\quad \psi^{(2)}(x_1, x_2) = \phi(x_1)\psi(x_2)$ (2.5.1.1b)

H,L $\quad \psi^{(3)}(x_1, x_2) = \psi(x_1)\phi(x_2)$ (2.5.1.1c)

H,H $\quad \psi^{(4)}(x_1, x_2) = \psi(x_1)\psi(x_2)$ (2.5.1.1d)

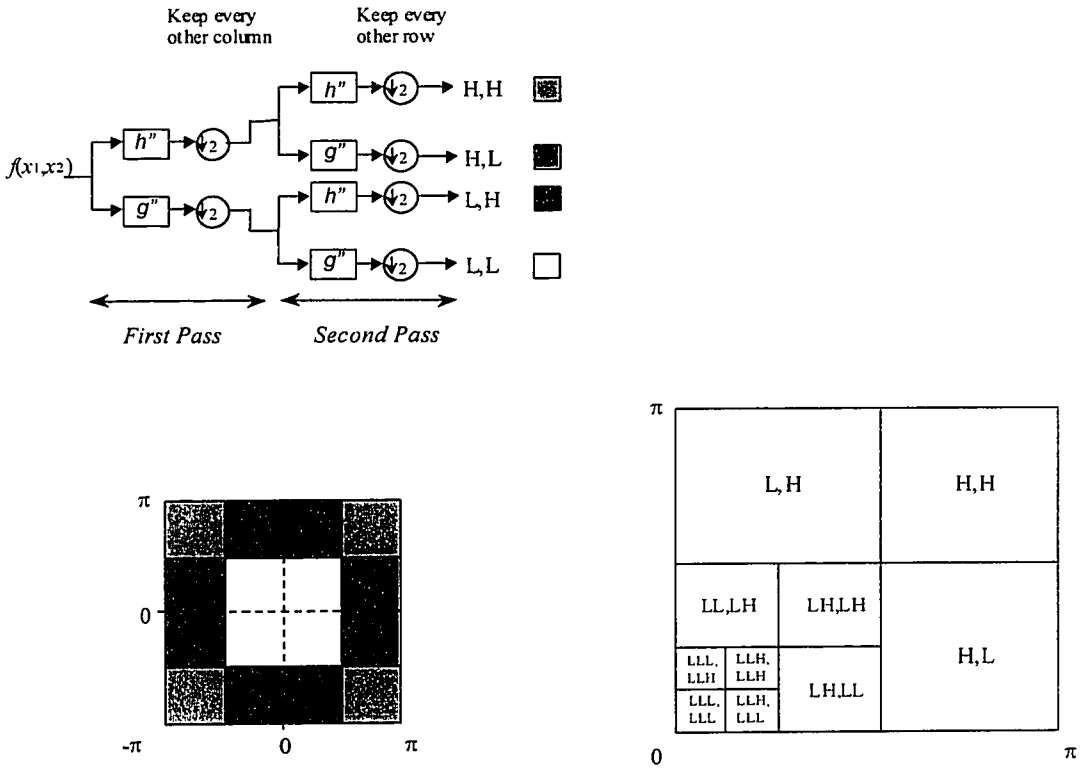


Figure 2-6 (Top Left) One level 2-D MRA decomposition – the analysis filters, (Bottom Left) Filters support regions for one level 2D MRA, (Right) Positive filters support regions for three-level 2D MRA decomposition.

2.5.2 Nonseparable Filter

Nonseparable multidimensional filter is very hard to design [44] due to the difficulty of imposing a zero at aliasing frequency. As in the 1-D case, zeros are essential for lowpass prototype filter stability and convergence to the scaling function. Even though 2-D nonseparable filter offers more free design variables, the difficulty in designing practical 2-D filter directly made it less appealing when comparing to separable design (using factorization method). In fact, nonseparable orthogonal wavelets with high regularity have not been found. Less restricted 2-D biorthogonal filter is possible [24] but such design process is still not as simple as the separable case.

Multidimensional wavelets are still an active research area. A family of multidimensional orthogonal filter similar to the 1-D Daubechies filter family is yet to be found [25].

Chapter 3

DTWT MODEL

Before discussing the approach used by this project to model a 1-D MRA-based DTWT in MATLAB, a literature research on the development of DTWT VLSI architecture and the finite wordlength analysis is presented here.

3.1 Previous Research in DTWT VLSI Implementation and Finite Wordlength Analysis

Most of the research available is concentrated on architecture design rather than finite wordlength effects, which are being investigated in this project. It is important to be familiar with the current trend so that the most popular and widely used algorithm for DTWT VLSI implementation (i.e., MRA) is selected for our study.

3.1.1 DTWT VLSI Architectures

The first published DTWT architecture was proposed by Knowles [45] in 1990. This architecture employed large multiplexors for routing the intermediate results. Hence, it is not suitable to map into VLSI. A year later, Knowles and Lewis [46] together proposed a 2-D DTWT architecture using no multiplier. However, this architecture is an ad hoc solution for Daubechies 4-tap wavelets (e.g., $N = 2$ in Table 4-1 and Table 4-3) only. In the same year, Aware Inc. has introduced a 30

MHz 1-D Wavelet Transform Processor (WTP) [47] which executes computation in a synchronous pipeline fashion.

Parhi and Nishitani [48] introduced two classes of VLSI architectures to implement 1-D and 2-D DTWT and are referred to as the folded architecture and the digital-serial architecture. The former architecture has lower latency but consumes relatively larger chip space than that of the latter one. Their paper also proposed a 2-D DTWT implementation using a combination of both architectures. Nonetheless, the architectures require extensive redesign in order to scale for different filter length and number of decomposition stages.

Vishwanath, Owens and Irwin studied the feasibility of implementing the 1-D and 2-D DTWT in [49]. Their proposed architectures, based on *linear systolic array,* rely on a so-called *global routing network* to manage inputs and outputs to and from the systolic filters. As a result, their designs provide higher scalability and modularity than previous proposals.

Grzeszczak, Mandal, Panchanathan and Yeap published a *systolic array* architecture for one or multidimensional DTWT in [50]. This architecture is an improvement over [48] by sharing one set of multipliers and adders for both highpass and lowpass coefficient calculations, in contrast to employing two parallel computational hardware. The design also employed a *global routing control* similar to the one used in [49] to simplify control circuit for individual filter cell. The drawback for this design is that the number of multipliers required equals to the filter length.

Fridman and Manolakos proposed two 1-D DTWT architectures (i.e., semi-systolic array architecture and systolic array architecture) in [51]. Both proposals are built on Processing Element (PE) that is based on the results of the *data dependence* and *localization* analysis [52]. The authors claimed that their designs

require small memory and simple control circuit. Moreover, the modularity property of PE has made both designs suitable for realization in VLSI and programmable processors.

All of the above DTWT VLSI architectures achieve asymptotic lower latency bound of *O(M)*, except [47] which is *O(MlogM)* where *M* is the length of input sequence. Majority of the designs utilize the MRA theory to ensure fast DTWT coefficients computation. Also, authors in [48, 50] use *forward-circular* scheme and *forward-backward register* allocation technique to derive minimum number of register usage. In fact, the *life time* analysis [53], a common design tool in the compiler field, shows that forward-backward register allocation technique provides the minimum number of register utilized. However, as mentioned in [48], there is a trade-off in the number of registers and the complexity of the control circuit. Generally speaking, reducing the number of registers results in extra routing which may take up even more space on the silicon after all.

3.1.2 Finite Wordlength Analysis

The effects of finite wordlength have been briefly mentioned for two-channel QMF bank in [54-56] and for the wavelet QMF bank in [50]. In [57], Westerink, Biemond and Boekee applied the *"gain plus additive noise"* model and broke down the output error of a two-channel QMF system into four different types of errors (e g, aliasing, QMF, signal and random errors). Later, Kovacevic [58] shows that all signal-dependent errors at the output of a QMF two-channel system can be cancelled with an appropriate choice of synthesis (reconstruction) filters, and the Daubechies family used in this project is one of them. In fact, using the "appropriate" filter only shift signal-dependent error into random error (i.e., the total energy of error remains comparable.) which can then be minimized with any noise removal technique. Independently, Uzun and Haddad derived similar results using cyclostationary modeling for a 2-channel subband codecs [59, 60].

In [61], Yang, Chan and Chen proposed a 24-tap QMF subband filterbank which is based on *recurrence* implementation and polyphase filtering. The authors then provide a quantitative study on the performance of their proposed implementation structure for HDTV application.

3.2 DTWT MATLAB Model

As mentioned in Chapter 2, Multi-Resolution Analysis (MRA) provides a computational-efficient algorithm to perform forward and inverse discrete time wavelet transforms (DTWT). MRA performs each decomposition or reconstruction stage by an iterative process of lowpass and highpass filter operations, followed by either upsampling or downsampling of the resultant signals. The configuration of a three stages DTWT system is illustrated in Figure 2-3. QMF bank is usually employed to implement the mentioned filter operations. Since its introduction, DTWT has quickly demonstrated its ability in a wide spectrum of applications ranging from image process to non-stationary signal analysis. Due to the real-time nature of many applications, hardware implementation of DTWT is usually preferred. One way to implement DTWT is to use programmable digital signal processor such as the TMS320Cx DSP family from Texas Instruments. However, VLSI remains as the most cost-effective way to implement DTWT in volume and this project is concentrated on the study of numerical accuracy for the MRA-based DTWT VLSI implementation.

3.2.1 Choice of Mother Wavelet

Unlike the DCT (Refer to Section A.2.4) or other linear transformation schemes where the transformation is performed onto one particular basis, in WT more than one single set of basis are used. Thus, extra degrees of freedom are available to the design of WT system so that it can possess abilities that other transformation methods do not have. For example, different wavelet filter leads

to different base. Because of the same reason, the Daubechies filter family [2] is selected for this project. This wavelet family is an optimum choice for our DTWT implementation because it has proven itself as a good basis for a wide range of application such as image processing [43]. Moreover, the Daubechies family corresponds to a filter class known as the orthogonal filter that ensures perfect reconstruction (Refer to Section 2.4.2). In addition, the lowpass and highpass QMF in analysis and synthesis filter banks are related to each other as stated in equations (2.4.2.1). The analysis and synthesis filter pair is mirror image of each other and all filters have the same length. For individual causal QMF filters, the system function has only zeros (except poles at $z = 0$).

In case when the number of filter coefficients equals M, the total number of multiplication and addition operations in a decomposition stage for a N length signal is $2MN$ and $2(M\text{-}1)N$ respectively (including both highpass and lowpass filters). Recall from Table 2-1, the next decomposition stage requires only half of the multiplication and addition operations performed in the previous stage.

As mentioned in previous paragraph, the longer the filter length, the more arithmetic operations are required in DTWT. Also pointed out in [50], 12 taps filter are the upper bound for many real-time applications. Therefore, our choices of Daubechies wavelet QMF are limited to 12 taps (i.e. $N = 6$)

3.2.2 Round-off methods

The binary number system is the most conventional internal representation for numbers within digital computers. Some common binary number formats are *sign and magnitude, 1's-complement and 2's-complement.* Since 2's-complement representation is the most commonly used format and it is best suit for representing binary numbers, 2's-complement is therefore selected for this project.

A real number can be expressed in 2's-complement format with infinite wordlength (3.1.2.1) or finite wordlength (3.1.2.2),

$$x = G\left(-a_0 + \sum_{i=1}^{\infty} a_i 2^{-i}\right), \tag{3.1.2.1}$$

$$\hat{x} = G\left(-a_0 + \sum_{i=1}^{M} a_i 2^{-i}\right), \tag{3.1.2.2}$$

where G is the scaling factor, a's are binary digits (a = 0 or 1) and a_0 is the sign bit. $\hat{x}$ is the $(M+1)$ bits finite representation of x and the resolution for $\hat{x}$ is $\Delta = G2^{-M}$. Any real number with magnitude less than or equal to G can be represented by the above equations. To simplify the investigation, G will be set to 1 throughout this project. In addition, a real number can be converted from infinite precision to finite one by nonlinear operations such as rounding or truncation. If we define quantization error ε as

$$\varepsilon = \hat{x} - x. \tag{3.1.2.3}$$

The quantization error for 2's-complement rounding and truncation will be

Rounding: $$\frac{-2^{-M}}{2} < \varepsilon \le \frac{2^{-M}}{2}, \tag{3.1.2.4}$$

Truncation: $$-2^{-M} < \varepsilon \le 0. \tag{3.1.2.5}$$

In a rounding operation, a real number is converted to a M-bit number closest to its unrounded value. Therefore, the quantization error has zero mean and a width of 2^{-M}. In a truncation operation, the less-significant bits of an unrounded number is chopped off so as to fit into the available finite wordlength. Therefore, the quantization error is shifted $2^{-M}/2$ but the width remains

unchanged. The quantization noise probability density for rounding and truncation are depicted in Figure 3-1. Quantization effects due to rounding and truncation operations will be investigated in this project.

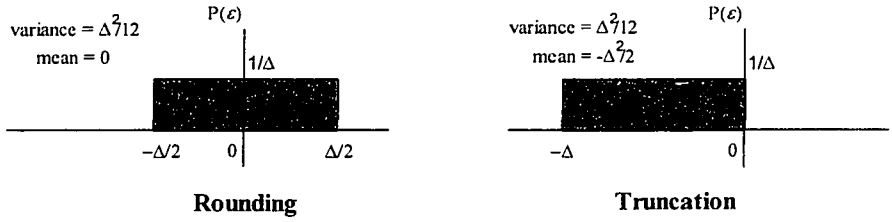


Figure 3-1 Quantization error distribution in rounding and truncation

3.2.3 Computational Quantization

The effect of round-off is modeled as an independent white noise source, which injects noise at each multiplication in fixed-point arithmetic. To simplify the investigation, each noise source is assumed to be uncorrelated with all other noise sources and the input. It is obvious that the assumption is not valid as the degree of quantization depends on the input. In fact, Grzeszczak, Mandal, Panchanathan and Yeap [50] has demonstrated that the signal reconstruction error in DTWT increases when the input data changes rapidly. Nonetheless, Gold and Rader [62] has pointed out that when the signal traverses many quantization stages, this linear noise model leads to accurate predictions of statistical characteristics such as mean and variance.

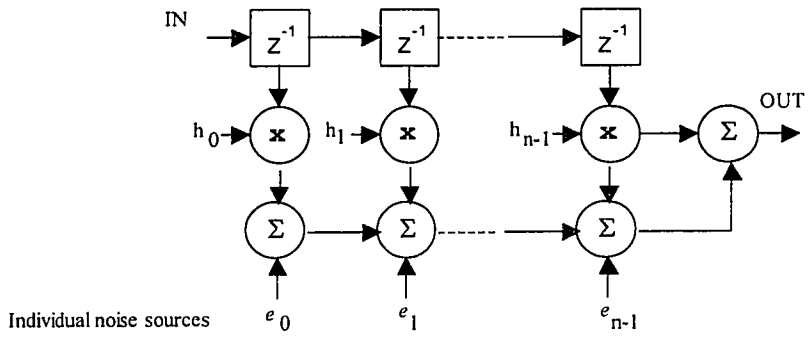


Figure 3-2 Linear noise model for digital FIR filter. All non-linear quantization errors due to finite arithmetic are replaced by uncorrelated noise sources.

Unlike IIR, FIR does not compose a feedback path so that each individual noise is passed to the output without entering the system for further calculation (Details given in Section 4.1.1). Hence, the round-off noise for FIR system is proportional to the number of filter taps and independent of filter coefficient. The noise variance is expressed as

$$\sigma^2 = (N+1)\left(\frac{2^{-2M}}{12}\right) \tag{3.1.2.6}$$

where $(N+1)$ is the number of FIR coefficients and the remaining fraction is the noise set by the quantization step-size of the filter computation. A graphic representation of the equation is illustrated in Figure 3-2.

3.2.4 Sample Output of DTWT Model

The DTWT model developed in this project is written in MATLAB based on the criteria listed in previous sections. The 1-D DTWT MATLAB model function is

listed in B.13 (`wt1d`). The input parameters of this function allows users to specify the input sequence, the length of data to be processed, the number of decomposition stages (or reconstruction stages), the number of Daubechies filter coefficients (e.g., the number is 4 for 2nd order Daubechies filter and 6 for 3rd order Daubechies filter), the transform mode (i.e., decomposition [forward DTWT] or reconstruction [inverse DTWT]), the quantization mode (i.e., no quantization, rounding or truncation), and the number of bits to represent filter coefficients.

In Figure 3-3, the sample output using MATLAB function `wt1d` is shown (Refer to MATLAB program `wtdemo1` in B.14). The input sequence (a 64-unit long square wave) is represented by the yellow curve. The input sequence is first decomposed by a three stages MRA-based DTWT system using 3rd order ($N = 3$ in Table 4-1) Daubechies filter. The round-off method employed for both the filter quantization and internal result is rounding (rather than truncation) and the number of bits to represent filter coefficient and internal results are 5-bit and 10-bit respectively. The resulting sequence is then reconstructed using similar settings. The output is represented by the purple curve. Notice that the output curve does not regenerate perfectly due to non-ideal effects of the DTWT system.

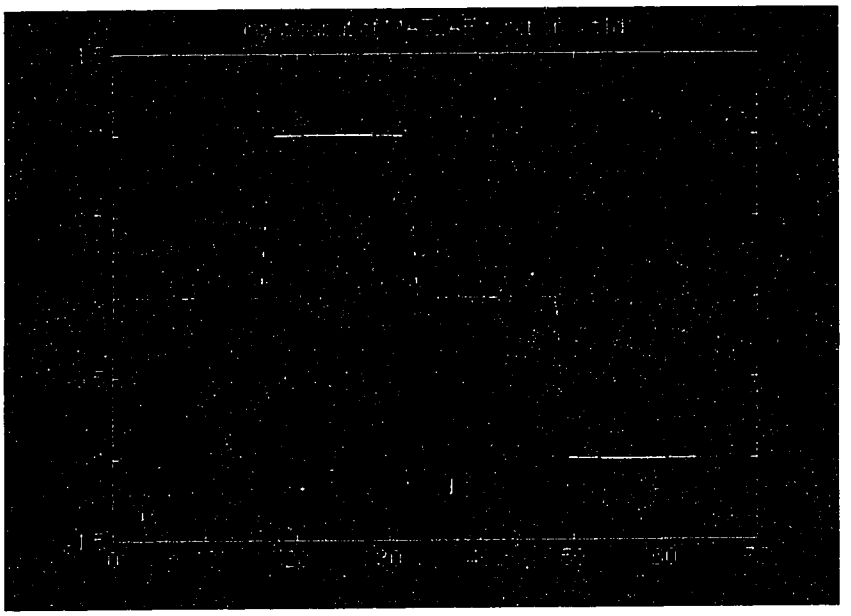


Figure 3-3 The input (yellow) and output (purple) of MATLAB function: wt1d.

3.2.5 Selective Subband Reconstruction

The MATLAB DTWT model not only provides a tool for prototyping and simulation of non-ideal effects due to quantization of filter coefficient and internal computation results, but also allows system designers to simulate selective subband reconstruction with the presence of non-ideal effects aforementioned.

Recall from Figure 2-1 that at each decomposition stage, the input sequence (e.g., α_0) is processed into high frequency portion (e.g., β_1) and low frequency portion (e.g., β_2). The low frequency portion is then used as input for the next decomposition stage if necessary. Due to the high frequency nature of non-ideal effects, the high frequency portion of the input sequence may be ignored in the

MRA-based DTWT system since the majority of the information encoded in this portion is noise (e.g, noise introduced in the transmission channel of the sequence). By removing the high frequency portion, component β_1 in Figure 2-3, the number of filters and corresponding control circuit for a 3-stage MRA-based DTWT configuration will be reduced to the modified configuration as shown in Figure 3-4. The removal of highpass filters will not reduce the chip size if the VLSI DTWT system is designed in a way that highpass and lowpass operations in a filter stage are sharing the same set of filter. Furthermore, the overall computation speed will not increase because the lower arm (lowpass filter path) is the limiting factor in processing time as suggested by equation (2.4.1.7) (i.e., removal of highpass filters does not change the length of the lowpass filter path). Nonetheless, the control circuit and scheduling algorithm employed for filter computation will be simplified due to the removal of the high frequency portion in a particular filter stage. Depending on the level of tolerance for errors in reconstruction, high frequency portion(s) in subsequent decomposition (reconstruction) stage(s) can be further removed in exchange for higher reconstruction errors.

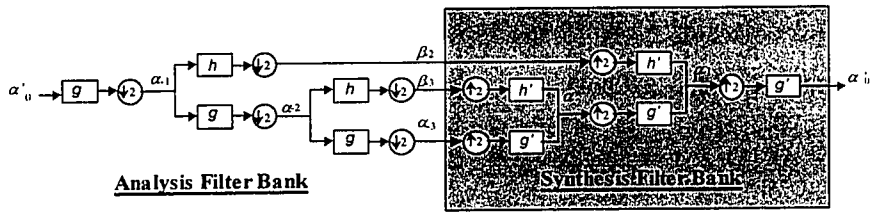


Figure 3-4 Modified Octave filter bank: wavelet coefficients expansion using three-level analysis filter bank; reconstruction of original signal using a corresponding three-levels synthesis filter bank.

In Figure 3-2, MATLAB DTWT model (wtd1) is used to illustrate the non-ideal effects of selective subband reconstruction discussed above (The figure is generated by MATLAB program wtdemo2 in B.15). The input sequence (a 128-unit long square wave) is represented by a yellow curve. The input sequence is first decomposed by a three stage MRA-based DTWT system using 2nd order (N = 2 in Table 4-1) Daubechies filters. The round-off method employed for both the filter quantization and internal result is rounding and the number of bits to represent filter coefficient and internal results are 5-bit and 10-bit respectively. The resulting sequence is then reconstructed using similar settings.

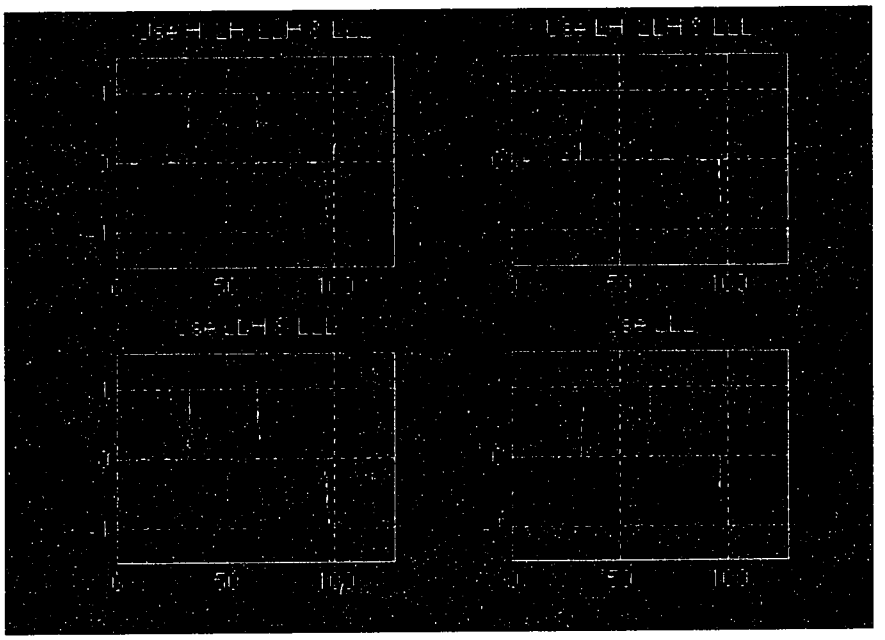


Figure 3-6 Simulation of selective subband reconstruction with the presence of non-ideal effects

The output for each scenario is represented by a purple curve. The top left subplot shows the input and output for the complete MRA-based DTWT system (no high frequency portion removed; refer to Figure 2-3). The top right subplot shows the input and output for the system depicted in Figure 3-4 (i.e., $\beta_{.1}$ is removed). An additional high frequency portion is removed in the bottom left subplot (i.e., both $\beta_{.1}$ and $\beta_{.2}$ are removed). Finally, the bottom right subplot shows the output of using only the low frequency portion for reconstruction (i.e., $\beta_{.1}$, $\beta_{.2}$ and $\beta_{.3}$ are removed). Notice the deterioration of output as less information is used in the reconstruction process.

In this chapter, we have described the 1-D MRA-based DTWT MATLAB model developed for this project. The model allows user to prototype N-stage decomposition and reconstruction DTWT with the freedom to specify the number of bits to represent and the rounding-off model, which are both used by the Daubechies filter coefficients and the internal computational results. An example to decompose and then to reconstruct a square wave utilizing the model is provided. Finally, we use the model to illustrate the idea of removing high frequency portion(s) of the DTWT system. The MATLAB DTWT model developed in this project provides a valuable tool for system designers to study the possibility of selective subband reconstruction. The model can be use to evaluate the trade off between the reduction of control circuit (i.e., removal of highpass filter in a DTWT stage) and the reconstruction error with the presence of non-ideal effects due to quantization of filter coefficients and computational results.

Chapter 4

FINITE WORDLENGTH EFFECTS OF FILTERS IN DTWT SYSTEM

The MATLAB model developed in previous chapter provides important information regarding overall performance of a MRA-based DTWT system with the presence of non-ideal effects. In this chapter, we focus our study on the non-ideal effects of individual filters within the DTWT system. First, we conduct an investigation of non-ideal effects of direct structure filter implementation. Then, we look into another filter implementation, namely the lattice structure filter implementation. Lastly, performance results for both filter implementations are compared.

4.1 Finite Wordlength Effects of Direct Structure Filter

In this section, we study the finite wordlength effect of filter model using direct structure implementation. First, the reason of concentrating the study on Finite Impulse Response (FIR) filter is given. Then, the non-ideal effects in both magnitude and phase responses due to finite wordlength are simulated using

MATLAB. Results are presented graphically later in this chapter. Complete numerical results are tabulated in Appendix C.

4.1.1 Choice of Filter Type

Under ideal conditions, such as the availability of unlimited wordlength for filter coefficients and internal storage, a DTWT composed by digital filters should behave the same regardless of the choice of filter types (e.g., FIR or IIR). Despite the fact that IIR system can usually fulfil the same filter requirement with less hardware and processing time [63], system designers often prefer FIR to IIR in reality. The reason can be seen easily if we look at the two lowpass FIR and IIR filters as shown in below.

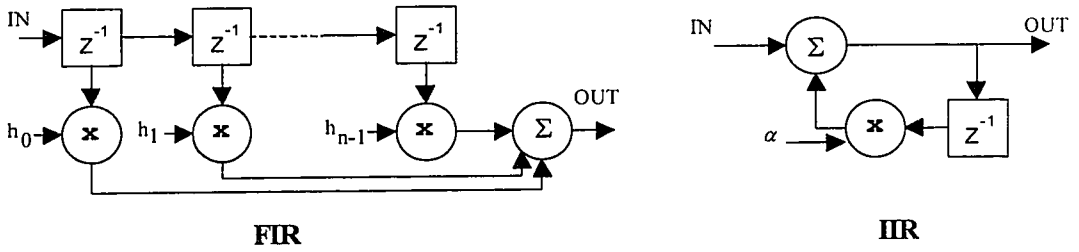


Figure 4-1 (Left) FIR lowpass filter; (Right) IIR lowpass filter.

The IIR filter feeds its output back for subsequent calculation whereas the FIR filter does not. The feedback path in IIR system raises two design issues. First of all, the number of bits required to express the IIR output exactly increases after every iteration. For example, if both the filter coefficient (α) and data are a bits wide, the number of bits required for the result which loops back to the summing point after one iteration will be doubled to $2a$ bits wide. On the contrary, this phenomenon does not occur in the FIR system due to the absence of feedback path. However, in case when the IIR system has a long memory (i.e., as α

approaches 1), the higher order products will not be negligibly small. Inevitably, the rounding error will propagate back to the system, resulted in relatively large error accumulation. Another related design issue for IIR system is the *zero-input limit cycle*. Finite wordlength generates a deadband within which input corrections are not fed back during iteration. Consequently, the limit cycle occurs when the IIR output oscillates indefinitely at small amplitude long after the input has gone to zero. Zero-input limit cycles caused by overflow [64] or round-off quantization [65] are unique to IIR system. FIR systems are immune from limit cycles due to the lack of feedback path. In view of these reasons, FIR systems are usually preferred in applications where limit cycle oscillation cannot be tolerated. In addition, linear phase conditions are easily preserved in the direct form FIR system [65] then IIR system. Therefore, our investigation only concentrates on a special case of FIR (e.g., QMF filter).

The direct structure FIR model shown in Figure 4-1 is the building block for the highpass and lowpass filter pairs in a DTWT system. Recall from Figure 2-3, the highpass filter for signal decomposition is labeled h whereas its counterpart in the reconstruction stage is labeled h'. Similarly, the lowpass filter for signal decomposition stage is labeled g whereas its counterpart in the reconstruction stage is labeled g'. The high-pass decomposition filter coefficients (h) for Daubechies family members ($N = 2$ to $N = 6$) are tabulated in Table 4-1. Due to the relationships described by equations (2.4.2.1.) in Section 2.4.2., we can obtain all the corresponding filters coefficients by merely knowing the filter coefficients for one particular filter (i.e., the prototype filter). For example, highpass reconstruction filter coefficients (h'), lowpass decomposition (g) and reconstruction (g') filter coefficients can all be readily computed from the high-pass decomposition filter coefficients (h). The MATLAB function (`pwtldgento`) computes all filter coefficients from the high-pass reconstruction filter coefficients is listed in B.9 of Appendix B. Also, the

MATLAB function (`daubfcoeff`) that generate the high-pass decomposition filter coefficients for Daubechies family in Table 4-1 is listed in B.6.

Daubechies family	Daubechies filter coefficients
$N = 2$	$h_2(0)$ = 0.482962913145 $h_2(1)$ = 0.836516303738 $h_2(2)$ = 0.224143868042 $h_2(3)$ = -0.129409522551
$N = 3$	$h_3(0)$ = 0.332670552950 $h_3(1)$ = 0.806891509311 $h_3(2)$ = 0.459877502118 $h_3(3)$ = -0.135011020010 $h_3(4)$ = -0.085441273882 $h_3(5)$ = 0.035226291882
$N = 4$	$h_4(0)$ = 0.230377813309 $h_4(1)$ = 0.714846570553 $h_4(2)$ = 0.630880767930 $h_4(3)$ = -0.027983769417 $h_4(4)$ = -0.187034811719 $h_4(5)$ = 0.030841381836 $h_4(6)$ = 0.032883011667 $h_4(7)$ = -0.010597401785
$N = 5$	$h_5(0)$ = 0.160102397974 $h_5(1)$ = 0.603829269797 $h_5(2)$ = 0.724308528438 $h_5(3)$ = 0.138428145901 $h_5(4)$ = -0.242294887066 $h_5(5)$ = -0.032244869585 $h_5(6)$ = 0.077571493840 $h_5(7)$ = -0.006241490213 $h_5(8)$ = -0.012580751999 $h_5(9)$ = 0.003335725285
$N = 6$	$h_6(0)$ = 0.111540743350 $h_6(1)$ = 0.494623890398 $h_6(2)$ = 0.751133908021 $h_6(3)$ = 0.315250351709 $h_6(4)$ = -0.226264693965 $h_6(5)$ = -0.129766867567 $h_6(6)$ = 0.097501605587 $h_6(7)$ = 0.027522865530 $h_6(8)$ = -0.031582039318 $h_6(9)$ = 0.000553842201 $h_6(10)$ = 0.004777257511 $h_6(11)$ = -0.001077301085

Table 4-1 Daubechies filter coefficients and their corresponding lattice filter coefficients for $N = 2$ to 6

4.1.2 Filter Coefficient Quantization

Since the DTWT filter coefficients can be expressed with finite number of bits, the quantized coefficients may lead to a shift in zeros for FIR, resulting in a different QMF filter from the one we have original designed. Nonetheless, the degradation in the filter characteristics for FIR should be less severe than that of IIR for the reasons mentioned in Section 4.1.1. The frequency response $\hat{H}$ of a FIR system with quantized coefficients can be expressed as the sum of the unquantized frequency response H and the frequency response error ΔH:

$$\hat{H}(e^{jw}) = H(e^{jw}) + \Delta H(e^{jw}), \tag{4.1.2.1}$$

$$\Delta H(e^{jw}) = \sum_{i=0}^{N} \Delta h[n] e^{-jwn}, \tag{4.1.2.2}$$

where N is the number of tap of the FIR filter and $\Delta h[n]$ is the coefficient error caused by quantization. Follow our previous discussion in rounding, if the filter coefficients are rounded to (M+1) bits, a bound of the coefficient error can be derived using (4.1.2.2). A bound on the frequency response error is obtained by replacing the coefficient errors $\Delta h[n]$ and the complex exponentials with their maximum magnitudes [65]. i.e,

$$|\Delta H| = \left| \sum_{n=0}^{N} \Delta h[n] e^{-jwn} \right| \le \sum_{n=0}^{N} |\Delta h[n]| \left| e^{-jwn} \right| \le (N+1) 2^{-(M+1)}. \tag{4.1.2.3}$$

4.2 Simulation Results

With the VLSI implementation of DTWT for real-time application in mind, only short filter members in the Daubechies family were studied. That includes Daubechies wavelet order ranged from $N = 2$ to 6 (i.e., 4 taps to 12 taps). Since

the lowpass/highpass QMF filter pair forms mirror images of each other at $\pi/2$ before downsampled, the lowpass FIR filter studied in this section can easily be extended to its highpass counterpart. For each wavelet order, the "infinite" precision filter coefficients in Table 4-1 were mapped into various finite wordlengths ranging from 3 to 12 bits. The MATLAB functions that perform round-off operations are listed in B.3 (rounding) and B.4 (truncation). Examples of using truncation and rounding in 8-bit and 12-bit representations of 3rd and 6th orders Daubechies filter coefficients are tabulated in Table 4-2. In the following sections, we will observe that the coarseness of the quantization contributes errors in both phase and magnitude responses of the filter.

Simulations were performed on the direct structure filter model. Results for each wavelet order are tabulated in three separate tables in Appendix C. They are namely the magnitude responses (rounding and truncation) and the phase responses. In the magnitude response tables, the magnitude responses errors $\left|\Delta H(e^{jw})\right|$ as defined in equation (4.1.2.3) are recorded in decibel (dB) for both passband and stopband regions. In addition, maximum value, mean as well as variance for each scenario (e.g., wavelet order, quantization bit, passband or stopband) are also presented. Furthermore, the maximum coefficient errors (e.g., $\Delta h[n]$ in equation [4.1.2.2]) for each quantization scenario are included. In the phase response tables, the phase responses are recorded in degree for both passband and stopband regions. The mean and the variance for both regions were are presented. In the following, findings, observations and graphs regarding the simulations are provided.

	Daubechies family	Daubechies filter coefficients	8-bit representation	12-bit representation
Truncation	$N=3$	$h_3(0)$ = 0.332670552950	3.28125×10^{-1}	$3.3251953125 \times 10^{-1}$
		$h_3(1)$ = 0.806891509311	8.046875×10^{-1}	$8.06640625 \times 10^{-1}$
		$h_3(2)$ = 0.459877502118	4.53125×10^{-1}	$4.5947265625 \times 10^{-1}$
		$h_3(3)$ = -0.135011020010	-1.40625×10^{-1}	$-1.3525390625 \times 10^{-1}$
		$h_3(4)$ = -0.085441273882	-8.59375×10^{-2}	$-8.544921875 \times 10^{-2}$
		$h_3(5)$ = 0.035226291882	3.125×10^{-2}	3.515625×10^{-2}
	$N=6$	$h_6(0)$ = 0.111540743350	1.09375×10^{-1}	$1.11328125 \times 10^{-1}$
		$h_6(1)$ = 0.494623890398	4.921875×10^{-1}	$4.94140625 \times 10^{-1}$
		$h_6(2)$ = 0.751133908021	7.5×10^{-1}	$7.509765625 \times 10^{-1}$
		$h_6(3)$ = 0.315250351709	3.125×10^{-1}	$3.1494140625 \times 10^{-1}$
		$h_6(4)$ = -0.226264693965	-2.265625×10^{-1}	-2.265625×10^{-1}
		$h_6(5)$ = -0.129766867567	-1.328125×10^{-1}	$-1.298828125 \times 10^{-1}$
		$h_6(6)$ = 0.097501605587	9.375×10^{-2}	$9.716796875 \times 10^{-2}$
		$h_6(7)$ = 0.027522865530	2.34375×10^{-2}	2.7343750×10^{-2}
		$h_6(8)$ = -0.031582039318	-3.90625×10^{-2}	$-3.173828125 \times 10^{-2}$
		$h_6(9)$ = 0.000553842201	0	4.8828125×10^{-4}
		$h_6(10)$ = 0.004777257511	0	$4.39453125 \times 10^{-3}$
		$h_6(11)$ = -0.001077301085	-7.8125×10^{-3}	$-1.46484375 \times 10^{-3}$
Rounding	$N=3$	$h_3(0)$ = 0.332670552950	3.359375×10^{-1}	$3.3251953125 \times 10^{-1}$
		$h_3(1)$ = 0.806891509311	8.046875×10^{-1}	$8.0712890625 \times 10^{-1}$
		$h_3(2)$ = 0.459877502118	4.609375×10^{-1}	$4.599609375 \times 10^{-1}$
		$h_3(3)$ = -0.135011020010	-1.328125×10^{-1}	$-1.3525390625 \times 10^{-1}$
		$h_3(4)$ = -0.085441273882	-8.59375×10^{-2}	$-8.544921875 \times 10^{-2}$
		$h_3(5)$ = 0.035226291882	3.90625×10^{-2}	3.515625×10^{-2}
	$N=6$	$h_6(0)$ = 0.111540743350	1.09375×10^{-1}	$1.11328125 \times 10^{-1}$
		$h_6(1)$ = 0.494623890398	4.921875×10^{-1}	$4.9462890625 \times 10^{-1}$
		$h_6(2)$ = 0.751133908021	7.5×10^{-1}	$7.509765625 \times 10^{-1}$
		$h_6(3)$ = 0.315250351709	3.125×10^{-1}	$3.154296875 \times 10^{-1}$
		$h_6(4)$ = -0.226264693965	-2.265625×10^{-1}	$-2.2607421875 \times 10^{-1}$
		$h_6(5)$ = -0.129766867567	-1.328125×10^{-1}	$-1.298828125 \times 10^{-1}$
		$h_6(6)$ = 0.097501605587	9.375×10^{-2}	9.765625×10^{-2}
		$h_6(7)$ = 0.027522865530	3.125×10^{-2}	2.7343750×10^{-2}
		$h_6(8)$ = -0.031582039318	-3.125×10^{-2}	$-3.173828125 \times 10^{-2}$
		$h_6(9)$ = 0.000553842201	0	4.8828125×10^{-4}
		$h_6(10)$ = 0.004777257511	7.8125×10^{-3}	4.8828125×10^{-3}
		$h_6(11)$ = -0.001077301085	0	-9.765625×10^{-4}

Table 4-2 Unquantized and Quantized (8-bit and 12-bit) Coefficients for 3rd and 6th orders of the Daubechies highpass filter (Direct Structure Implementation)

4.2.1 Magnitude Responses for the Direct Structure

The simulation results of the direct structure magnitude responses using rounding and truncation are tabulated in Table C-1 and Table C-2 respectively. According to these tables, the maximum coefficient errors $\Delta h[n]\big|_{\max}$ for both rounding and truncation cases decrease as the number of bits used to quantize the Daubechies wavelet filter coefficients increases. The results for wavelet orders 2 to 6 are plotted in Figure 4-2.

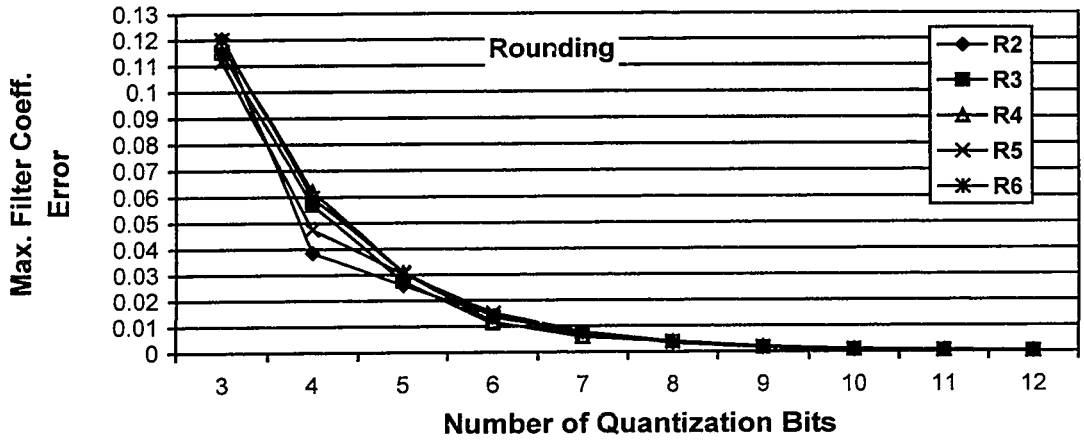
The errors for scenarios using the same number of quantization bits are reduced by approximately a factor of two if rounding, instead of truncation, is adopted by the direct structure filter. For instances, in order to have the maximum filter coefficient error equals 0.01 (i.e., -40dB) or less, only seven bits or more are required for rounding whereas eight bits or more are required for truncation. In the scenarios under investigation, the wavelet order does not affect the error at all, except for slight deviations observed for the 4-bit rounding and 3-bit truncation cases. It is because the Daubechies filter coefficients in various wavelet orders under investigation have similar magnitude.

In Table C-1 and Table C-2, the maximum magnitude errors $\left|\Delta H(e^{jw})\right|$ for both rounding and truncation cases decrease as the number of bits used to quantize the real Daubechies wavelet filter coefficients increases. Similar patterns are observed in the mean magnitude errors and they are plotted in Figure 4-3 and Figure 4-4 for Daubechies wavelet orders 2 to 6. According to these figures, the mean magnitude errors for passband and stopband decrease almost linearly for the truncation method. On the other hand, the mean magnitude errors for the rounding method remained unchanged or increased even when the number of quantization bit increased in several cases. For examples, the mean magnitude errors for the second wavelet order using rounding (Refer to **R2P** and **R2S** in

Figure 4-3) remained at –48.60dB (Passband) and –57.75dB (Stopband) for 6, 7 and 8-bit filter coefficients quantization. Furthermore, in order to achieve a mean magnitude error of 0.01 (i.e., -40dB) or less in both passband and stopband, only eight bits or more are required for the rounding method whereas nine bits or more were required for truncation.

The magnitude responses and quantization errors for the 3rd and the 6th Daubechies wavelet orders under investigation are illustrated in Figure 4-5 and Figure 4-6. Due to the large amount of information, only a selected set of scenarios (e.g., 3, 5, 7, 8, 10 and 12-bits filter coefficient quantization) are shown. In each figure, results related to rounding and truncation are displayed on the left and right columns respectively. Notice that the poorest magnitude responses observed in the figures are due to poor resolution provided by the 3-bit wordlength (represented by yellow curve) in quantizing the filter coefficients.

Generally, the higher the Daubechies wavelet order, the larger the mean magnitude error for the same number of coefficient quantization bits. One of the exceptions is in the passband truncation scenario where only relatively small deviations are observed among various wavelet orders. In addition, for the same number of quantization bits used, the mean magnitude errors of rounding method are usually 10 dB smaller than that of truncation method.



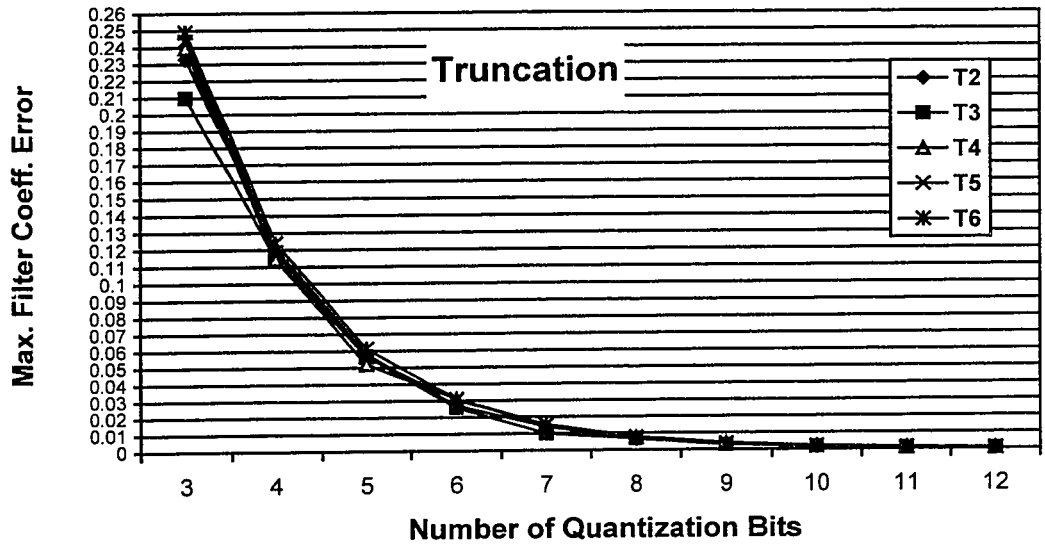


Figure 4-2 Plots of maximum filter coefficient error versus the number of quantization bits for the direct structure are plotted for (top) rounding and (bottom) truncation. In the legend, T2 refers to the maximum coefficient errors for the second Daubechies wavelet order, $N = 2$, using truncation (T); R3 refers to the maximum coefficient errors for the third Daubechies wavelet order using rounding (R).

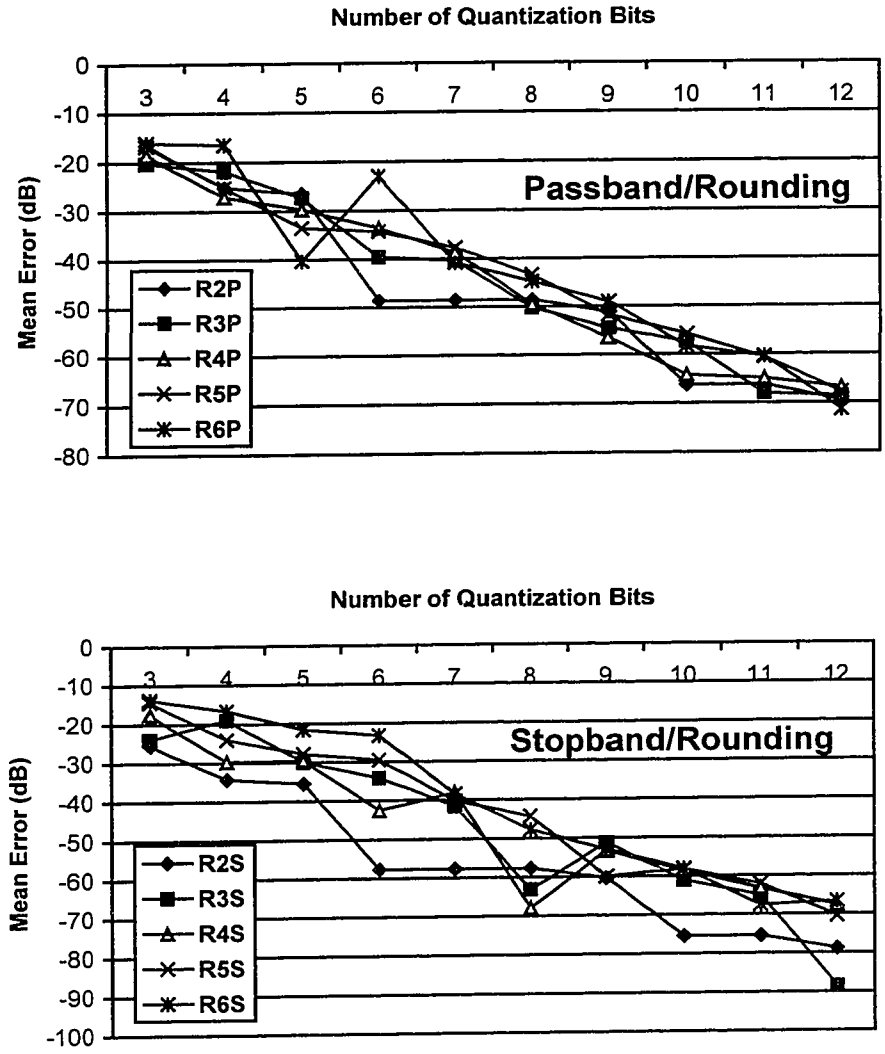


Figure 4-3 Plots of mean magnitude response error versus the number of quantization bits for the direct structure using rounding are plotted for (top) passband and (bottom) stopband. In the legend, R2P refers to the Passband P mean magnitude errors for the second order Daubechies wavelet, $N = 2$, using rounding R; R3S refers to the stopband S mean magnitude errors for third Daubechies wavelet order using rounding.

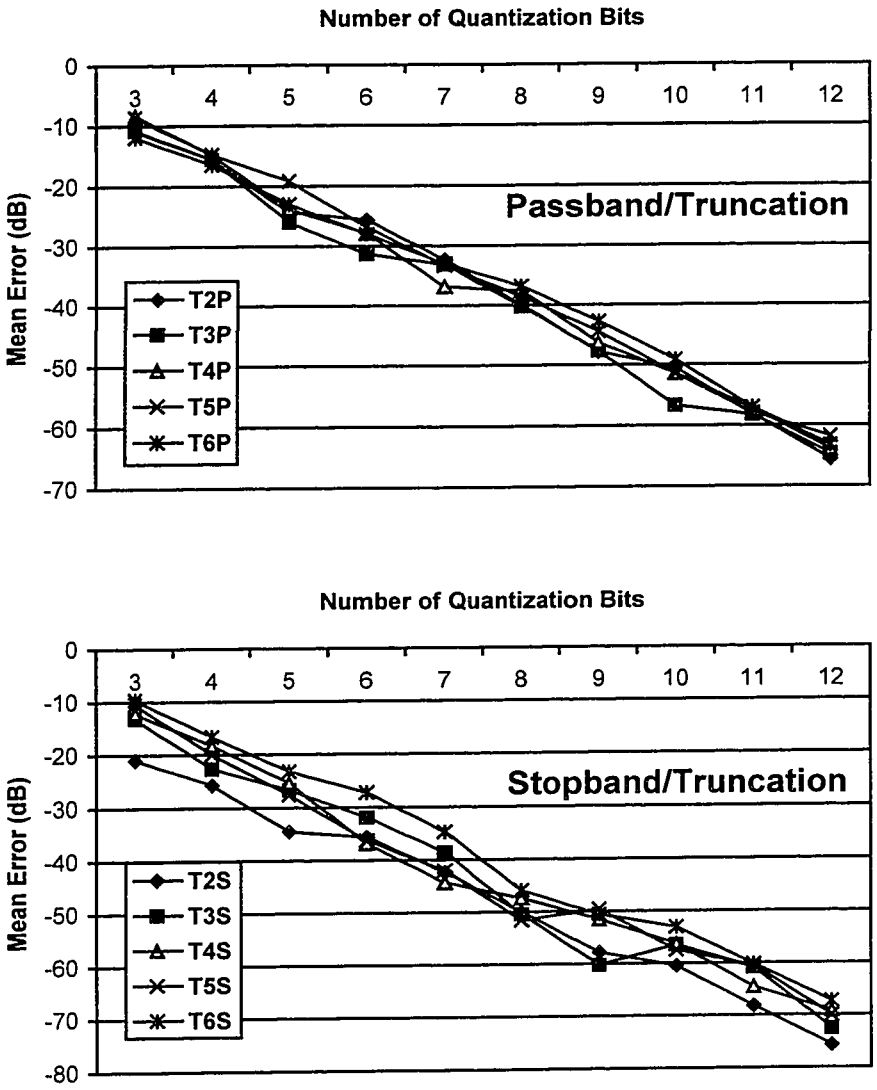


Figure 4-4 Plots of mean magnitude response error versus the number of quantization bits for the direct structure using truncation are plotted for (top) passband (bottom) stopband. In the legend, T2P refers to the Passband (P) mean magnitude errors for the second order Daubechies wavelet, $N = 2$, using truncation (T); T3S refers to the stopband (S) mean magnitude errors for third Daubechies wavelet order using truncation.

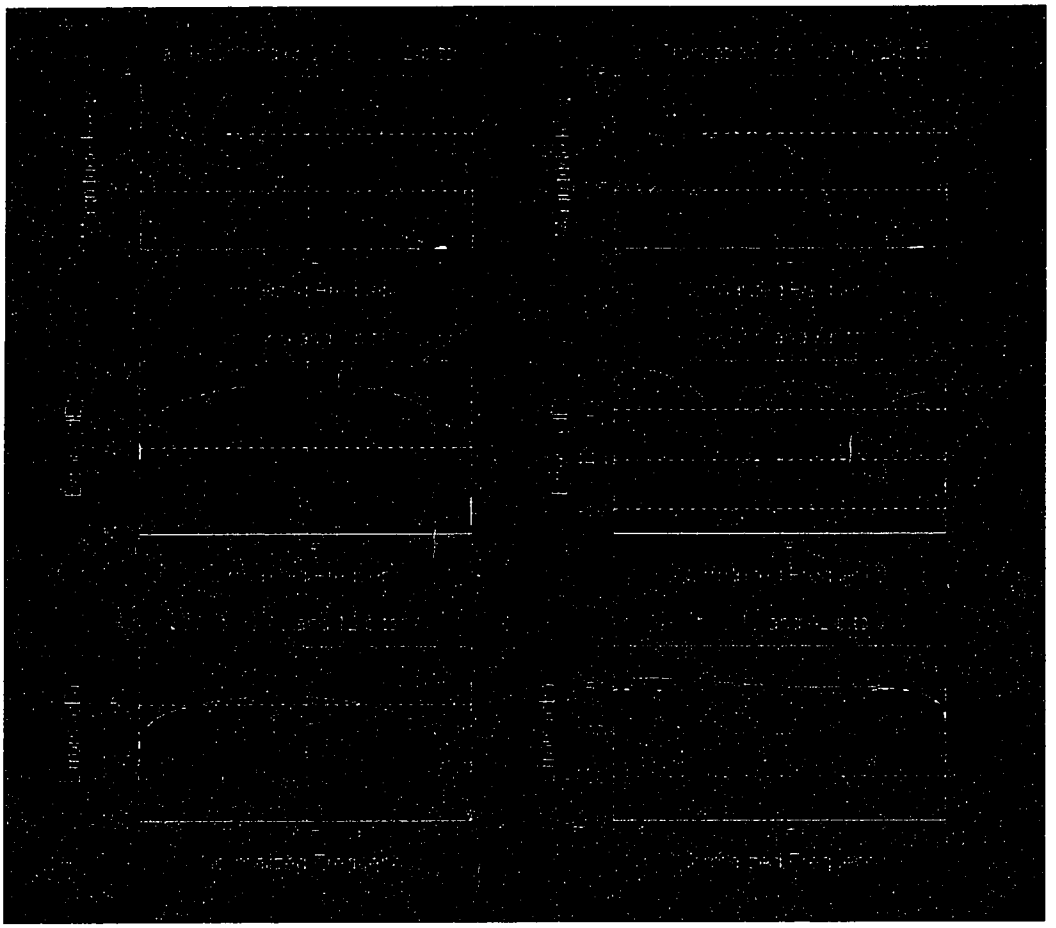


Figure 4-5 Wavelet Order *N*3: Quantized magnitude responses (3, 5, 7, 8, 10 & 12 quantization bits) for the direct structure using rounding (a) and truncation (d). Their deviations with the unquantized magnitude responses for rounding and truncation are plotted in (b), (c) and (e), (f) respectively.

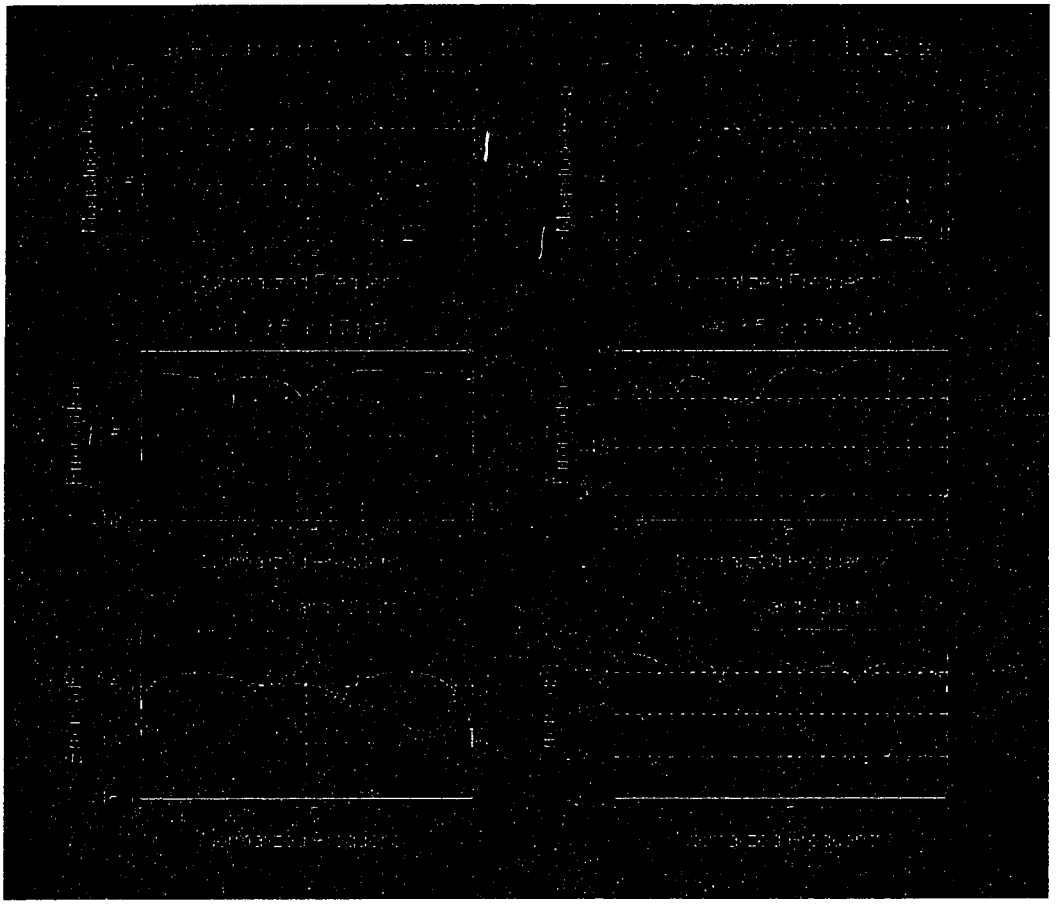


Figure 4-6 Wavelet Order *N*6: Quantized magnitude responses (3, 5, 7, 8, 10 & 12 quantization bit) for the direct structure using rounding (a) and truncation (d). Their deviations with the unquantized magnitude responses for rounding and truncation are plotted in (b), (c) and (e), (f) respectively.

4.2.2 Phase Responses for the Direct Structure

The magnitude response only provides half of the information of a system. In fact, the phase response also has a significant impact to the overall system response. It is possible to have two systems possess the same magnitude responses but different phase responses. On the contrary, the phase shift can totally distort the shape of the original waveform which was fed into the system. At the first glance, it seems desirable to design a system that has zero phase for each DTWT subband. Nonetheless, it is well known that nonzero phase responses are not attainable for causal frequency-selective filters. Hence, the next best thing is to have a linear phase or *near* linear phase system. A non-linear phase shift means that different frequency components of the input waveform arrive at the output of the system in different rate. Consequently, this *signal dispersion* in the DTWT filter bank may lead to inaccurate wavelet coefficients generation and signal reconstruction. In truth, a linear phase system is sufficient to avoid distortion since the lag is known and thus can be compensated. For instance, a linear phase N-tap QMF bank behaves just like a N-stage delay line.

Group delay τ can be used to quantify the linearity of the phase and it is defined as a continuous function of frequency:

$$\tau(\omega) = -\frac{d}{d\omega}\arg\left[H(e^{jw})\right], \tag{4.1.2.1}$$

where arg is referred to as the continuous phase. A constant group delay indicates a linear phase. Conversely, the phase nonlinearity is indicated by the deviation of group delay away from a constant. The decomposition or reconstruction QMF banks in the DTWT are *near* linear phase [12]. The phase response for the second wavelet order (N = 2) and its group delay are plotted in

Figure 4-7. Even though the direct structure filter's phase response appears fairly linear, it is evident that the group delay is not a constant.

In the project, the phase responses for direct structure filter for the Daubechies wavelet filter are simulated. The simulation results of the direct structure phase responses using rounding and truncation are tabulated in Table C-3. In the table, the mean and variance for both passband and stopband are also shown. Furthermore, phase responses for quantized filter coefficient using rounding and truncation are plotted for 2nd, 3rd, 4th and 6th orders Daubechies wavelet under investigation from Figure 4-8 to Figure 4-11. Visually, the rounded filter coefficients introduce less abrupt phase shift at the stopband region when comparing to the phase responses of the truncated coefficients.

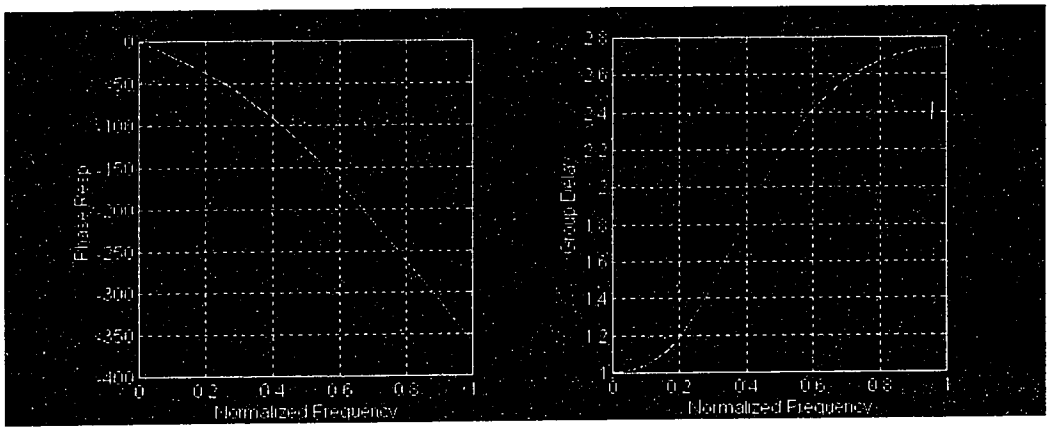


Figure 4-7 Phase responses (left) and group delay (right) for the 2nd order Daubechies wavelet ($N = 2$).

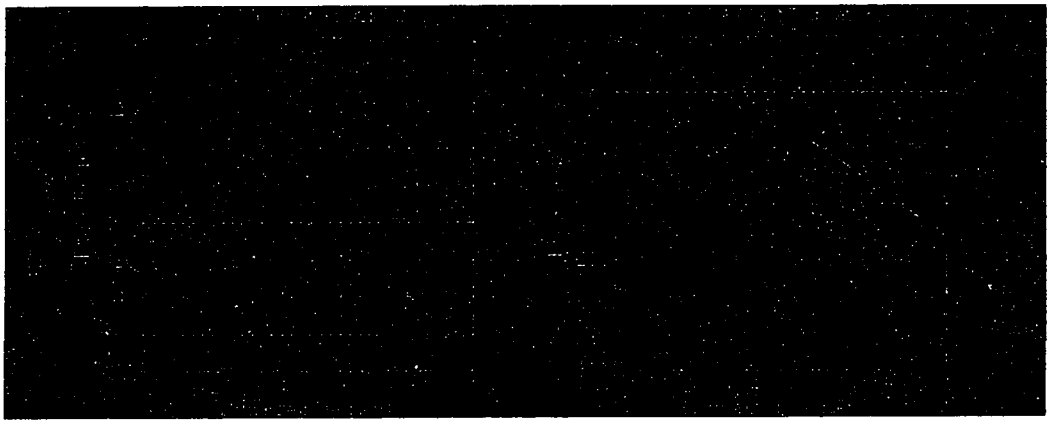


Figure 4-8 The direct structure phase responses of the second Daubechies wavelet order ($N = 2$). Selected set of quantization bits using rounding (left) and truncation (right) are used to show the finite wordlength effects.

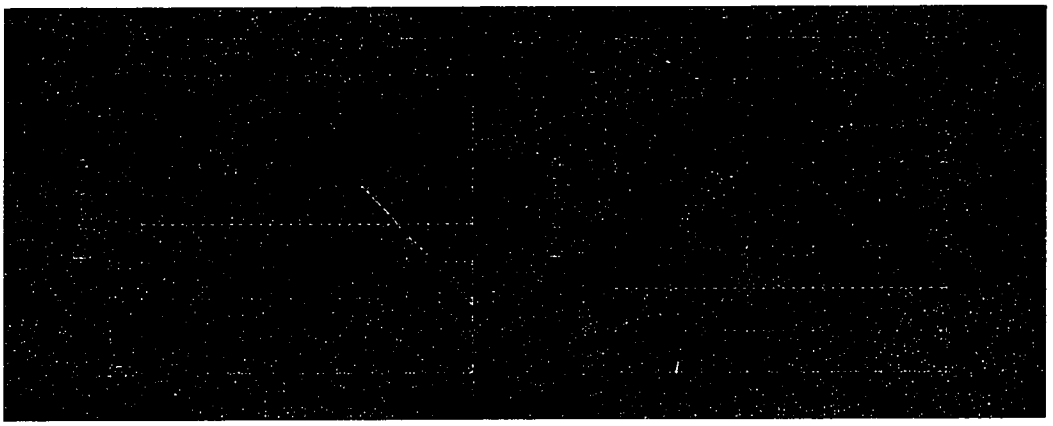


Figure 4-9 The direct structure phase responses of the third Daubechies wavelet order ($N = 3$). Selected set of quantization bits using rounding (left) and truncation (right) are used to show the finite wordlength effects.

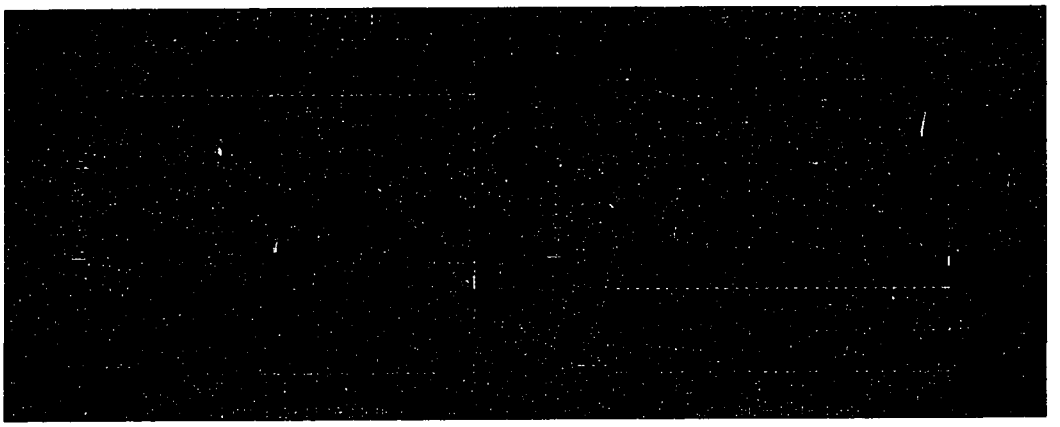


Figure 4-10 The direct structure phase responses of the fourth Daubechies wavelet order ($N = 4$). Selected set of quantization bits using rounding (left) and truncation (right) are used to show the finite wordlength effects.

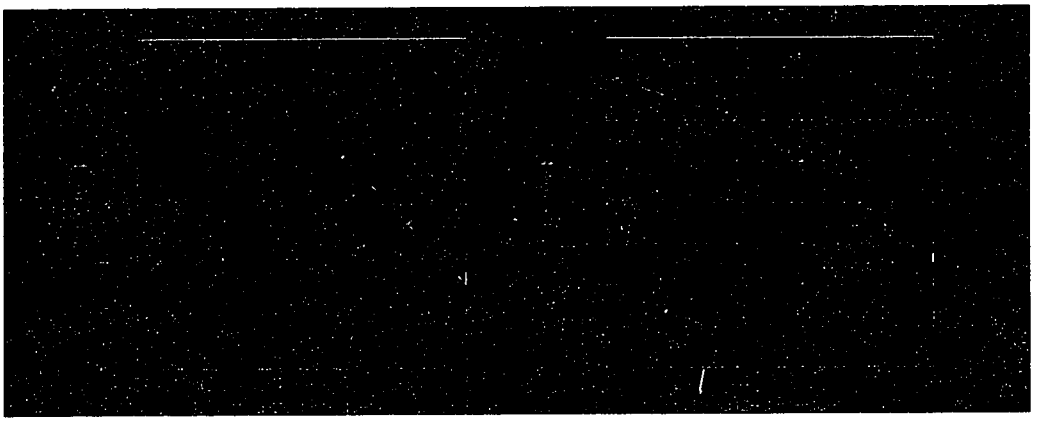


Figure 4-11 The direct structure phase responses of the sixth Daubechies wavelet order ($N = 6$). Selected set of quantization bits using rounding (left) and truncation (right) are used to show the finite wordlength effects.

4.3 Finite Wordlength Effects of Lattice Structure Filter

For comparison purpose, this project studies the coefficient effects on two different filters, namely the direct structure and the lattice structure. Traditionally, the lattice structure is not an efficient way to implement a FIR system because it generally requires more multipliers than the same system implemented by direct form (the number usually doubles when comparing to a similar direct form system). From a VLSI design point of view, multipliers take up a lot of floor space in a chip. Thus, the additional number of multipliers in the lattice structure often makes itself an unfavorable implementation choice. However, the DTWT belongs to a class of orthogonal filter that can be implemented in the lattice structure using only half the number of filter coefficients in an equivalent direct structure filter. The reduction gained translates into smaller chip size and reduced computation complexity. This property makes the lattice structure filter an attractive alternative to the direct structure filter for VLSI DTWT implementation.

4.4 DTWT Lattice Structure Model

The lattice structure filter was first adopted in the theory of autoregressive signal modeling [66]. Currently, this implementation structure is popular in adaptive prediction applications, such as speech analysis and synthesis. Lattice structure promises greater stability than direct IIR structure and fewer coefficients to compute than direct FIR structure in some cases [67]. Lattice structure can be used to realize both FIR and IIR filter but this project only studies the former case.

For the DTWT lattice implementation, each FIR filter pair in the MRA decomposition stage is implemented using a cascade of causal QMF lattice stages.

Similar to the direct structure filter implementation, the overall structure consists of cascades of causal lattice stages as shown in Figure 4-12. From the diagram, the following relationships are observed:

$$H_i(z) = H_{i-2}(z) - q_i z^{-2} G_{i-2}(z), \quad i = 3,5,... \tag{4.1.1}$$

$$G_i(z) = q_i H_{i-2}(z) + z^{-2} G_{i-2}(z), \quad i = 3,5,... \tag{4.1.2}$$

where the lattice coefficients q_i are the coefficients of the highest power of z^{-i} in $H_i(z)$. In Figure 2-3, the transfer functions for highpass filter (h) and lowpass filter (g) in the direct structure filter equal to that of $H_{max(i)}$ and $G_{max(i)}$ in the lattice structure filter respectively.

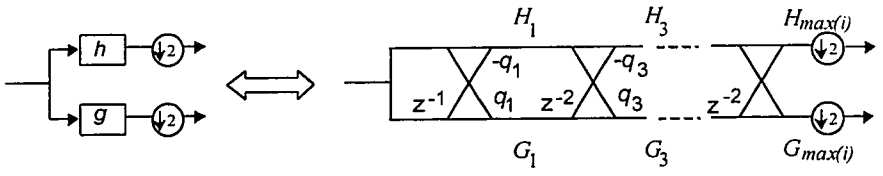


Figure 4-12 (Left) A two-channel analysis filter bank or a MRA decomposition stage; (Right) the QMF lattice equivalence.

Applying the polyphase interpretation of downsampling operation that $Y(z^M) = \frac{1}{M}\sum_{k=0}^{M-1} X(zW_M^k)$, all delay elements z^{-2} of the lattice structure in Figure 4-12 reduce to z^{-1} when the downsampling operations are moved to the beginning of the first lattice stage. The transformation is summarized in Figure 4-13. Furthermore, the downsampling operations and the delay element near the input of the structure can be realized as an input commutator.

For an orthogonal filter, its corresponding QMF lattice has a special characteristic [12, 68] which can reduce the amount of multiplication and addition operations when comparing to direct computation. Since all lattice coefficients q_i with even-valued indices are zero, the number of lattice coefficients equals $M/2$ (i.e., half the number required by direct form). The total number of multiplication and addition operations in a lattice stage for a N-length signal are then equal to MN (including both highpass and lowpass; refer to Section 3.2.1). As derived in Table 2-1, the next lattice stage only requires half of the multiplication and addition operations performed in the previous stage in a MRA-based DTWT system. According to equation 2.4.1.6, the upper bound of a complete MRA decomposition for the lattice structure equals $2C$ (i.e., $2C = 2 \times [MN$ multiplications + MN additions]). Comparing the results to that of the direct structure filter, both multiplication and addition operations are reduced almost by half.

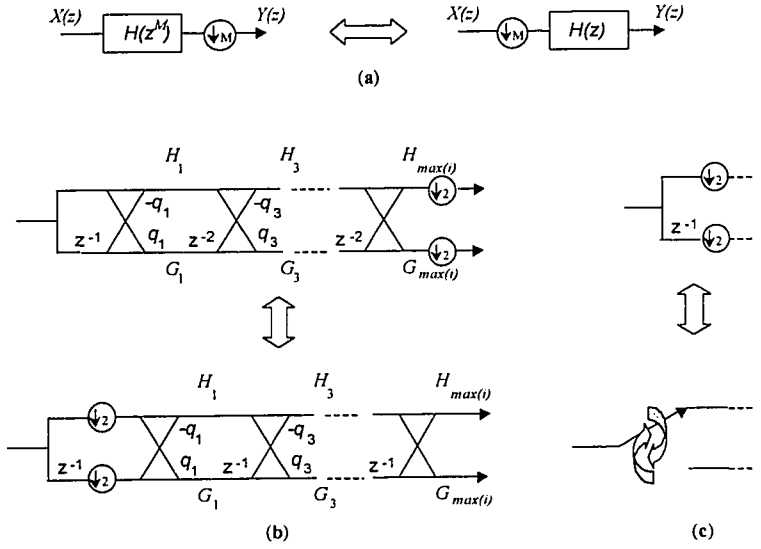


Figure 4-13 (a) Downsampling equivalence; (b) New lattice structure with less delay element; (c) Input commutator.

Above all, Vaidyanathan and Hoang have shown that the paraunitary property is inherent in the QMF lattice structure. Consequently, the DTWT lattice implementation is robust in the sense that the perfect reconstruction property is preserved in spite of coefficient quantization [69]. Furthermore, the same hardware can be used for both analysis (decomposition) and synthesis (reconstruction) by simply changing the processing order from the output to the input. The lattice coefficient used in this project is listed in Table 4-3 along with their corresponding Daubechies wavelet family. The MATLAB function that converts filter coefficients from direct structure format to lattice structure format (and vice versa) are listed in B.8 (B.10). Noticed that the scaling factors β for each lattice stage can be combined into one single factor during implementation.

Daubechies family	Daubechies filter coefficients	QMF lattice coefficients
$N = 2$	$h_2(0)$ = 0.482962913145 $h_2(1)$ = 0.836516303738 $h_2(2)$ = 0.224143868042 $h_2(3)$ = -0.129409522551	q_1 = -1.732050807568 q_3 = 0.267949192430 β = $h_2(0)$
$N = 3$	$h_3(0)$ = 0.332670552950 $h_3(1)$ = 0.806891509311 $h_3(2)$ = 0.459877502118 $h_3(3)$ = -0.135011020010 $h_3(4)$ = -0.085441273882 $h_3(5)$ = 0.035226291882	q_1 = -2.425497243932 q_3 = 0.546096402965 q_5 = -0.105889419937 β = $h_3(0)$
$N = 4$	$h_4(0)$ = 0.230377813309 $h_4(1)$ = 0.714846570553 $h_4(2)$ = 0.630880767930 $h_4(3)$ = -0.027983769417 $h_4(4)$ = -0.187034811719 $h_4(5)$ = 0.030841381836 $h_4(6)$ = 0.032883011667 $h_4(7)$ = -0.010597401785	q_1 = -3.102931485825 q_3 = 0.810931344496 q_5 = -0.259293881631 q_7 = 0.046000097113 β = $h_4(0)$
$N = 5$	$h_5(0)$ = 0.160102397974 $h_5(1)$ = 0.603829269797 $h_5(2)$ = 0.724308528438 $h_5(3)$ = 0.138428145901 $h_5(4)$ = -0.242294887066 $h_5(5)$ = -0.032244869585 $h_5(6)$ = 0.077571493840 $h_5(7)$ = -0.006241490213 $h_5(8)$ = -0.012580751999 $h_5(9)$ = 0.003335725285	q_1 = -3.771519212149 q_3 = 1.063943426476 q_5 = -0.424829277744 q_7 = 0.133184545637 q_9 = -0.020834948928 β = $h_5(0)$
$N = 6$	$h_6(0)$ = 0.111540743350 $h_6(1)$ = 0.494623890398 $h_6(2)$ = 0.751133908021 $h_6(3)$ = 0.315250351709 $h_6(4)$ = -0.226264693965 $h_6(5)$ = -0.129766867567 $h_6(6)$ = 0.097501605587 $h_6(7)$ = 0.027522865530 $h_6(8)$ = -0.031582039318 $h_6(9)$ = 0.000553842201 $h_6(10)$ = 0.004777257511 $h_6(11)$ = -0.001077301085	q_1 = -4.434468308238 q_3 = 1.308336039621 q_5 = -0.589618123372 q_7 = 0.243014597224 q_9 = -0.069999878205 q_{11} = 0.009658363865 β = $h_6(0)$

Table 4-3 Daubechies filter coefficients and their corresponding lattice filter coefficients for $N = 2$ to 6.

4.5 Simulation Results

This section presents the simulation results for the finite wordlength effects of lattice structure filter. To illustrate the non-ideal effects due to filter coefficient quantization under different round-off methods, the unquantized and quantized (8-bit and 12-bit) filter coefficients for 3rd and 6th orders of the Daubechies highpass filter are tabulated in Table 4-4.

	Daubechies family	QMF lattice coefficients	8-bit representation	12-bit representation
Truncation	$N = 3$	$q_1 =$ -2.425497243932	-2.4375	-2.42578125
		$q_3 =$ 0.546096402965	5.3123×10^{-1}	$5.44921875 \times 10^{-1}$
		$q_5 =$ -0.105889419937	-1.25×10^{-1}	$-1.07421875 \times 10^{-1}$
	$N = 6$	$q_1 =$ -4.434468308238	-4.4375	-4.4375
		$q_3 =$ 1.308336039621	1.25	1.3046875
		$q_5 =$ -0.589618123372	-6.25×10^{-1}	$-5.8984375 \times 10^{-1}$
		$q_7 =$ 0.243014597224	1.875×10^{-1}	2.421875×10^{-1}
		$q_9 =$ -0.069999878205	-1.25×10^{-1}	-7.03125×10^{-2}
		$q_{11} =$ 0.009658363865	0	7.8125×10^{-3}
Rounding	$N = 3$	$q_1 =$ -2.425497243932	-2.4375	-2.42578125
		$q_3 =$ 0.546096402965	5.3125×10^{-1}	5.46875×10^{-1}
		$q_5 =$ -0.105889419937	-9.375×10^{-2}	$-1.0546875 \times 10^{-1}$
	$N = 6$	$q_1 =$ -4.434468308238	-4.4375	-4.43359375
		$q_3 =$ 1.308336039621	1.3125	1.30859375
		$q_5 =$ -0.589618123372	-5.625×10^{-1}	$-5.8984375 \times 10^{-1}$
		$q_7 =$ 0.243014597224	2.5×10^{-1}	2.421875×10^{-1}
		$q_9 =$ -0.069999878205	-6.25×10^{-1}	-7.03125×10^{-2}
		$q_{11} =$ 0.009658363865	0	7.8125×10^{-3}

Table 4-4 Unquantized and Quantized (8-bit and 12-bit) Coefficients for 3rd and 6th orders of the Daubechies highpass filter (Lattice Structure Implementation)

4.5.1 Magnitude Responses for the Lattice Structure

The same investigations performed for the direct structure filter are repeated for the lattice structure. The simulation results for the lattice structure magnitude using rounding and truncation are tabulated in Table C-4 and Table C-5 of Appendix C respectively.

Since the wordlength limitation and the number representation are the same, the maximum coefficient errors $\Delta h[n]|_{max}$ for both direct and lattice structures using rounding and truncation are, as expected, exactly the same in our simulations as expected. That is, the errors decreased as the number of bits used to quantize the Daubechies wavelet filter coefficients increased.

Unlike the direct structure filter that showed smaller mean magnitude errors with rounded coefficients, the mean magnitude errors for the lattice structure filter remained approximately the same in both passband and stopband regions no matter whether rounding or truncation was used. In fact, when compared to the results for the direct structure filter, larger differences (e.g., ranged from 15 to 30dB) in the mean magnitude errors are observed among small and large wavelet orders in the lattice cases under equivalent conditions. For example, the mean magnitude errors of the second and the sixth Daubechies wavelet orders with 7-bit coefficient quantization in the passband rounding scenarios are –70.58dB and –38.97dB respectively. Furthermore, in order to achieve a mean magnitude error of 0.01 (i.e., -40dB) or less in the passband truncation scenarios for the lattice structure filter, only four bits or more are required for the 2nd wavelet order whereas ten bits or more were required for the six wavelet order.

The step-wise curve pattern for the direct structure using rounding is observed in all scenarios of the lattice structure as shown in Figure 4-14 and Figure 4-15. Thus, the mean magnitude errors for the lattice structure filter remained unchanged or increased even though the number of quantization bits increased. For example, fairly constant mean magnitude errors are observed for the second wavelet order in all lattice scenarios with 4, 5, 6 and 7-bit as well as 8, 9 and 10-bit coefficient quantizations.

For comparison purposes, the magnitude errors for the 3rd and the 6th Daubechies wavelet orders using lattice structure filter are depicted in Figure 4-16 and Figure 4-17. Again, the same selected set of scenarios (e.g., 3, 5, 7, 8, 10 and 12-bits filter coefficient quantization) were shown. In these figures, results related to rounding and truncation are displayed on the left and right columns respectively.

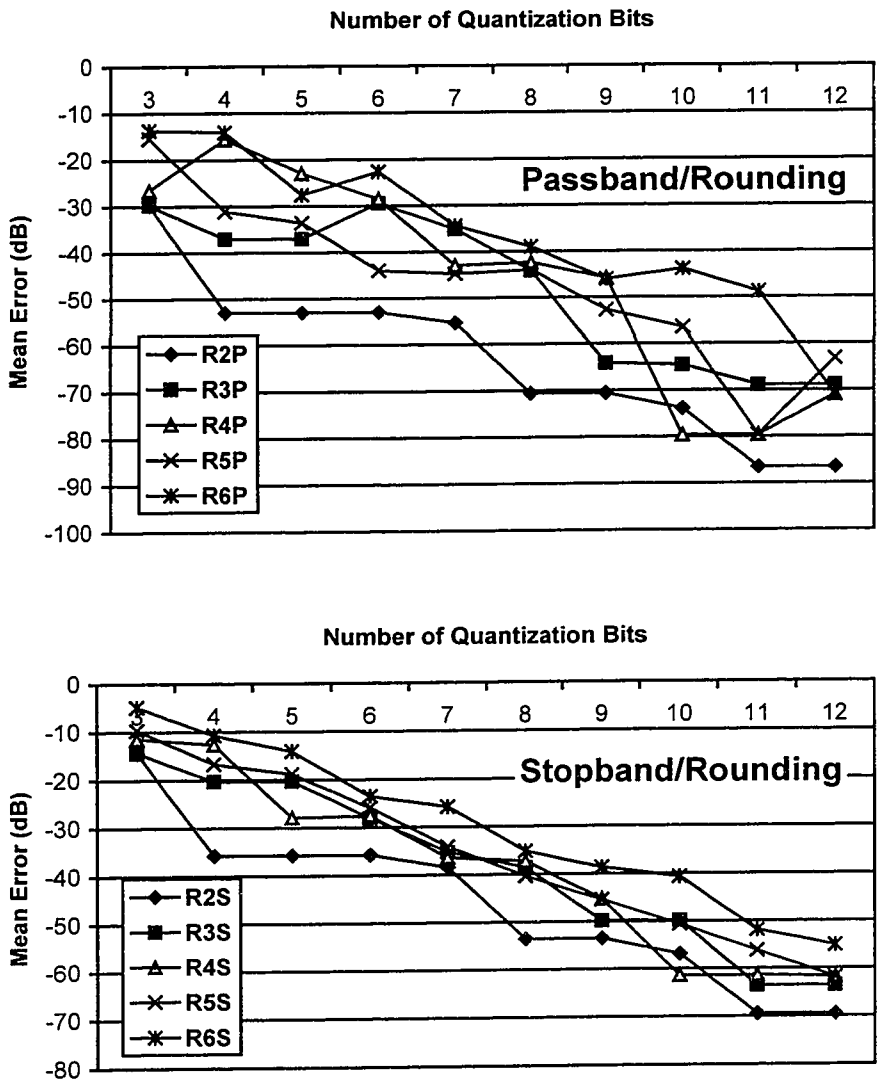


Figure 4-14 Plots of mean magnitude response error versus the number of quantization bits for the lattice structure using rounding are plotted for (top) passband and (bottom) stopband. In the legend, R2P refers to the Passband P mean magnitude errors for the second order Daubechies wavelet, $N = 2$, using rounding (P); R3S refers to the stopband (S) mean magnitude errors for third Daubechies wavelet order using rounding.

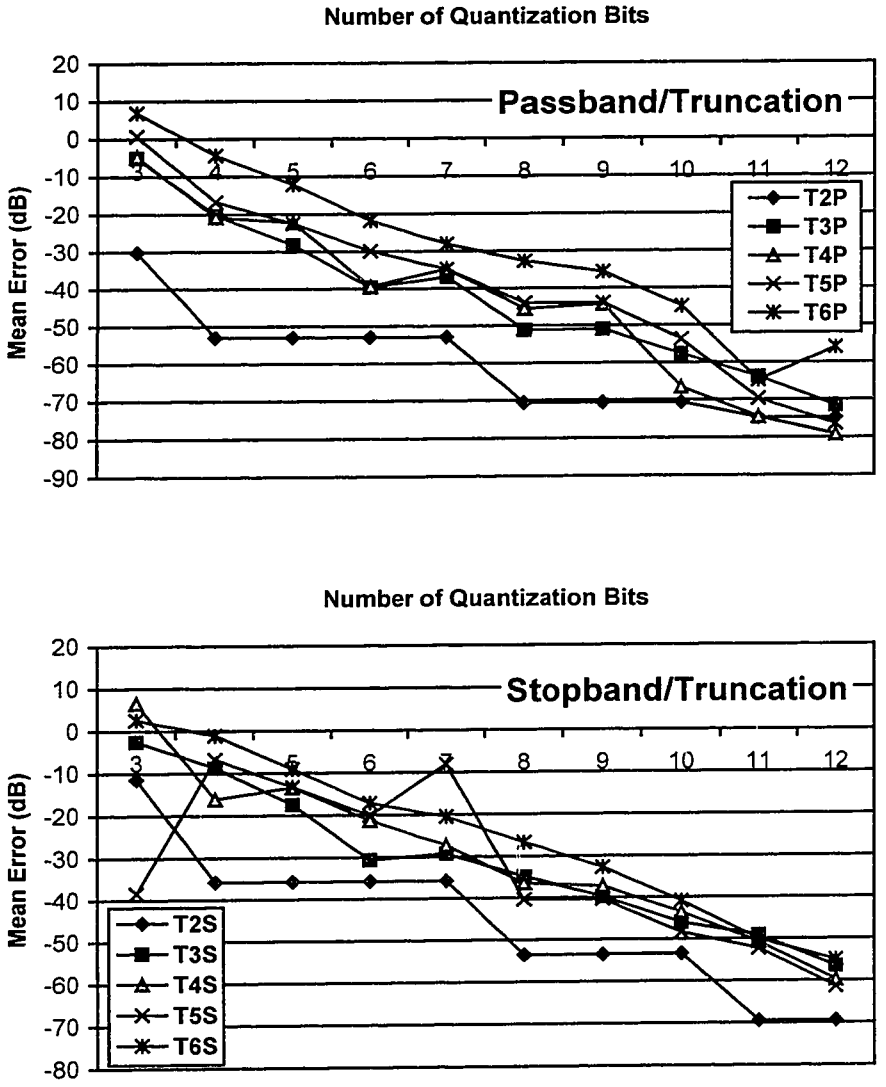


Figure 4-15 Plots of mean magnitude response error versus the number of quantization bits for the lattice structure using truncation are plotted for (top) passband and (bottom) stopband. In the legend, T2P refers to the Passband (P) mean magnitude errors for the second order Daubechies wavelet, $N = 2$, using truncation (P); T3S refers to the stopband (S) mean magnitude errors for third Daubechies wavelet order using truncation.

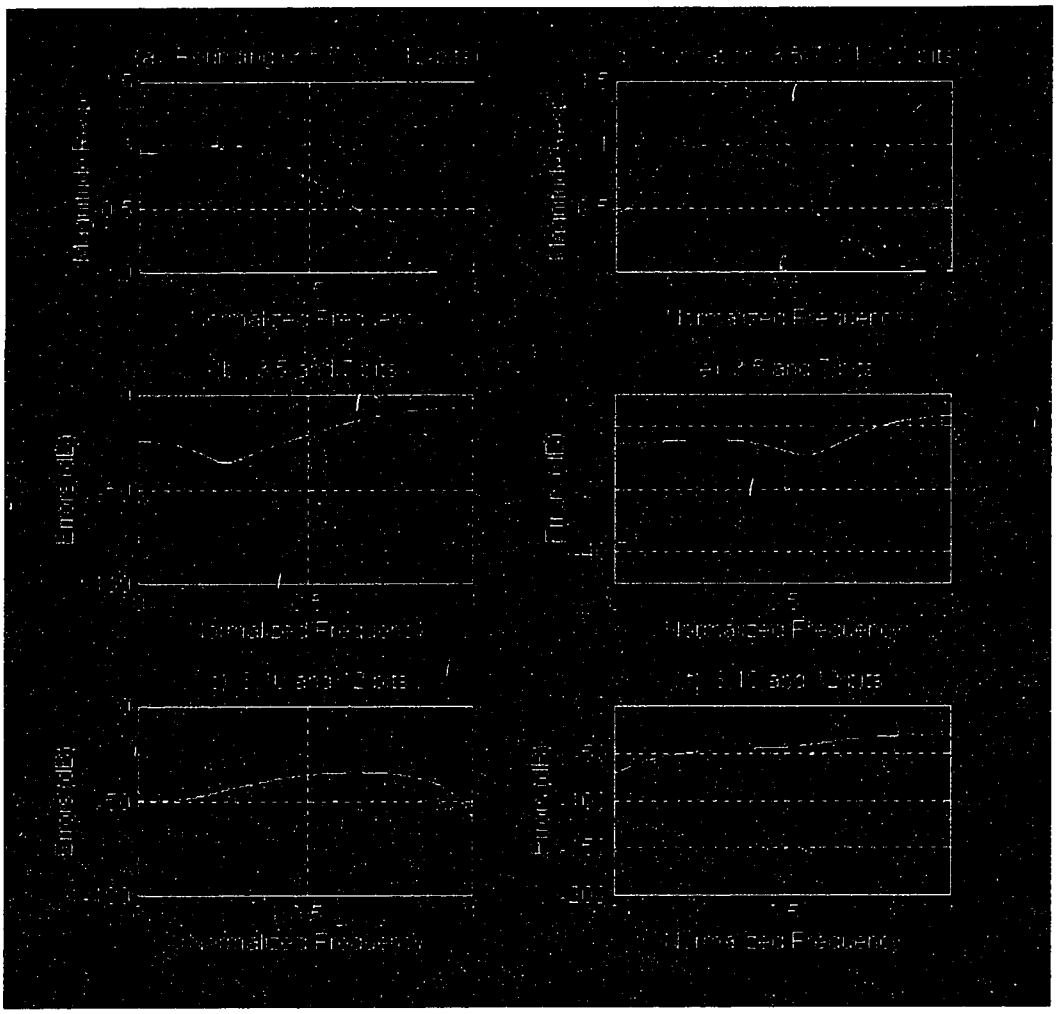


Figure 4-16 Wavelet Order *N*3: Quantized magnitude responses (3, 5, 7, 8, 10 & 12 quantization bits) for the lattice structure using rounding (a) and truncation (d). Their deviations with the unquantized magnitude responses for rounding and truncation are plotted in (b), (c) and (e), (f) respectively.

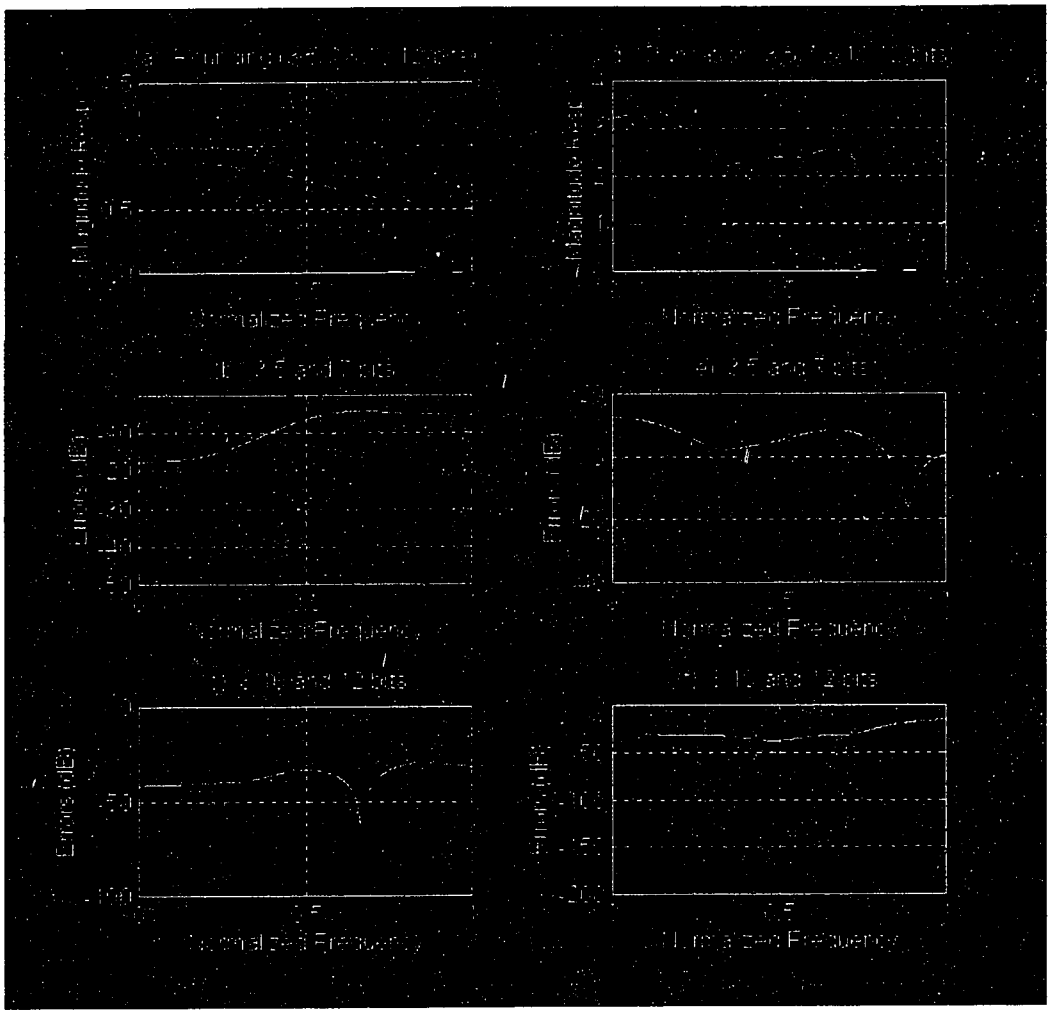


Figure 4-17 Wavelet Order *N*6: Quantized magnitude responses (3, 5, 7, 8, 10 & 12 quantization bits) for the lattice structure using rounding (a) and truncation (d). Their deviations with the unquantized magnitude responses for rounding and truncation are plotted in (b), (c) and (e), (f) respectively.

4.5.2 Phase Responses for the Lattice Structure

The simulation results of the phase responses for lattice structure filter using rounding and truncation are tabulated in Table C-6. In the table, the mean and variance for both passband and stopband are also shown. Furthermore, phase responses for quantized filter coefficient using rounding and truncation are plotted for 2^{nd}, 3^{rd}, 4^{th}, 6^{th} order Daubechies wavelet under investigation from Figure 4-18 to Figure 4-21. The linear phase property is observed in these figures. Similar to the direct structure filter, all abrupt phase shifts of lattice structure occur mainly in the stopband regions. Moreover, the lattice structure performances, in terms of phase linearity and degree of phase shift, for both rounding and truncation vary with the quantization bits. Except for the 2^{nd} and the 3^{rd} wavelet orders (red and blue lines in Figure 4-18 to Figure 4-21), the rounded filter coefficients introduce less abrupt phase shift in the stopband region when compared to the phase responses of their direct structure counterparts with the same number of quantization bits. Above all, except for the second wavelet order, the truncated filter coefficients introduce "smoother" phase shift (i.e., smaller group delay) than their direct structure counterparts under most investigated scenarios. As a result, for phase-sensitive applications such as radar and image processing, the lattice structure is a better implementation architecture than the direct structure based on our study. Nonetheless, there is no general phase error pattern observed. Hence, the decision to adopt either rounding or truncation in quantizing filter coefficients has to be made on a case by case basis.

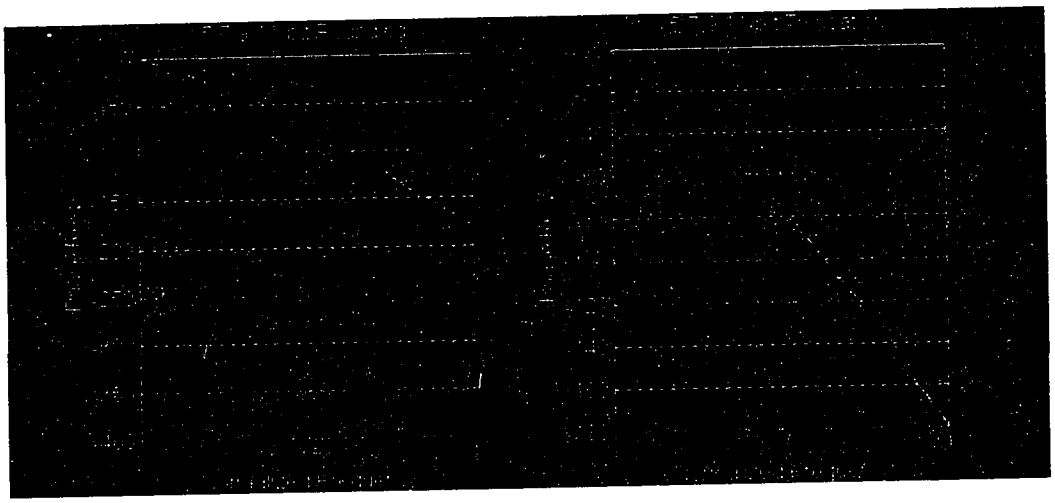


Figure 4-18 The lattice structure phase responses of the second Daubechies wavelet order ($N = 2$). Selected set of quantization bits using rounding (left) and truncation (right) are shown to show the finite wordlength effects.

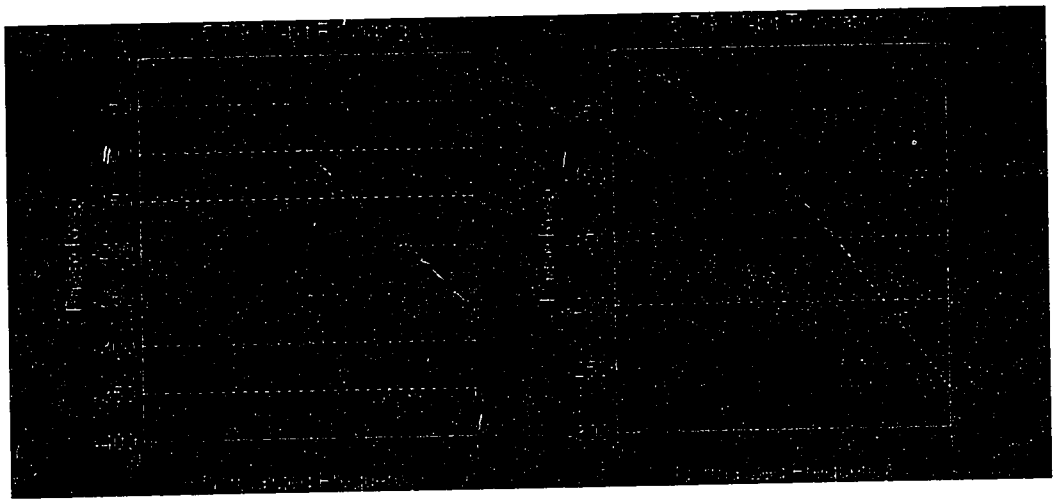


Figure 4-19 The lattice structure phase responses of the third Daubechies wavelet order ($N = 3$). Selected set of quantization bits using rounding (left) and truncation (right) are shown to show the finite wordlength effects.

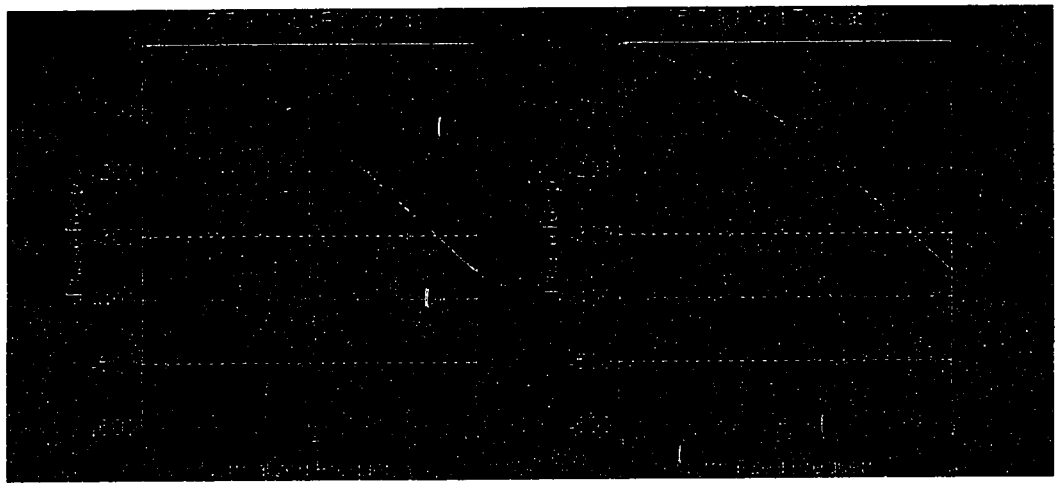


Figure 4-30 The lattice structure phase responses of the fourth Daubechies wavelet order ($N = 4$). Selected set of quantization bits using rounding (left) and truncation (right) are shown to show the finite wordlength effects.

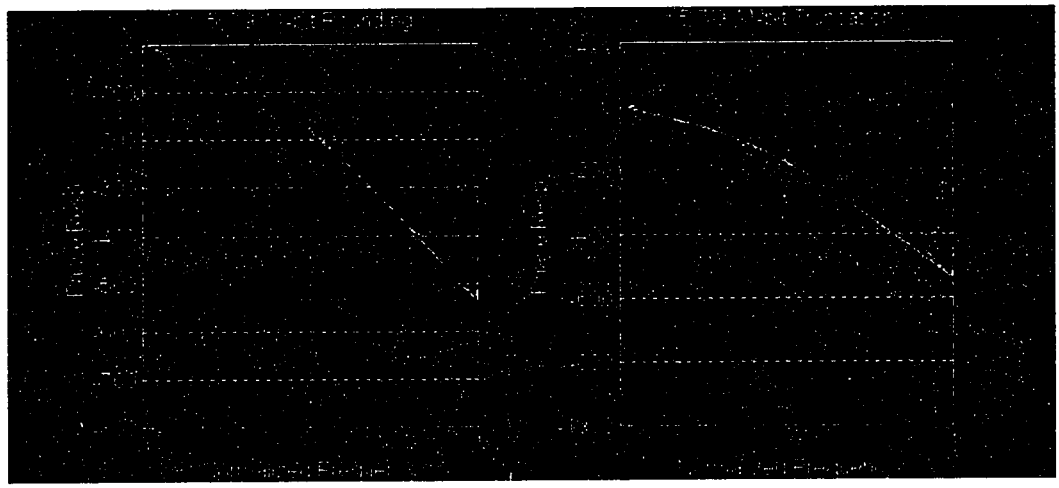


Figure 4-31 The lattice structure phase responses of the sixth Daubechies wavelet order ($N = 6$). Selected set of quantization bits using rounding (left) and truncation (right) are shown to show the finite wordlength effects.

4.6 Filter Type Recommendation for VLSI DTWT Implementation

The common challenge in implementing a VLSI MRA-based DTWT system is the complexity of control circuit/scheduling algorithm (References). The control circuit is essential to share one set of multipliers between the lowpass and highpass filter operations in a decomposition (reconstruction) stage. The lattice structure filter is superior to direct structure filter in implementing the DTWT system from both structural and performance point of views.

From the structural viewpoint, lattice structure filter only requires half of the filter coefficients used by direct structure filters to implement a DTWT stage with the same wavelet order. With the reduction of computational complexity in lattice-based orthogonal QMF, the lattice structure f ilter is an appealing alternative to implement VLSI DTWT. In implementing a device in which decomposition (signal out) and reconstruction stages (signal in) co-locate on the same side (e.g., an ADSL modem), there are special properties. For example, a decomposition stage can become a reconstruction stage by simply switching the input and output point allowing hardware sharing with a relatively simpler control circuit/scheduling algorithm.

From the performance viewpoint, lattice structure filter generally introduces less non-ideal effects than direct structure filter in both the phase and magnitude response in the similar setting (i.e., same filter coefficient quantization and same wavelet order).

Based on the above observations, lattice structure filter is preferred to direct structure in VLSI DTWT implementation. In addition, unlike the direct structure filter, there is no significant improvement by using rounding over truncation in all scenarios we have investigated. Since it requires less hardware to implement

truncation than rounding, truncation should be adopted as the rounding method in implementing DTWT system when lattice structure filter is used.

Chapter 5

CONCLUSIONS

5.1 Summary and Conclusions

The wavelet transformation has attracted a lot of attention from researchers in both the commercial world as well as the academic community. WT not only has huge potentials to be employed in many applications, but it also provides a framework to link theories and concepts developed individually across different academic disciplines. WT (i.e., subband coding) has shown to be an excellent candidate to represent a signal as compactly yet as accurately as possible to accommodate limitations in transmission or storage. This will by all means lead to a high demand for hardware implementations of DTWT systems in the near future. Currently, the most economical way to implement DTWT system in volume is by VLSI.

The main goal of this project is to examine the finite wordlength effects on a MRA-based VLSI DTWT system. The results of this study will provide essential information for ASIC designers to account for the non-ideal effects that can alter the desired performance objectives. A precise analysis of quantization effects is generally not required in practical applications. The most effective approach, which is adopted in this project, is to simulate the system and measure the

performance. This goal is achieved by the completion of a versatile MATLAB DTWT model that allows user to simulate the overall DTWT system response under the influence of quantization errors in filter coefficient and/or internal computation. The scope of this project was restricted to the Daubechies wavelet family. However, the model can be easily modified to accept filter coefficients based on other classic mother wavelets. Further, we focus our study in short filters length and 3-decomposition stage as they are both preferred for rapid applications, such as data transmission and real-time image processing. Above all, we demonstrated the potential of using the MATLAB DTWT model in studying selective subband reconstruction with the presence of non-ideal effects.

Another goal of this project is to compare and contrast the performances and other major design concerns of using either direct structure filter or the lattice structure filter as the building block for a MRA-based DTWT system. We developed conversion programs to obtain the lattice structure filter coefficient for Daubechies wavelet family. We observed that that lattice structure filter has a superior phase response to that of direct structure in most cases in our study. Moreover, lattice structure demands relatively less computation than direct structure in the same filter length and DTWT configuration. Therefore, lattice structure filter is a better alternative to build a MRA-based DTWT system.

Wavelet provides excellent research opportunities as well as a different perspective to view matters and to solve problems. In the following, an outline of the future work and directions for this project is presented.

5.2 Future Work and Directions

This project focused on the finite wordlength effects of DTWT using minimal length Daubechies QMF filters. The results of this project will be verified with hardware prototype built with programmable DSP. Further extension of this

project can be divided into three categories. First of all, we can explore linear transform bases other than the Daubechies wavelet basis which we have studies here. Secondly, we can improve the accuracy of our MATLAB DTWT models. Finally, we should ask ourselves the golden question:" Could it be done in a different and better way?"

The scope of this study only investigated the short Daubechies wavelet. As was previously mentioned, it is possible to improve the performance of the DTWT by using different transformation bases [70] for specific applications. Some possible candidates for further study are linear phase biorthogonal bases, *wavelet packets*, and *uneven* filter bank [71]. A subset of the linear phase biorthogonal bases is also capable of perfect reconstruction. They can be applied in phase sensitive applications where the Daubechies wavelet bases do not perform well. For wavelet packets [13], computational complexity is sacrificed for better frequency localization. In [71], filter bank with asymmetrical analysis and synthesis filters are claimed to have good performance in image processing. Furthermore, the study can be extended to study new filter design such as pseudo-QMF banks [72].

Filter convolution made up of series of multiplication and addition operations. The computational complexity and speed of VLSI digital filters using fixed-point arithmetic is normally dominated by adders used in the implementation of the multipliers. Performance of the DTWT can be improved by using better multiplier designs, such as using Booth multiplier recoding and carry-save addition together to reduce internal operations [73] as in many high-performance computers. The MATLAB models developed in this project can be extended to included various multiplier designs, such as the minimum-adder FIR filters design [74], fast VLSI adder design [75] and Booth encoded multiplier using Wallace trees [76].

To completely avoid finite wordlength effects in wavelet transform, an integer arithmetic wavelet transform that map integers to integers are required. Recently, Calderbank, Daubechies, Sweldens and Yeo [77] have suggested a new wavelet transform algorithm that maps integers to integers for lossless image coding. However, the internal computations are still performed with floating point arithmetic. In fact, the feasibility of having a complete integer arithmetic wavelet transform is still an open question and a complete treatment remains a topic for further investigation.

Appendix A

A.1 Signal Basis

A signal f can be expressed as a linear combination of another elementary signal φ with a coefficient sequence α

$$f = \sum_i \alpha_i \varphi_i . \tag{A.1.1}$$

The elementary signal $\{\varphi_i\}$ is only considered as a complete basis for a space S (e.g., the Euclidean n-space $\Re^n$) if all signals in the space (i.e., $f \in S$) can be expanded as in equation (A.1.1). For any complete basis, there will exist a dual basis set $\{\tilde{\varphi}_i\}$ from which the coefficient sequences can be calculated. The inner product relationships between the dual elementary signal set and the original signal are shown in (A.1.2) for real continuous functions and in (A.1.3) for discrete sequences.

$$\alpha_i = \langle \tilde{\varphi}_i, f \rangle = \int \tilde{\varphi}_i(t) f(t) dt \tag{A.1.2}$$

$$\alpha_i = \langle \tilde{\varphi}_i, f \rangle = \sum_n \tilde{\varphi}_i[n] f[n] \tag{A.1.3}$$

The choice of basis depends on individual application. In fact, we can consider equation (A.1.1) as performing a transformation and the selected basis provides a new viewpoint for the original signal. A good basis for data compression may be one that allows compact representation of the original signal as well as perfect reconstruction from the coefficient set. For real-time application, a good basis

may be one that allows fast computation. In many cases, an optimum basis is picked in view of the multiple constraints posted by an application.

A.2 Fourier Basis and Localization

A.2.1 Fourier Series Representation for periodic signals

In the early eighteenth century, Joseph Fourier introduced a new way to interpret function. He asserted that any 2π-periodic function $f(t)$ is the sum of its Fourier series which consist of harmonic sines and cosines (i.e., $e^{jk\omega_0 t}$) as the basis function. For example,

$$f(t) = \sum_{k=-\infty}^{\infty} \alpha_k e^{jk\omega_0 t} . \qquad \text{(A.2.1.1)}$$

The coefficients $\{\alpha_k\}$ are often called Fourier series coefficients or the spectral coefficient of $f(t)$. These coefficients measure the contribution of signal $f(t)$ at different harmonic components ($k\omega_0$) and they are obtained as

$$\alpha_k = \frac{1}{T_0} \int_{T_0} f(t) e^{jk\omega_0 t} dt . \qquad \text{(A.2.1.2)}$$

The synthesis equation (A.2.1.1) and the analysis equation (A.2.1.2) provide tools to transform data from the time domain into the frequency domain and vice versa. In fact, Fourier basis allows efficient computation of convolution. Since then, Fourier transform and Fourier's representation of functions as a superposition of sines and cosines has become ubiquitous in numerous science, mathematics and engineering fields. The discrete-time version of the synthesis (A.2.1.3) and analysis equations (A.2.1.4) is shown below. Signal $x[n]$ is a discrete-time sequence and it is periodic with a period N (i.e., $x[n] = x[n+N]$):

$$x[n] = \sum_{k=\langle N \rangle} \alpha_k e^{jk(\frac{2\pi}{N})n}, \tag{A.2.1.3}$$

$$\alpha_k = \frac{1}{N} \sum_{n=\langle N \rangle} x[n] e^{-jk(\frac{2\pi}{N})n}. \tag{A.2.1.4}$$

A.2.2 Fourier Transform for Aperiodic Signal

The result of Fourier representation of periodic signals can be extended to include aperiodic signals as well. In short, we can construct a periodic signal for which each period is identical to the aperiodic signal. We then consider the aperiodic signal as the limit of the "constructed" periodic signal when the period is arbitrarily large. Using these facts, we obtain the synthesis equation or the inverse Fourier transform (A.2.2.1), as well as the analysis equation or the Fourier transform (A.2.2.2) for aperiodic signal $f(t)$ as

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(\omega) e^{j\omega t} d\omega, \tag{A.2.2.1}$$

$$F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{-j\omega t} dt. \tag{A.2.2.2}$$

$F(\omega)$ is commonly referred to as the spectrum of $f(t)$ because it gives information regarding the composition of sinusoidal elementary signals at various frequencies. In equations (A.2.1.1) and (A.2.1.3), the amplitudes of complex coefficient $\{\alpha_k\}$ occur at a discrete set of harmonically related frequencies $(k\omega_0)$ whereas these coefficients occur as a continuum of frequencies with amplitudes equal to $F(\omega)(d\omega / 2\pi)$ for aperiodic signal. The synthesis equation (the inverse Fourier transform) (A.2.2.3) and the analysis equation (the Fourier transform) (A.2.2.4) for aperiodic discrete-time signal are shown below. Again, the spectrum $X(\Omega)$ is a continuous function due to limit property as in the case of continuous

signal. In (A.2.2.3), the integration can be taken in any 2π interval since $X(\Omega)e^{j\Omega n}$ is periodic with period 2π:

$$x[n] = \frac{1}{2\pi} \int_{2\pi} X(\Omega) e^{j\Omega n} d\Omega , \tag{A.2.2.3}$$

$$X(\Omega) = \sum_{n=-\infty}^{+\infty} x[n] e^{-j\Omega n} . \tag{A.2.2.4}$$

A.2.3 Discrete Fourier Transform

Discrete Fourier Transform (DFT) is an alternate Fourier representation for finite-duration sequences. Unlike Fourier transform for aperiod signals (A.2.2.2) or sequences (A.2.2.4), this transform results in a sequence rather than a function of a continuous variable. There is also the Discrete Fourier Series (DFS) defined for periodic sequences and they are corresponding to the Fourier series representation of the periodic sequence (A.2.1.4). DFS is important in its own right; however, DFT for finite-length sequences often plays a central role in the development of various digital signal processing algorithms. Theoretically, Fourier transform is equal to z-transform evaluated on the unit circle. DFT estimates the Fourier transform of a sequence from a finite number of its sampled points. The sampling[4] takes place at equally spaced points (e.g., at frequencies $\omega_k = 2\pi k / N$) along the unit circle for the z-transform of the original sequence. The N-point DFT analysis (A.2.3.1) and synthesis (A.2.3.2) equations are generally expressed as

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} , \quad 0 \le k \le N-1 , \tag{A.2.3.1}$$

[4] The chirp transform algorithm, an algorithm based on expressing the DFT as a convolution, can compute any set of equally spaced samples of the Fourier transform along the unit circle.

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]W_N^{-kn}, \quad 0 \le n \le N-1, \tag{A.2.3.2}$$

where N is the length of the finite-length $x[n]$ and the complex quantity W_N is defined as $W_N = e^{-j(2\pi/N)}$.

The existence of computational efficient algorithms has made DFT suitable for a wide range of practical discrete-time applications. These algorithms, such as the Cooley-Tukey Algorithm [78], the Prime Factor Algorithm [79, 80] or the Winograd Fourier Transform Algorithm [81], are collectively known as fast Fourier transform (FFT). In general, FFT algorithms are based on the fundamental principle to decompose a DFT computation of the original sequences into successively shorter DFT. Besides providing efficient computation for DFT, different FFT algorithms offer a variety of advantages such as the avoidance of intermediate results storage (i.e., *in-place computation*). The computation complexity (usually measured by the number of complex multiplications and additions) for the fastest FFT algorithm approaches $O(N \log_2 N)$ whereas the computation complexity for direct computation of DFT is $O(N^2)$.

A.2.4 Discrete Cosine Transform

Discrete Cosine Transform (DCT) is a cosine basis transform that is closely related to DFT. In contrast to DFT, DCT is real-valued instead of complex-valued which requires less multiplications and additions. DCT is first proposed by Ahmed, Natarajan and Rao [82] in 1974. Four classes of discrete cosine transform, DCT-I to DCT-IV, are defined by Wang [83]. Here, we show the forward (A.2.4.1) and inverse (A.2.4.2) DCT-II:

$$X[m]=\left(\frac{2}{N}\right)^{\frac{1}{2}} k_m \sum_{n=0}^{N-1} x[n]\cos\left(\frac{(2n+1)m\pi}{2N}\right), \quad m=0,\ldots,N\text{-}1; \tag{A.2.4.1}$$

$$x[n]=\left(\frac{2}{N}\right)^{\frac{1}{2}} \sum_{m=0}^{N-1} k_m X[m]\cos\left(\frac{(2n+1)m\pi}{2N}\right), \quad n=0,\ldots,N\text{-}1; \tag{A.2.4.2}$$

where $x[n]$ is a finite-length sequence with length N and

$$k_j=\begin{cases}1 & if\ j\neq 0\ or\ N\\ \frac{1}{\sqrt{2}} & if\ j=0\ or\ N\end{cases}.$$

DCT (more specifically DCT-II) is an excellent choice for many real-time applications due to the presence of various fast algorithms. For examples, efficient computation can be achieved by computing DCT using FFT, sparse matrix factorization, Decimation-in-Time (DIT), and Decimation-in-frequency (DIF) algorithms. The real-valued nature of DCT has generated at least a saving of N complex operations when comparing to the traditional N-Point DFT via FFT. Because of the fast computation nature, DCT is selected as the commercial standard for lossy compression of still images by JPEG[5] committee. Many individuals have utilized different approaches and algorithms to further improve the fast DCT computation. The followings are some selected contributions in the development of efficient algorithms for the computation of DCT: Vetterli and Nussbaumer [84], Tseng and Miller [85] as well as Duhamel [86] for DCT computation using FFT; Kamangar and Rao [87] as well as Nasrabadi and King [88] for fast 2D-DCT computation; Yip and Rao [89, 90] for fast DIF and DIT DCT. For more details, a comparison of different algorithms for DCT-II can be found on Page 82 in [91]. Above all, DCT can be supplemented with other

[5] JPEG stands for the Joint Photographic Experts Group which was formed by industry experts to develop a worldwide standard on image compression.

redundancy reduction algorithms, like Vector Quantization[6] [92], to achieve good performance in image and video compressions.

A.2.5 Windowed Fourier Transforms

Fourier basis has many useful properties. For example, the fact that Fourier basis is the eigenfunctions of a linear time-invariant (LTI) systems provides useful information in system analysis and design. Nonetheless, Fourier basis has obvious limitation to aperiodic signal or nonstationary[7] signal in the context of random processes. The summation of the period functions, harmonic sine and cosine, does not accurately represent an aperiodic signal. Fourier transform reveals the frequency content of a signal but fail to localize frequency in time. As mentioned in previous sections, we can "artificially" extend the signal to periodic by segmentation and piecewise Fourier series expansion of each segment. Inevitably, this would lead to artificial boundary effects and poor convergence at these boundaries due to the *Gibbs phenomenon.*

In practice, it is often possible to treat nonstationary signals as stationary ones by dividing them into blocks of short, pseudostationary segments. The statistics nature of these segments essentially unchanged for their duration. This technique for creating local Fourier bases is known as *windowed Fourier transform* (WFT) or *short-time Fourier transform* (STFT)[8].

---

[6] This technique matches blocks of pixels from the original image to a block template from some pre-defined table referred to as the code book. The matching procedure is based on error criterion such as the mean square error. The pixels block is then represented by the index of the template to which it was matched. Reconstruction is accomplished by the index of the template to access an identical template table at the receiver's end to retrieve the matched template.

[7] The statistical properties of nonstationary signals vary with time. On the contrary, the statistical properties of stationary signals do not.

[8] When the Gaussian function window is used to perform an analysis, the resultant transform is refer to as the Gabor Transform.

In STFT, a window $w(t)$ is applied to a signal $x(t)$ at equally spaced frequency points. Then, a Fourier expansion is applied to the windowed signal. Formally, the STFT is given by

$$X_{STFT}(e^{j\theta}, m) = \sum_{k=-\infty}^{+\infty} x(k)w(k-m)e^{-jk\theta}. \qquad \text{(A.2.5.1)}$$

The STFT at time m is computed by shifting the center of the window $w(t)$ to m. Provided that $w(0) \neq 0$, the signal $x(t)$ can be recovered from $X_{STFT}(e^{j\theta}, m)$ by using the inverse DFT (A.2.2.1)

$$x(m) = \frac{1}{2\pi w(0)} \int_{-\pi}^{\pi} X_{STFT}(e^{j\theta}, m)e^{jm\theta} d\theta. \qquad \text{(A.2.5.2)}$$

The result of (A.2.5.1) is a time-frequency representation as we obtain approximate information about the frequency content of the signal around the center of the window signal. The center points between consecutive m are equally spaced and we will get a sampling of the time-frequency plane on a rectangular grid as shown below.

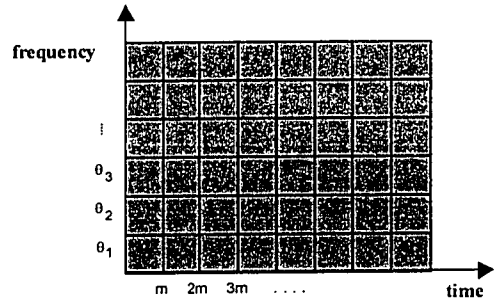


Figure A-1 Time-frequency tilings for STFT

Gabor expressed the condition for optimal localization in both spatial and frequency domains by using the Gaussian function as the window signal. The relationship between spatial and frequency resolution is found to be an uncertainty formula [26]

$$\Delta x \Delta \omega \geq \frac{1}{2}, \tag{A.2.5.3}$$

where Δx is the time resolution and $\Delta \varpi$ is the frequency resolution. This formula puts a lower bound on the product of space and frequency resolution. In order to distinguish between two pulses x_1 and x_2, they must be apart by at least Δx. Similarly, in order to distinguish between two frequency components ω_1 and ω_2, they must be separated by at least $\Delta \varpi$. The product for both time and frequency resolutions should be equal to or greater then 0.5. Although the time resolution can be traded for frequency resolution or vice versa, both variables remain constant throughout the STFT analysis.

There are many situations in which a simultaneous characterization of a signal in both spatial and frequency domains is required. For instances, radar signals have a time dependent spectrum; speech would have different frequency behavior at different segments of the spoken words. While valuable in some applications, the short-time Fourier transform is fundamentally flawed: *if the time-domain analysis window is made too short, frequency resolution will suffer. But lengthening it, on the other hand, may invalidate the assumption of stationary within the window.*

Localization of a transform refers to the ability of the transform to provide information about an interval of arbitrary length in the spaces that the transform covers. Ideally, it would be nice if the transform can simultaneously discriminate between any two frequency components, and any two pulses in the spatial (time)

domains. Obviously, these two requirements conflict with each other in STFT and a compromise is required.

Appendix B

All MATLAB functions and programs employed and developed for this project are listed in alphabetical order in this section for reference.

B.1 MATLAB Function: circonv

```
function z = circonv(x,y)

% circular convolution

Lx = length(x);
Ly = length(y);

if (Lx >= Ly)
       tmp = conv(x,y);
       z(1:Ly-1) = tmp(1:Ly-1) + tmp (Lx+1:Lx+Ly-1);
       z(Ly:Lx)  = tmp(Ly:Lx);
else
       tmp = conv(y,x);
       z(1:Lx-1) = tmp(1:Lx-1) + tmp (Ly+1:Ly+Lx-1);
       z(Lx:Ly)  = tmp(Lx:Ly);
end
```

B.2 MATLAB Function: circonvq

```
function z = circonvq(x,y, i_mode, i_bit)

% function z = circonvq(x,y, i_mode, i_bit)
% circular convolution with arithemic quantization
%
% i_mode - '0' for no internal results quantization, '1' for
%          rounding and '2' for truncation.
% i_bit - Number of bit for representing internal results
%

Lx = length(x);
Ly = length(y);

if (Lx >= Ly)
       tmp = conv(x,y);
```

```
        if i_mode == 1
                [tmp, nfactor] = cround(tmp, i_bit);
                tmp = tmp * nfactor;
        elseif i_mode == 2
                [tmp, nfactor] = ct(tmp, i_bit);
                tmp = tmp * nfactor;
        end

        z(1:Ly-1) = tmp(1:Ly-1) + tmp (Lx+1:Lx+Ly-1);
        z(Ly:Lx)  = tmp(Ly:Lx);
else
        tmp = conv(y,x);

        if i_mode == 1
                [tmp, nfactor] = cround(tmp, i_bit);
                tmp = tmp * nfactor;
        elseif i_mode == 2
                [tmp, nfactor] = ct(tmp, i_bit);
                tmp = tmp * nfactor;
        end
        z(1:Lx-1) = tmp(1:Lx-1) + tmp (Ly+1:Ly+Lx-1);
        z(Lx:Ly)  = tmp(Lx:Ly);
end
```

B.3 MATLAB Function: cround

```
function [aq, nfactor] = cround(a,width);

% function [aq, nfactor] = cround(a,width);
% Quantizes a given vector a by rounding to wordlength w.

b = log(max(abs(a)))/log(2);
n = 2^ceil(b);
an = a/n;
aq = fxquant(an,width,'round','sat')
nfactor = n;
```

B.4 MATLAB Function: ct

```
function [aq, nfactor] = ct(a,width);

% function [aq, nfactor] = ct(a,width);
% Quantizes a given vector a by truncation to wordlength w.

b = log(max(abs(a)))/log(2);
n = 2^ceil(b);
an = a/n;
aq = fxquant(an,width,'trunc','sat')
nfactor = n;
```

B.5 MATLAB Function: Daub

```
% Daub.m
% Daubechies wavelet coefficient published in [41]
%

N2 =  [ .482962913145 .836516303738 .224143868042 -.129409522551 ]

N3 =  [ .332670552950 .806891509311 .459877502118 -.135011020010
        -.085441273883 .035226291882 ]

N4 =  [ .230377813309 .714846570553 .630880767930 -.027983769417
        -.187034811719 .030841381836 .032883011667 -.010597401785 ]

N5 =  [ .160102397974 .603829269797 .724308528438 .138428145901
        -.242294887066 -.032244869585 .077571493840 -.006241490213
        -.012580751999 .003335725285 ]

N6 =  [ .111540743350 .494623890398 .751133908021 .315250351709
        -.226264693965 -.129766867567 .097501605587 .027522865530
        -.031582039318 .000553842201 .004777257511 -.001077301085 ]

N10 = [ .026670057901 .188176800078 .527201188932 .688459039454
        .281172343661 -.249846424327 -.195946274377 .127369340336
        .093057364604 -.071394147166 -.029457536822 .033212674059
        .003606553567 -.010733175483 .001395351747 .001992405295
        -.000685856695 -.000116466855 .000093588670 -.000013264203]
```

B.6 MATLAB Function: daubfcoeff

```
function [h] = daubfcoeff(N)
%
% Generate length-N Orthonormal Daubechies filter coefficients
% Warning: Not accurate for N > 20

N2 = N/2;
a = 1;
p = 1;
q = 1;
h = [1 1];

for j = 1:N2-1
      h = conv(h,[1 1]);
      a = -a * .25 * (j+N2-1)/j;
      p = conv(p, [1 -2 1];
      q = [0 q 0] + a*p;
end

q = sort(roots(q));
h = conv(h, real(poly(q(1:N2-1))));
h = h*sqrt(2)/sum(h);
```

B.7 MATLAB Function: latfilt

```
function [h,g] = latfilt (q, x)

% function [g,h] = latfilt (q, x)
%
% h and g are the output of lattice filter with odd coefficents q.
% x is the input with even element. The length of h and g will be
% half of that of x. The extra ouputs in h and g due to delay element
% are discarded. q must be more than or equal to two elements.
%

order = length(q);

xo = x(1:2:length(x))                    % odd elemenst of x
xe = x(2:2:length(x))                    % even elements of x

% Initialization (After first butterfly and before the delay element)
yo(1,:) = xo - q(1)*xe;
ye(1,:) = q(1)*xo + xe;

n =  (length(x)/2 + order - 1)

for i = 2:order
      for n = 1:(length(x)/2 + order - 1)
            if n == 1
                  yo(i,1) = yo(i-1,1)                    % ye(i-1,0) = 0
                  ye(i,1) = q(i)*yo(i-1,1)               % ye(i-1,0) = 0
            else
                  yo(i,n) = yo(i-1,n) - q(i)*ye(i-1,n-1)
                  ye(i,n) = q(i)*yo(i-1,n) + ye(i-1,n-1)
            end
      end
end

g = yo(order,:)
h = ye(order,:)
```

B.8 MATLAB Function: ploy2qmf

```
function [q,H] = poly2qmf (poly)
%
% Calculate the QMF lattice coefficients from transfer function poly.
%
% The naming convention and equations used are based on Section 2.3.2
% of Multirate Digital Signal Processing (John Wiley & Sons; 1994)
% by N.J. Fliege.
%
% poly is a normalized. (i.e., poly(1) = 1). Also the poly contains
% even number of element. For meaningful results, the input transfer
% function poly must fit the special properties limit imposed by QMF.
% Daubechies filter
```

```
% coefficients are one of the valid examples.
%
% q contains the lattice coefficients. The first element in vector q
% is q1.  All coefficients with even-valued indices are zero.
%
% H contains the intermediate results for transfer function of the
% lattice.  It is used for verification purpose.
%

order = length (poly) - 1;

H = zeros(size(H));                % fill H with zeros
G = zeros(size(G));                % fill G with zeros
q = zeros(size(q));                % fill q with zeros

H(order,:) = poly;
q(order) = -1*poly(order + 1);
t = poly;

% change signs for signal index (even for MATLAB notation)
for k = 2:2:length(t);
    t(k) = -1*t(k);
end

G(order,:) = fliplr(t)             % flip vector

tempH = H(order,:);
tempG = G(order,:);

for i = order:-2:1+2
       tempH = (tempH + q(i)*tempG)/(1+q(i)^2)
       q(i-2) = -1*tempH (i-2 + 1)

       tempH(i+1) = [] ;           % remove last element
       tempH(i) = []               % remove second last element

       % store the overall transfer function
       H(i-2,1:length(tempH)) = tempH;

       t = tempH;
       for j = 2:2:length(t);
              t(j) = -1*t(j);      % change size for even index
       end

       tempG = fliplr(t);

       tempG(i+1) = [];
       tempG(i) = [];
       G(i-2,1:length(tempG)) = tempG;
End
```

B.9 MATLAB Function: pwtldgen

```
function matrix = pwtldgen (matrix, len, coeff, mode, q_mode, q_bit,
i_mode, i_bit )

% function matrix = pwtldgen (matrix, len, coeff, mode, q_mode, q_bit,
% i_mode, i_bit)
%
% Perform forward or inverse wavelet transform on data stored in array
%
% matrix - data to be processed
% len - length of the data to be processed. The length is halved for
% each stage.
% coeff - number of filter coefficients.
% mode - '1' for forward transform and else for inverse transform
% q_mode - '0' for no coefficient quantization, '1' for rounding and
%       '2' for truncation.
% q_bit - Number of bit for representing filter coefficient
% i_mode - '0' for no internal results quantization, '1' for rounding
and '2' for truncation.
% i_mode - Number of bit for representing internal results
%
% see also pwtldgen dau



if q_mode == 0
       lp = dau(coeff);
elseif q_mode == 1
       [lp, nfactor] = cround(dau(coeff), q_bit)
       lp = lp * nfactor;
elseif q_mode == 2
       [lp, nfactor] = ct(dau(coeff), q_bit)
       lp = lp * nfactor;
end



sign = -1;

for m = 1:coeff
       hp(coeff-m+1) = lp(m) .*sign;
       sign = -sign;
end;

if (mode == 1)


       %wksp1 = circonv(matrix(1:len),lp);
       wksp1 = circonvq(matrix(1:len),lp, i_mode, i_bit);
       wksp1 = [ wksp1(coeff:len) wksp1(1:coeff-1)];
       wksp1 = wksp1(1:2:len);

       %wksp2 = circonv(matrix(1:len),hp);
```

```
        wksp2 = circonvq(matrix(1:len),hp, i_mode, i_bit);
        wksp2 = [wksp2(coeff:len) wksp2(1:coeff-1)];
        wksp2 = wksp2(1:2:len);
        matrix = [ wksp1 wksp2 ];

else
        lp = lp(coeff:-1:1);
        hp = hp(coeff:-1:1);

        y = [matrix(1:len/2) ; zeros(1,len/2)];
        ups = y(1:len);
        %wksp1 = circonv(ups(1:len),lp);
        wksp1 = circonvq(ups(1:len),lp, i_mode, i_bit);

        y = [matrix(len/2+1:len); zeros(1,len/2)];
        ups = y(1:len);
        %wksp2 = circonv(ups(1:len),hp);
        wksp2 = circonvq(ups(1:len),hp, i_mode, i_bit);

        for j = 1:len
              matrix(j) = wksp1(j) + wksp2(j);
        end

end
```

B.10 MATLAB Function: qmf2p

```
function [H] = qmf2p (lq, scale)
%
% function [H] = qmf2p (lq) or
% function [H] = qmf2p (lq, scale)
%
% Calculate the transfer function from QMF lattice coefficients (lq).
%
% The naming convention and equations used are based on Section 2.3.2
% of Multirate Digital Signal Processing (John Wiley & Sons; 1994)
% by N.J. Fliege.
%
%       lq contains the lattice coefficients. The first element in
%       vector lq is q1.
%       Since all even-valued indices are zero, even-valued indices are
%       omitted in lq (e.g. [q1 q3 q5 ....]).
%
%       scale is the scaling factor that will multiple to the final
%       result in H.
%
%       H contains the transfer function calculated from the input
%       lattice.
%       coefficients. H is normalized (i.e., H(1) = 1).
%

format long;
len = length (lq) * 2;
H = zeros(1,len);
```

```
tempH = zeros( length(lq), length(H));
tempG = zeros( length(lq), length(H));

% Initialization
tempH (1,:) = [ 1, -1*lq(1), zeros(1,length(H)-2)];
tempG (1,:) = [ lq(1), 1, zeros(1,length(H)-2)];
if length(lq) > 1
       for i = 3:2:len
              j = ceil(i/2);
              tempG2 = [ 0, 0, tempG(j-1,1:length(H)-2) ];
              tempH(j,:) = tempH(j-1,:) - lq(j)*tempG2;
              tempG(j,:) = lq(j)*tempH(j-1,:) +tempG2;
       end
end

if nargin == 1,
       H = tempH(ceil(len/2),:);
elseif nargin == 2,
       H = scale * tempH(ceil(len/2),:);
end
```

B.11 MATLAB Function: quan

```
function quan.m
%
% Demo for plotting magnitude and phase for Daubechies family with
% quantized filter coefficients.
%

format long;
daub;  % Daubechies coefficients

[H,W] = freqz(N2/sqrt(2),1,512);

subplot (3,2,1), plot (abs(W)/pi, abs(H), abs(W)/pi, abs(fliplr(H')));
set(gca, 'xgrid', 'on','ygrid','on');
xlabel('Normalized frequency')
ylabel('Magnitude')
text(.07, 1.2, 'G')
text(.9, 1.2, 'H')
title ('(a) Magnitude Responses')
ax = gca;

subplot (3,2,2), plot(abs(W)/pi, unwrap(angle(H))*180/pi);
set(gca, 'xgrid', 'on','ygrid','on');
xlabel('Normalized frequency')
ylabel('Phase')
title ('(b) Frequency Response for G')

% Quantization (Rounding)

N2_3R  = cround(N2/sqrt(2),3)      % 3 bits
N2_4R  = cround(N2/sqrt(2),4)
```

```
N2_5R  = cround(N2/sqrt(2),5)      % 5 bits
N2_6R  = cround(N2/sqrt(2),6)
N2_7R  = cround(N2/sqrt(2),7)
N2_8R  = cround(N2/sqrt(2),8)      % 8 bits
N2_9R  = cround(N2/sqrt(2),9)
N2_10R = cround(N2/sqrt(2),10)
N2_11R = cround(N2/sqrt(2),11)
N2_12R = cround(N2/sqrt(2),12)     % 12 bits

[HQ_3R,W]  = freqz(N2_3R,1,512);
[HQ_4R,W]  = freqz(N2_4R,1,512);
[HQ_5R,W]  = freqz(N2_5R,1,512);
[HQ_6R,W]  = freqz(N2_6R,1,512);
[HQ_7R,W]  = freqz(N2_7R,1,512);
[HQ_8R,W]  = freqz(N2_8R,1,512);
[HQ_9R,W]  = freqz(N2_9R,1,512);
[HQ_10R,W] = freqz(N2_10R,1,512);
[HQ_11R,W] = freqz(N2_11R,1,512);
[HQ_12R,W] = freqz(N2_12R,1,512);

D_3R  = HQ_3R - H;
D_4R  = HQ_4R - H;
D_5R  = HQ_5R - H;
D_6R  = HQ_6R - H;
D_7R  = HQ_7R - H;
D_8R  = HQ_8R - H;
D_9R  = HQ_9R - H;
D_10R = HQ_10R - H;
D_11R = HQ_11R - H;
D_12R = HQ_12R - H;

subplot   (3,2,3),   plot(abs(W)/pi,   abs(HQ_3R),   '.',   abs(W)/pi,
abs(HQ_5R),   '-.',   abs(W)/pi,   abs(HQ_8R),'--'   ,   abs(W)/pi,
abs(HQ_12R));
set(gca, 'xgrid', 'on','ygrid','on');
xlabel('Normalized frequency')
ylabel('Quantized Freq. Resp.')
text(.3, .4, 'Q=3')
text(.5, 1, 'Q=12')
title ('(c) Quantized Rounding (3,5,8 & 12 bits)')

subplot (3,2,4), plot(abs(W)/pi, abs(D_3R), '.', abs(W)/pi, abs(D_5R),
'-.', abs(W)/pi, abs(D_8R),'--' , abs(W)/pi, abs(D_12R), abs(W)/pi,
abs(D_4R),   abs(W)/pi,   abs(D_6R),abs(W)/pi,   abs(D_7R),abs(W)/pi,
abs(D_9R),abs(W)/pi, abs(D_10R), abs(W)/pi, abs(D_11R));
set(gca, 'xgrid', 'on','ygrid','on');
xlabel('Normalized frequency')
ylabel('Deviated Freq. Resp.')
text(.6, .25, 'Q=3')
title ('(d) Quantized Rounding (3-12 bits)')

% Quantization (Truncation)

N2_3T  = ct(N2/sqrt(2),3)  % 3 bits
N2_4T  = ct(N2/sqrt(2),4)
N2_5T  = ct(N2/sqrt(2),5)  % 5 bits
N2_6T  = ct(N2/sqrt(2),6)
```

```
N2_7T  = ct(N2/sqrt(2),7)
N2_8T  = ct(N2/sqrt(2),8)  % 8 bits
N2_9T  = ct(N2/sqrt(2),9)
N2_10T = ct(N2/sqrt(2),10)
N2_11T = ct(N2/sqrt(2),11)
N2_12T = ct(N2/sqrt(2),12) % 12 bits

[HQ_3T,W]  = freqz(N2_3T,1,512);
[HQ_4T,W]  = freqz(N2_4T,1,512);
[HQ_5T,W]  = freqz(N2_5T,1,512);
[HQ_6T,W]  = freqz(N2_6T,1,512);
[HQ_7T,W]  = freqz(N2_7T,1,512);
[HQ_8T,W]  = freqz(N2_8T,1,512);
[HQ_9T,W]  = freqz(N2_9T,1,512);
[HQ_10T,W] = freqz(N2_10T,1,512);
[HQ_11T,W] = freqz(N2_11T,1,512);
[HQ_12T,W] = freqz(N2_12T,1,512);

D_3T  = HQ_3T - H;
D_4T  = HQ_4T - H;
D_5T  = HQ_5T - H;
D_6T  = HQ_6T - H;
D_7T  = HQ_7T - H;
D_8T  = HQ_8T - H;
D_9T  = HQ_9T - H;
D_10T = HQ_10T - H;
D_11T = HQ_11T - H;
D_12T = HQ_12T - H;

subplot   (3,2,5),   plot(abs(W)/pi,   abs(HQ_3T),   '.',   abs(W)/pi,
abs(HQ_5T),   '-.',   abs(W)/pi,   abs(HQ_8T),'--'   ,   abs(W)/pi,
abs(HQ_12T));
set(gca, 'xgrid', 'on','ygrid','on');
xlabel('Normalized frequency')
ylabel('Quantized Freq. Resp.')
text(.1, .4, 'Q=3')
title ('(e) Quantized Truncation (3,5,8 & 12 bits)')

subplot (3,2,6), plot(abs(W)/pi, abs(D_3T), '.', abs(W)/pi, abs(D_5T),
'-.', abs(W)/pi, abs(D_8T),'--' , abs(W)/pi, abs(D_12R), abs(W)/pi,
abs(D_4R),   abs(W)/pi,   abs(D_6T),abs(W)/pi,   abs(D_7T),abs(W)/pi,
abs(D_9T),abs(W)/pi, abs(D_10T), abs(W)/pi, abs(D_11T));
set(gca, 'xgrid', 'on','ygrid','on');
xlabel('Normalized frequency')
ylabel('Deviated Freq. Resp.')
text(.25, .42, 'Q=3')
title ('(f) Quantized Truncation (3-12 bits)')
```

B.12 MATLAB Function: randv

```
function y = randv(n, v)

% function y = randv(n, v)
% Generate a length-v vector consists of integer from 1 to n in
```

```
% random order

y = 1 + floor(n*rand(1,v));
```

B.13 MATLAB Function: wtld

```
function array = wtld (array, length, nstage, coeff, wt_mode, q_mode,
q_bit, i_mode, i_bit)

% function array = wtld (array, length, nstage, coeff, wt_mode,
% q_mode, q_bit, i_mode, i_bit)
%
% Perform forward or inverse wavelet transform on data stored in array
%
% array - data to be processed
% length - length of the data to be processed. The length is halved
% for each stage.
% nstage - number of decomposition stage.
% coeff - number of filter coefficients.
% wt_mode - '1' for forward transform and else for inverse transform
% q_mode - '0' for no coefficient quantization, '1' for rounding and
%          '2' for truncation.
% q_bit - Number of bit for representing filter coefficient
% i_mode - '0' for no internal results quantization, '1' for rounding
%          and '2' for truncation.
% i_bit - Number of bit for representing internal results
%
% see also pwtldgen dau

if (wt_mode == 1)

    % foward wavelet transform

    len = length;
    for i = 1:nstage
        wksp = pwtldgen (array, len, coeff, wt_mode, q_mode, q_bit,
                         i_mode, i_bit);
        if ( i > 1)
            array = [ wksp(1:len) array(len+1:length)];
        else
            array = wksp(1:len);
        end
        len = len/2;
    end
else

    % inverse wavelet transform

    len = length/(2^(nstage-1));
    for i = 1:nstage
        wksp = pwtldgen (array, len, coeff, wt_mode, q_mode, q_bit,
                         i_mode, i_bit);

        if ( i < nstage)
```

```
            array = [ wksp(1:len) array(len+1:length)];
        end

        len = len * 2;
    end
    array = wksp(1:length);
end
```

B.14 MTALAB Function: wtdemo1

```
% wtdemo1.m
% This program is an example of using MATLAB function wt1d
% to decompose and to reconstruct a signal sequence.

clear
length = 64;  % sequence length
nstage = 3;   % number of decomposition/reconstruction stage
coeff = 6;    % number of Daubechies filter coefficient
              % (3rd order; N=3)

% step
data = [ zeros(1,length/4), randv(1,length/4), zeros(1,length/4), -
1*randv(1,length/4)];

xscale = 1:1:length;

data0 = wt1d(data,length,nstage, coeff,1,1,5,1,10 );
data1 = wt1d(data0,length,nstage, coeff,-1,1,5,1,10);

plot(xscale, data, xscale, data1)
title('Input/ouput of MATLAB function: wt1d')
```

B.15 MATLAB Function: wtdemo2

```
% wtdemo2.m
% This program illustrates the effect of using a subset of subband
% to reconstruct the signal.

clear
length = 128;
nstage = 3;
coeff = 4;


%step
data = [ zeros(1,length/4), randv(1,length/4), zeros(1,length/4), -
1*randv(1,length/4)];

%ramp
%data = [1:1:length];

xscale = 1:1:length;

data0 = wt1d(data,length,nstage, coeff,1,1,5,1,10 );
```

```
data1 = wt1d(data0,length,nstage, coeff,-1,1,5,1,10);

%subplot (5,1,1), plot(xscale, data, '-.', xscale, data1);
%set(gca, 'xgrid', 'on','ygrid','on');


% Throw away high frequency components (H)
data2 = [ data0(1:length/2) zeros(size(data0(1:length/2))) ];
data2 = wt1d(data2,length,nstage, coeff,-1,1,5,1,10);

% Use low frequency components only (LLH, LLL)
data3 = [ data0(1:length/4)  zeros(size(data0(1:length/4*3))) ];
data3 = wt1d(data3,length,nstage, coeff,-1,1,5,1,10);

% Throw away all low frequency components (Use H, LH)
data4 = [ data0(1:length/8)  zeros(size(data0(1:length/8*7))) ];
data4 = wt1d(data4,length,nstage, coeff,-1,1,5,1,10);

subplot (2,2,1), plot(xscale, data, xscale, data1, ':');
set(gca, 'xgrid', 'on','ygrid','on');
title ('Use H, LH, LLH & LLL');
axis([0 length -1.5 1.5])

subplot (2,2,2), plot(xscale, data, xscale, data2, ':');
set(gca, 'xgrid', 'on','ygrid','on');
title ('Use LH, LLH & LLL');
axis([0 length -1.5 1.5])

subplot (2,2,3), plot(xscale, data, xscale, data3, ':');
set(gca, 'xgrid', 'on','ygrid','on');
title ('Use LLH & LLL');
axis([0 length -1.5 1.5])

subplot (2,2,4), plot(xscale, data, xscale, data4, ':');
set(gca, 'xgrid', 'on','ygrid','on');
title ('Use LLL');
axis([0 length -1.5 1.5])
```

Appendix C

This appendix tabulates the simulation results for the finite wordlength effects due to filter coefficient quantization. Results for two structures (e.g., the direct and lattice structure) and two quantization methods (e.g., rounding and truncation) are included.

C.1 The Direct Structure

C.1.1 Magnitude Response Error (Rounding)

Table C-1 Simulation results of magnitude errors on the direct structure due to *N*-bit coefficient quantization with rounding for the Daubechies wavelet family (*N*2-*N*6).

Wavelet Order (*N*)	*N*-bit Quantization	$\Delta h[n]\|_{max}$	Magnitude Error $\lvert\Delta H(e^{j\omega})\rvert$ (dB)					
			Passband			Stopband		
			Max.	Mean	Var.	Max.	Mean	Var.
2	3	0.120590477449	-11.00	-16.52	252.60	-11.71	-25.54	244.72
	4	0.038483696262	-19.73	-25.22	237.27	-20.44	-34.27	244.72
	5	0.025856131958	-20.98	-26.46	235.12	-21.69	-35.52	244.72
	6	0.014212913145	-43.21	-48.60	199.01	-43.92	-57.75	244.72
	7	0.007233696262	-43.21	-48.60	199.01	-43.62	-57.75	244.72
	8	0.003402977449	-43.21	-48.60	199.00	-43.92	-57.75	244.72
	9	0.001487618042	-45.82	-51.20	195.03	-46.53	-60.35	244.72
	10	0.000578803738	-60.97	-66.29	172.94	-61.68	-75.51	244.72
	11	0.000473289949	-60.97	-66.29	172.94	-61.68	-75.51	244.72
	12	0.000090522488	-64.31	-69.62	168.31	-65.02	-78.85	244.72
3	3	0.114988979990	-12.28	-20.24	240.13	-12.28	-23.94	223.23
	4	0.056891509311	-18.06	-21.89	12.10	-18.05	-19.05	2.35
	5	0.027273708118	-23.81	-27.38	15.62	-24.08	-29.98	12.11
	6	0.011079447050	-30.32	-39.55	282.06	-30.88	-34.22	29.34
	7	0.007316273883	-36.12	-40.40	11.23	-36.12	-41.60	28.14

Wavelet Order (N)	N-bit Quantization	$\Delta h[n]\|_{max}$	Magnitude Error $\|\Delta H(e^{j\omega})\|$ (dB)					
			Passband			Stopband		
			Max.	Mean	Var.	Max.	Mean	Var.
	8	0.003836208118	-42.74	-50.26	202.19	-43.30	-63.22	538.46
	9	0.001707729990	-47.31	-54.52	196.38	-48.07	-51.33	33.60
	10	0.000893127118	-53.22	-57.39	28.02	-54.19	-61.15	16.66
	11	0.000480336383	-62.68	-68.08	148.41	-60.75	-65.05	29.38
	12	0.000242886240	-64.01	-68.67	142.10	-66.46	-88.54	571.40
4	3	0.119119232070	-12.71	-18.58	231.97	-13.48	-17.58	27.80
	4	0.062034811719	-17.22	-27.12	358.73	-21.58	-29.82	141.01
	5	0.030841381836	-21.60	-29.71	211.68	-21.26	-29.55	43.34
	6	0.011627813309	-26.84	-33.61	210.01	-34.58	-42.56	136.35
	7	0.005880767930	-29.74	-38.71	236.66	-32.51	-38.15	42.88
	8	0.003903429447	-41.29	-49.56	228.71	-41.87	-67.97	947.73
	9	0.001931732070	-48.79	-56.51	211.49	-49.99	-53.39	46.91
	10	0.000831776785	-54.19	-64.29	42.47	-51.78	-58.56	51.60
	11	0.000465188281	-58.02	-65.06	55.56	-59.89	-63.26	19.45
	12	0.000168167917	-63.36	-66.88	6.70	-63.86	-67.69	10.50
5	3	0.111571854099	-12.04	-16.72	9.53	-10.45	-14.41	15.35
	4	0.047428506160	-15.32	-25.42	300.21	-16.52	-24.10	35.51
	5	0.030255130415	-24.08	-33.53	30.27	-22.17	-27.86	44.05
	6	0.015071493840	-28.09	-34.29	21.92	-26.43	-29.57	34.36
	7	0.007705112934	-31.90	-37.78	217.63	-33.05	-40.28	36.03
	8	0.003852397974	-38.52	-43.42	188.39	-38.44	-44.36	23.37
	9	0.001709395901	-46.07	-51.85	162.46	-53.29	-60.00	48.51
	10	0.000958255415	-53.00	-55.77	5.99	-52.69	-58.24	34.10
	11	0.000423056400	-55.96	-60.64	28.68	-57.80	-62.15	4.92
	12	0.000243729099	-63.90	-67.89	9.877	-65.43	-70.57	29.19
6	3	0.120233132433	-12.04	-16.04	11.64	-11.54	-13.76	10.49
	4	0.059749648291	-2.50	-16.49	63.19	-10.32	-16.67	50.20
	5	0.030917960682	-26.31	-40.39	203.67	-16.99	-21.58	13.70
	6	0.013459256650	-10.10	-23.02	58.40	-18.44	-23.25	16.44
	7	0.007514693965	-33.40	-40.86	53.54	-34.17	-38.47	18.24
	8	0.003751605587	-37.78	-44.81	180.77	-37.07	-47.85	124.31
	9	0.001740506650	-43.60	-49.03	12.01	-46.08	-53.05	9.02
	10	0.000875823915	-51.94	-58.36	149.07	-49.81	-58.12	42.99
	11	0.000483265398	-55.43	-60.44	16.33	-60.21	-67.65	22.32
	12	0.000212618350	-67.12	-71.42	131.68	-60.57	-66.69	49.92

C.1.2 Magnitude Response Error (Truncation)

Table C-2 Simulation results of magnitude errors on the direct structure due to *N*-bit coefficient quantization with truncation for the Daubechies wavelet family (*N*2-*N*6).

Wavelet Order (N)	N-bit Quantization	$\Delta h[n]\|_{max}$	Magnitude Error $\lvert\Delta H(e^{j\omega})\rvert$ (dB)					
			Passband			Stopband		
			Max.	Mean	Var.	Max.	Mean	Var.
2	3	0.232962913145	-6.02	-10.87	20.81	-16.91	-20.96	35.58
	4	0.120590477449	-12.04	-15.66	9.89	-21.69	-25.71	42.45
	5	0.058090477449	-18.06	-24.31	**46.37**	-29.34	-34.52	39.12
	6	0.026840477449	-24.08	-25.67	1.78	-28.41	-35.67	**60.12**
	7	0.014212913145	-30.10	-32.38	3.58	-36.03	-42.39	53.64
	8	0.006400413145	-36.12	-39.97	11.45	-46.27	-50.03	40.85
	9	0.003402977449	-42.12	-47.82	33.27	-53.33	-57.97	34.96
	10	0.001487618042	-48.16	-50.49	3.74	-54.22	-60.51	53.17
	11	0.000578803738	-54.19	-58.16	12.39	-64.46	-68.22	40.03
	12	0.000473289949	-60.21	-65.59	28.17	-71.31	-75.71	34.58
3	3	0.209877502118	-2.50	-10.79	50.60	-8.81	-13.17	9.83
	4	0.114988979990	-8.52	-15.58	24.04	-18.06	-22.68	20.65
	5	0.056891509311	-18.06	-26.01	27.87	-21.88	-26.72	28.10
	6	0.025641509311	-20.56	-31.34	69.23	-25.99	-31.89	**58.59**
	7	0.010016509311	-26.58	-33.16	28.34	-36.12	-38.59	1.69
	8	0.006752502118	-36.12	-40.19	11.14	-46.37	-50.48	38.42
	9	0.002846252118	-42.14	-47.69	15.62	-53.77	-60.30	27.06
	10	0.001707729990	-48.16	-56.71	49.03	-52.07	-56.48	26.38
	11	0.000893127118	-50.66	-58.23	26.22	-56.18	-60.85	23.62
	12	0.000404845868	-56.68	-64.65	**75.48**	-66.23	-72.56	34.24
4	3	0.239402598215	0.00	-8.35	23.92	-6.47	-12.21	**53.52**
	4	0.114402598215	-6.02	-14.85	35.29	-13.44	-18.31	25.37
	5	0.051902598215	-12.04	-23.80	69.11	-18.40	-25.38	38.88
	6	0.030841381836	-18.06	-27.93	59.49	-33.40	-36.87	21.94
	7	0.015216381836	-24.08	-36.86	**77.56**	-37.95	-44.38	37.09
	8	0.007403881836	-30.10	-37.65	34.20	-37.84	-47.41	0.01
	9	0.003815313309	-36.12	-46.36	44.51	-42.41	-51.55	0.02
	10	0.001862188309	-42.14	-51.42	33.85	-50.67	-56.31	43.45
	11	0.000885625809	-46.23	-57.2	51.24	-59.29	-64.45	26.39
	12	0.000465188281	-54.19	-64.00	35.84	-64.58	-69.21	18.46
5	3	0.243758509787	1.94	-8.63	56.07	-6.73	-10.47	6.42
	4	0.118758509787	-2.50	-14.28	60.91	-11.61	-19.97	75.47
	5	0.056258509787	-8.52	-19.21	53.50	-21.95	-27.58	57.17
	6	0.030255130415	-16.12	-26.84	**61.44**	-29.16	-36.30	40.37
	7	0.015071493840	-24.08	-33.32	29.81	-38.70	-42.06	23.79

Wavelet Order (N)	N-bit Quantization	$\Delta h[n]\|_{max}$	Magnitude Error $\|\Delta H(e^{j\omega})\|$ (dB)					
			Passband			Stopband		
			Max.	Mean	Var.	Max.	Mean	Var.
	8	0.007705112934	-30.10	-38.55	29.54	-42.97	-51.60	134.01
	9	0.003852397974	-36.12	-44.49	41.23	-43.58	-49.63	22.66
	10	0.001899272974	-42.14	-51.40	32.51	-50.48	-57.48	43.16
	11	0.000958255415	-46.23	-57.93	56.25	-54.87	-60.49	11.79
	12	0.000469974165	-50.66	-61.82	57.02	-61.76	-70.06	54.03
6	3	0.248922698915	1.94	-11.90	53.73	-3.63	-9.65	31.38
	4	0.123922698915	-2.50	-16.49	63.19	-10.32	-16.67	50.20
	5	0.061422698915	-10.10	-23.02	58.40	-18.44	-23.25	16.44
	6	0.030917960682	-16.12	-27.98	50.83	-21.72	-27.24	17.09
	7	0.015292960682	-22.14	-33.01	36.38	-30.58	-34.75	14.24
	8	0.007480460682	-26.58	-36.81	42.41	-37.27	-45.97	38.91
	9	0.003751605587	-32.60	-42.76	36.41	-41.58	-50.57	34.88
	10	0.001798480587	-40.21	-49.18	16.59	-49.55	-53.09	3.35
	11	0.000875823915	-46.23	-57.06	35.35	-55.65	-60.20	37.67
	12	0.000483265398	-50.66	-63.53	73.68	-63.30	-67.37	21.12

C.1.3 Phase Response Error

Table C-3 Simulation results of phase errors on the direct structure due to *N*-bit coefficient quantization with truncation for the Daubechies wavelet family (*N*2-*N*6).

Wavelet order (*N*)	*N*-bit Quantization	Phase Error (degree)							
		Rounding				Truncation			
		Passband		Stopband		Passband		Stopband	
		Mean	Var.	Mean	Var.	Mean	Var.	Mean	Var.
2	3	202.67	1497.12	68.03	1536.59	90.41	3525.38	-178.48	3677.66
	4	23.38	1592.04	-111.97	1536.59	144.06	2737.44	126.18	267.46
	5	202.67	1497.12	68.03	1536.59	103.97	2467.39	-191.93	2681.66
	6	23.38	1592.04	-111.97	1536.59	154.84	173.41	115.49	174.09
	7	23.38	1592.04	-111.97	1536.59	151.52	190.71	118.79	189.16
	8	23.38	1592.04	-111.97	1536.59	142.64	296.76	127.59	289.62
	9	-157.33	1497.12	68.03	1536.59	97.63	3023.96	-185.64	3208.33
	10	23.38	1592.04	-111.97	1536.59	151.26	192.58	119.05	190.85
	11	23.38	1592.04	-111.97	1536.59	141.84	310.77	128.39	303.15
	12	20.27	1497.12	68.03	1536.59	94.87	3232.41	-182.90	3404.27
3	3	-195.48	4068.20	-63.04	4533.92	26.60	1556.55	-188.07	338.11
	4	-54.85	680.67	-194.17	7766.16	142.96	349.03	172.56	1255.00
	5	48.34	3463.29	-166.13	1575.66	157.50	715.69	166.37	1217.18
	6	73.28	1653.60	-20.37	973.15	8.81	1664.27	163.83	5397.97
	7	-63.25	632.27	-325.36	2171.62	77.72	2403.86	-159.25	1394.10
	8	-72.74	7164.44	30.02	4830.19	54.36	5236.18	-162.53	350.40
	9	-169.38	3894.68	-116.89	9327.64	15.77	1098.62	-197.63	878.30
	10	103.88	4977.92	-146.83	396.72	-26.45	2410.71	-117.72	1068.63
	11	-261.29	1009.87	-297.48	9783.05	131.25	355.21	-109.70	6657.89
	12	-1.57	275.49	-181.34	2633.70	113.70	1190.28	200.37	121.26
4	3	-34.14	689.20	16.01	3628.75	71.18	4655.98	-174.46	2670.65
	4	-109.09	5312.28	108.50	2506.83	41.77	4861.38	-378.81	27232.99
	5	-90.80	17030.51	-25.31	893.19	-20.89	12962.73	-378.53	28643.10
	6	101.42	10248.96	-207.88	6957.45	-73.20	28047.42	-372.48	26796.42
	7	-155.43	2226.02	-203.32	23089.42	81.06	3376.86	-191.18	1997.74
	8	22.64	2568.27	-217.58	6593.64	-56.74	25166.13	-201.99	13742.89
	9	114.71	6578.85	3.02	2735.04	-54.34	26856.44	-216.36	7946.30
	10	165.98	1617.44	-413.16	15901.40	105.36	1479.80	-255.23	13587.37
	11	-138.72	16581.05	-135.97	11502.78	111.07	1439.83	-167.88	2121.91
	12	-276.16	22787.68	-286.14	3175.00	-81.64	31233.94	-163.40	2745.39
5	3	11.69	1086.79	-167.16	685.43	-138.42	47327.80	-114.85	30341.74
	4	-187.01	5580.38	-250.11	23427.74	-136.24	50050.05	-223.12	1696.71
	5	-242.39	20455.45	-41.28	5091.65	-87.08	24179.01	-257.89	4465.49
	6	-252.37	17330.48	-142.40	10055.89	-452.67	25280.70	25.66	11517.66
	7	-265.04	20655.35	-285.57	51335.32	-113.41	33206.37	-119.40	7085.38

Wavelet order (N)	N-bit Quantization	Phase Error (degree)							
		Rounding				Truncation			
		Passband		Stopband		Passband		Stopband	
		Mean	Var.	Mean	Var.	Mean	Var.	Mean	Var.
	8	-94.14	6268.29	-15.99	26616.26	77.74	9339.07	-138.95	28247.43
	9	-383.04	36687.83	22.85	2950.71	-530.70	49639.31	159.93	838.40
	10	-368.11	7755.97	-91.18	7020.94	152.40	973.54	147.56	1473.79
	11	-518.75	31231.12	-342.39	47801.63	-530.10	58211.36	-86.38	14334.46
	12	-42.83	28260.83	-244.25	11456.58	-521.27	47094.83	-1.23	12545.64
6	3	143.77	378.74	23.81	13320.13	-53.64	41494.94	-505.84	34631.01
	4	-6.55	857.57	13290.93	-115.93	92.90	5794.48	-558.92	64156.74
	5	-210.42	8017.68	-350.76	23899.73	-44.02	41205.12	-575.24	78267.07
	6	29.91	640.09	-204.38	26870.55	-44.16	41415.22	-395.29	16447.59
	7	126.69	2322.68	-234.98	9692.17	72.47	2664.91	-555.01	67319.31
	8	-132.58	12905.62	-414.48	46639.28	90.49	2045.78	-546.45	83923.36
	9	-230.90	25116.30	-230.90	25116.30	-498.80	27470.87	-398.73	28683.31
	10	-390.49	28339.12	-323.27	46103.17	-154.81	30437.55	-392.80	23805.70
	11	-601.49	50991.20	55.31	1438.54	-133.72	30786.14	-148.19	2126.31
	12	-91.32	13424.08	7.14	4035.40	-247.00	7959-.60	-376.57	32850.79

C.2 The Lattice Structure

C.2.1 Magnitude Response Error (Rounding)

Table C-4 Simulation results of magnitude errors on the lattice structure due to *N*-bit coefficient quantization with rounding for the Daubechies wavelet family (*N*2-*N*6).

Wavelet Order (N)	N-bit Quantization	$\Delta h[n]\|_{max}$	Magnitude Error $\lvert\Delta H(e^{j\omega})\rvert$ (dB)					
			Passband			Stopband		
			Max.	Mean	Var.	Max.	Mean	Var.
2	3	0.120590477449	-20.38	-29.74	66.98	-11.83	-14.39	5.91
	4	0.038483696262	-40.98	-52.95	86.49	-33.41	-35.73	4.73
	5	0.025856131958	-40.98	-52.95	86.49	-33.41	-35.73	4.73
	6	0.014212913145	-40.98	-52.95	86.49	-33.41	-35.73	4.73
	7	0.007233696262	-43.77	-55.35	74.16	-36.10	-38.44	4.86
	8	0.003402977449	-58.83	-70.58	77.30	-51.21	-53.54	4.79
	9	0.001487618042	-58.83	-70.58	77.30	-51.21	-53.54	4.79
	10	0.000578803738	-62.19	-73.88	74.53	-54.56	-56.89	4.81
	11	0.000473289949	-74.78	-86.52	78.16	-67.15	-69.48	4.80
	12	0.000090522488	-74.78	-86.52	78.16	-67.15	-69.48	4.80
3	3	0.120590477449	-20.38	-29.74	66.98	-11.83	-14.39	5.92
	4	0.056891509311	-20.76	-36.99	189.28	-18.36	-20.27	3.84
	5	0.027273708118	-20.76	-36.99	189.28	-18.36	-20.27	3.84
	6	0.011079447050	-28.84	-29.42	0.39	-26.76	-28.12	1.69
	7	0.007316273883	-31.91	-35.12	3.91	-31.81	-35.28	11.26
	8	0.003836208118	-36.25	-44.18	22.29	-34.69	-38.55	30.75
	9	0.001707729990	-52.13	-64.62	47.03	-48.16	-49.83	2.12
	10	0.000893127117	-52.13	-64.62	47.03	-48.16	-49.83	2.12
	11	0.000480336383	-68.02	-68.97	1.00	-60.22	-63.57	9.51
	12	0.000242886240	-68.02	-68.97	1.00	-60.22	-63.57	9.51
4	3	0.119119232070	-16.68	-26.58	30.37	-9.74	-11.53	4.15
	4	0.620348117190	-13.08	-15.63	1.69	-11.31	-12.67	1.85
	5	0.030841381836	-21.33	-22.95	1.04	-21.64	-27.92	23.27
	6	0.011627813309	-23.66	-28.53	6.57	-22.49	-27.63	24.75
	7	0.005880767930	-39.78	-42.96	4.84	-32.27	-36.39	17.98
	8	0.003903429447	-35.54	-42.35	15.88	-34.62	-37.30	13.69
	9	0.001931732070	-43.56	-45.77	2.01	-43.56	-45.27	1.10
	10	0.000831776785	-73.00	-79.62	14.44	-49.54	-61.44	0.01
	11	0.000465188281	-73.00	-79.62	14.44	-49.54	-61.44	0.01
	12	0.000168167917	-61.91	-70.99	31.20	-58.62	-61.75	18.12
5	3	0.111571854099	-8.29	-15.45	16.95	-7.41	-9.68	4.06

Wavelet Order (N)	N-bit Quantization	$\Delta h[n]\|_{max}$	Magnitude Error $\|\Delta H(e^{jw})\|$ (dB)					
			Passband			Stopband		
			Max.	Mean	Var.	Max.	Mean	Var.
	4	0.047428506160	-21.70	-31.21	13.01	-12.90	-16.70	6.60
	5	0.030255130415	-21.40	-33.63	38.79	-15.97	-18.86	9.25
	6	0.015071493840	-29.98	-44.00	30.54	-21.37	-25.88	5.09
	7	0.007705112934	-36.85	-44.74	30.71	-29.81	-34.15	22.16
	8	0.003852397974	-39.95	-43.97	4.27	-30.28	-40.38	73.61
	9	0.001709395901	-51.74	-52.77	0.62	-36.66	-45.42	79.71
	10	0.000958255415	-54.03	-56.50	3.54	-46.12	-50.65	16.32
	11	0.000423056340	-60.08	-79.62	141.32	-52.06	-56.26	10.07
	12	0.000243729100	-59.83	-63.31	3.24	-56.02	-61.84	26.21
6	3	0.120233132433	-6.44	-13.71	13.01	-4.20	-4.86	0.23
	4	0.059749648291	-13.50	-14.13	0.22	-6.50	-10.76	16.46
	5	0.030917960682	-12.92	-27.58	54.99	-9.84	-14.13	23.25
	6	0.013459256650	-18.66	-22.65	4.29	-18.66	-23.55	30.32
	7	0.007514693965	-28.46	-34.39	5.09	-19.95	-25.86	13.29
	8	0.003751605587	-33.43	-38.97	6.53	-29.66	-35.21	43.94
	9	0.001740506650	-42.80	-46.05	20.29	-34.21	-38.67	20.49
	10	0.000875823915	-41.67	-43.84	1.12	-34.68	-40.70	22.54
	11	0.000483265398	-46.08	-48.85	2.48	-46.23	-51.92	10.59
	12	0.000212618350	-56.69	-70.97	56.84	-50.55	-55.17	8.19

C.2.2 Magnitude Response Error (Truncation)

Table C-5 Simulation results of magnitude errors on the lattice structure due to *N*-bit coefficient quantization with truncation for the Daubechies wavelet family (*N*2-*N*6).

Wavelet Order (N)	N-bit Quantization	$\Delta h[n]\|_{max}$	Magnitude Error $\|\Delta H(e^{j\omega})\|$ (dB)					
			Passband			Stopband		
			Max.	Mean	Var.	Max.	Mean	Var.
2	3	0.232962913145	-16.12	-30.19	**108.44**	-9.33	-11.45	3.88
	4	0.120590477449	-40.98	-52.95	86.49	-33.41	-35.73	4.73
	5	0.058090477449	-40.98	-52.95	86.49	-33.41	-35.73	4.73
	6	0.026840477449	-40.98	-52.95	86.49	-33.41	-35.73	4.73
	7	0.014212913145	-40.98	-52.95	86.49	-33.41	-35.73	4.73
	8	0.006400413145	-58.83	-70.58	77.30	-51.21	-53.54	4.79
	9	0.003402977449	-58.83	-70.58	77.30	-51.21	-53.54	4.79
	10	0.001487618042	-58.83	-70.58	77.30	-51.21	-53.54	4.79
	11	0.000578803738	-74.78	-86.52	78.16	-67.15	-69.48	**4.80**
	12	0.000473289949	-74.78	-86.52	78.16	-67.15	-69.48	**4.80**
3	3	0.209877502118	-4.25	-5.10	0.63	2.99	-2.57	20.61
	4	0.114988979990	-11.71	-20.35	40.56	-7.11	-8.80	1.91
	5	0.056891509311	-21.47	-28.20	27.96	-14.37	-17.53	5.73
	6	0.025641509311	-37.09	-39.65	4.23	-23.59	-30.68	**44.65**
	7	0.010016509311	-36.28	-36.88	0.23	-25.03	-29.30	14.63
	8	0.006752502118	-43.94	-51.20	**54.52**	-29.87	-34.91	20.14
	9	0.002846252118	-46.82	-51.02	6.62	-35.57	-39.69	12.75
	10	0.001707729990	-56.28	-57.67	1.05	-40.47	-46.00	27.97
	11	0.000893127118	-60.71	-63.85	16.52	-44.59	-49.12	19.66
	12	0.000404845868	-66.48	-71.59	1.84	-52.18	-56.55	16.99
4	3	0.239402598215	-3.35	-4.71	3.91	2.30	-6.53	73.36
	4	0.114402598215	-18.97	-20.78	3.72	-5.86	-16.20	**115.34**
	5	0.051902598215	-18.25	-22.17	9.06	-9.60	-13.46	8.49
	6	0.030841381836	-35.26	-39.45	1.31	-14.36	-21.35	40.60
	7	0.015216381836	-32.19	-34.80	2.07	-17.50	-27.35	66.07
	8	0.007403881836	-42.20	-45.56	2.46	-25.16	-36.47	103.71
	9	0.003815313309	-38.33	-44.38	10.50	-30.55	-37.30	19.39
	10	0.001862188309	-50.67	-66.58	88.98	-35.27	-43.57	37.54
	11	0.000885625809	-57.39	-74.55	**104.04**	-43.16	-50.79	29.68
	12	0.000465188281	-64.94	-79.26	56.07	-54.87	-59.80	10.40
5	3	0.888132333245	5.16	0.61	6.94	0.00	-38.46	**13447.80**
	4	0.118758509787	-10.65	-16.71	12.21	-2.66	-6.79	5.97
	5	0.056258509780	-18.09	-22.52	11.07	-6.36	-13.38	20.12
	6	0.030255130415	-20.31	-29.93	37.94	-13.33	-20.16	19.31
	7	0.015071493840	-29.23	-34.87	10.28	-23.80	-28.15	5.06

Wavelet Order (N)	N-bit Quantization	$\Delta h[n]\|_{max}$	Magnitude Error $\|\Delta H(e^{jw})\|$ (dB)					
			Passband			Stopband		
			Max.	Mean	Var.	Max.	Mean	Var.
	8	0.007705112934	-39.95	-43.97	4.27	-30.28	-40.38	73.61
	9	0.003852397974	-39.95	-43.97	4.27	-30.28	-40.38	73.61
	10	0.001899272974	-47.51	-53.74	9.10	-42.06	-48.43	18.25
	11	0.000958255415	-55.22	-69.65	**83.81**	-46.79	-52.26	7.67
	12	0.000469974165	-68.55	-76.48	31.83	-51.62	-61.42	33.67
6	3	0.248922698915	12.53	6.81	**14.11**	8.43	2.53	42.65
	4	0.123922698915	-3.27	-4.41	0.68	3.96	-1.13	6.79
	5	0.061422698915	-7.78	-12.08	7.64	-0.12	-9.08	31.01
	6	0.030917960682	-19.91	-21.78	2.27	-5.00	-17.28	1.47
	7	0.015292960682	-26.96	-28.11	0.42	-12.99	-20.56	25.80
	8	0.007480460682	-30.08	-32.75	5.26	-15.83	-26.77	**54.00**
	9	0.003751605587	-35.39	-35.68	0.04	-23.28	-32.72	32.99
	10	0.001798480587	-41.30	-44.93	3.12	-29.57	-40.87	42.81
	11	0.000875823915	-61.60	-64.82	6.80	-42.50	-50.12	25.35
	12	0.000483265398	-51.72	-56.06	5.29	-46.76	-55.06	21.45

C.2.3 Phase Response Error

Table C-6 Simulation results of phase errors on the lattice structure due to *N*-bit coefficient quantization with truncation for the Daubechies wavelet family (*N*2-*N*6).

Wavelet order (*N*)	*N*-bit Quantization	Phase Error (degree)							
		Rounding				Truncation			
		Passband		Stopband		Passband		Stopband	
		Mean	Var.	Mean	Var.	Mean	Var.	Mean	Var.
2	3	166.23	2720.40	-269.30	2731.72	30.00	2717.85	-89.30	2731.72
	4	6.00	2721.11	-89.30	2731.72	6.00	2721.11	-89.30	2731.72
	5	6.00	2721.11	-89.30	2731.72	6.00	2721.11	-89.30	2731.72
	6	6.00	2721.11	-89.30	2731.72	6.00	2721.11	-89.30	2731.72
	7	-176.83	2721.19	-269.30	2731.72	6.00	2721.11	-89.30	2731.72
	8	4.59	2721.16	-89.30	2731.72	4.59	2721.16	-89.30	2731.72
	9	4.59	2721.16	-89.30	2731.72	4.59	2721.16	-89.30	2731.72
	10	1.84	2721.18	-269.30	2731.72	4.59	2721.16	-89.30	2731.72
	11	3.88	2721.18	-89.30	2731.72	3.88	2721.18	-89.30	2731.72
	12	3.88	2721.18	-89.30	2731.72	3.88	2721.18	-89.30	2731.72
3	3	-189.60	643.41	-254.15	3263.89	-331.02	8102.95	-366.37	13023.52
	4	-232.25	5542.83	-163.57	9541.90	-52.32	3610.87	0.33	10869.24
	5	-232.25	5542.83	-163.57	9541.90	-49.80	3321.61	-365.02	11103.37
	6	109.89	1689.93	-51.31	4414.96	-23.60	687.65	-82.54	2233.74
	7	-71.00	2316.86	-329.12	10682.30	93.66	2328.27	-61.09	4010.33
	8	101.56	2683.39	-128.24	4970.52	-6.16	8590.71	-19.68	9739.04
	9	-350.96	19132.17	8.28	9035.45	77.77	2805.82	-76.89	2994.67
	10	-350.96	19132.17	8.28	9035.45	78.80	2910.98	-58.77	4247.69
	11	-74.67	1757.87	-229.79	4796.33	34.53	14661.86	-26.55	7941.99
	12	-74.67	1757.87	-229.79	4796.33	21.87	13585.51	-21.43	8785.62
4	3	101.09	4118.63	-208.45	8663.61	-319.59	6443.55	-371.51	15781.54
	4	-94.48	3151.63	19.17	9.41	28.60	6469.34	-381.17	12519.07
	5	103.00	2511.63	-262.78	31408.07	-69.88	3313.52	-0.1025	10639.73
	6	94.00	2832.37	-270.37	30354.15	-375.12	12472.80	1.33	10317.30
	7	-62.25	2459.05	-111.59	22615.55	86.23	2535.10	-311.88	25741.69
	8	106.68	3202.38	-251.22	19431.61	75.39	2802.92	-52.66	3760.21
	9	-80.18	2863.56	-3.80	10757.08	94.74	2810.98	-290.33	26761.04
	10	-128.52	2549.51	-155.05	21717.00	49.07	2058.15	-308.93	23037.25
	11	-128.52	2549.51	-155.05	21717.00	-135.82	26399.90	-305.34	23005.26
	12	-64.49	3670.60	-72.89	17253.88	-159.75	18462.64	-293.55	22422.33
5	3	85.67	3843.99	-201.50	8593.07	-38.46	13447.80	-205.75	41218.89
	4	-29.67	484.05	-314.49	19407.23	-13.96	11064.03	-205.67	39197.78
	5	-118.48	3504.92	-79.91	16694.14	-16.72	9549.47	-215.44	40450.56
	6	-155.66	21362.53	-225.14	5611.69	-65.09	5391.84	-201.63	45288.07
	7	122.62	4262.05	-356.94	29211.40	-78.19	3454.62	-202.22	41245.84

Wavelet order (*N*)	*N*-bit Quantization	Phase Error (degree)							
		Rounding				Truncation			
		Passband		Stopband		Passband		Stopband	
		Mean	Var.	Mean	Var.	Mean	Var.	Mean	Var.
	8	-87.85	3396.68	-217.99	49589.43	-87.85	3396.68	-217.99	49589.43
	9	100.49	2260.72	-110.34	15688.97	-87.85	3396.68	-217.99	49589.43
	10	69.80	4651.99	-208.68	23707.54	-93.31	3738.05	-19.88	6849.28
	11	32.53	1914.37	-353.39	24080.88	-48.15	5601.49	-204.10	41202.00
	12	-95.84	3593.33	15.49	6946.02	-39.40	4939.45	-42.26	4924.62
6	3	72.57	4973.73	-228.97	6962.45	-13.98	11088.79	-159.82	40628.12
	4	81.04	4044.17	-331.21	18837.56	-31.17	13805.25	-211.58	39343.08
	5	-202.17	22017.69	-289.92	14802.62	-16.49	14155.78	-221.92	42341.67
	6	71.74	4470.56	-392.81	45363.75	-0.08	10850.40	-247.47	39780.65
	7	116.35	779.94	-201.53	39123.28	54.97	4949.12	-285.08	19958.92
	8	69.88	4581.79	-197.64	7068.50	71.20	3603.95	-248.90	29944.76
	9	-141.99	15944.18	-441.18	34872.74	74.19	4102.86	-249.32	24628.20
	10	-103.67	4292.07	-63.09	20062.59	78.80	3925.82	-501.58	68127.83
	11	72.11	4508.49	-220.52	17019.73	-147.88	14442.54	-370.74	10900.54
	12	101.86	4515.82	-482.43	62407.97	75.44	4248.86	-481.24	67827.51

BIBLIOGRAPHY

[1] A. Grossmann and J. Morlet, "Decompositions of Hardy Function into Square Integrable Wavelets of Constant Shape," *SIAM J. Math*, vol. 15, pp. 723-736, 1984.

[2] I. Daubechies, "Orthonormal Bases of Compactly Supported Wavelets," *Communications on Pure and Applied Mathematics*, vol. 41, pp. 909-996, 1988.

[3] Y. Meyer, "Ondelettes et functions splines," *Sem. Equations aux Derivees Partielles*, 1986.

[4] S. G. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 11, pp. 674-693, 1989.

[5] M. Akay, "Wavelet applications in medicine," in *IEEE Spectrum*, vol. 34, 1997, pp. 50-56.

[6] G. W. Wornell and A. V. Oppenheim, "Wavelet-Based Representations for a Class of Self-Similar Signals with Application to Fractal Modulation," *IEEE Trans. on Information Theory*, vol. 38, pp. 785-800, 1992.

[7] M. Tzannes, M. Tzannes, and H. L. Resnikoff, "The DWMT: A Multicarrier Transceiver of ADSL using M-Band Wavelet Transforms," Aware Inc. 8 March 1993.

[8] S. F. McCormick and ed, "Multigrid Methods," *SIAM*, 1987.

[9] M. A. Stoksik, R. G. Lane, and D. T. Nguyen, "Accurate Synthesis of Fractional Brownian Motion Using Wavelets," *Electronic Letters*, vol. 30, pp. 383-384, 1994.

[10] P. J. Burt and E. H. Adelson, "The Laplacian Pyramid as a Compact Image Code," *IEEE Trans. Commun*, vol. COM-31, pp. 532-540, 1983.

[11] T. Nguyen and H. Strang, "*Image and Video Compression*," in *Wavelets and Filter Banks*. Wellesley, MA: Wellesley-Cambridge Press, 1996, pp. 365-384.

[12] N. J. Fliege, "*Examples of Wavelet Systems*," in *Multirate Digital Signal Processing*. Baffins Lane, Chichester: John Wiley & Sons Ltd., 1995, pp. 276-283.

[13] N. Hess-Nielsen and V. Wickerhauser, "Wavelets & Time-Frequency Analysis," presented at *Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, 1996.

[14] J. Gotze, J. A. Nossek, and P. Rieder, "Multiwavelet Transform Based on Several Scaling Functions," presented at *Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, 1994.

[15] W. Sweldens, "The Lifting Scheme: A Construction of Second Generation Wavelets," *to appear in SIAM Journal on Mathematical Analysis*, pp. 42, 1996.

[16] T. P. Barnwell and M. J. Smith, "A New Filter Bank Theory for Time-Frequency Representation," *IEEE Trans. on ASSP*, vol. 35, pp. 314-327, 1997.

[17] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*. Englelwood Cliff, NJ: Prentice-Hall, 1993.

[18] M. Vetterli, "A Theory of Multirate Filter Banks," *IEEE Trans. on ASSP*, vol. 35, pp. 356-372, 1987.

[19] A. Croisier, D. Esteban, and C. Galand, "Perfect-channel splitting by use of interpolation/decimation tree decomposition techniques," presented at *Proc. Int. Conf. Inform. Sci. Syst*, 1976.

[20] P. P. Vaidyanathan, "Quadrature Mirror Filter Banks: M-band Extensions and Perfect-reconstruction Techniques," in *IEEE ASSP Magazine*, vol. 4, 1987, pp. 4-20.

[21] H. Nussbaumer, "Pseudo QMF Filter Bank," *IBM Tech. Disclosure Bull.*, vol. 24, pp. 3081-3087, 1981.

[22] R. V. Cox, "The Design of Uniformly and Nonuniformly Spaced Pseudo Quadrature Mirror Filters," *IEEE Trans. ASSP*, vol. ASSP-34, pp. 1090-1096, 1986.

[23] T. Q. Nguyen, "Near-perfect reconstruction pseudo-QMF Banks," *IEEE Trans. Signal Process*, vol. 42, pp. 65-76, 1994.

[24] J. Kovacevic and M. Vetterli, *Wavelets and subband coding*. Englewood Cliffs, N.J.: Prentice Hall PTR, 1995.

[25] C. Herley, K. Ramchandran, and M. Vetterli, "Wavelets, Subband Coding, and Best Bases," presented at *Proceedings of the IEEE*, 1996.

[26] G. Kaiser, *A friendly Guide to Wavelets*. Woodbine, N.J.: Birkhäuser, 1995.

[27] Y. T. Chan, *Wavelets Basics*. Boston: Kluwer academic Publishers, 1995.

[28] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE Transactions on Information Theory*, vol. 36, pp. 961-1005, 1990.

[29] A. Cohen and J. Kovacevic, "Wavelets: The Mathematical Background," presented at Procs. of the IEEE, 1996.

[30] I. Gohberg and S. Goldberg, *Basic Operator Theory*. Boston, MA: Birkhauser, 1981.

[31] M. Vetterli, "Filter banks allowing perfect reconstruction," *Signal Processing*, vol. 6, pp. 97-112, 1986.

[32] S. G. Mallat, "Multiresolution approximations and wavelet orthonormal bases of $L^2(\mathbf{R})$," *Trans. Amer. Math. Soc.*, vol. 315, pp. 69-87, 1988.

[33] W. M. Lawton, "Tight frames of compactly supported wavelets," *Journal of Mathematical Physics*, vol. 31, pp. 1398-1901, 1990.

[34] W. M. Lawton, "Necessary and sufficient conditions for constructing orthonormal wavelet bases," *Journal of Mathematical Physics*, vol. 32, pp. 57-61, 1991.

[35] C. Herley and M. Vetterli, "Wavelets generalized by IIR filter banks," *Center for Telecommunications Research* CU/CTR/TR 206-91-36,.

[36] C. Herley and M. Vetterli, "Wavelets and Recursive Filter Banks," *IEEE Transactions on Signal Processing*, vol. 41, pp. 2536-2556, 1993.

[37] S. Basu, C. H. Chiang, and H. M. Choi, "Wavelets and Perfect Reconstruction Subband Coding with Causal Stable IIR Filters," *IEEE Transactions on Circuits and Systems II*, vol. 42, pp. 24-38, 1995.

[38] O. Rioul and M. Vetterli, "Wavelets and Signal Processing," in *IEEE Signal Processing Magazine*, vol. Oct, 1991, pp. 14-38.

[39] S. K. Mitra and P. P. Vaidyanathan, "Low Passband Sensitivity Digital Filters: A Generalized Viewpoint and Synthesis Procedures," presented at *Proc. of IEEE*, 1984.

[40] T. P. Barnwell and M. J. Smith, "Exact reconstruction techniques for tree-structured subband coders," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-34, pp. 434-441, 1986.

[41] I. Daubechies, "*Ten Lectures on Wavelets*," . Philadelphia, PA: SIAM, 1992.

[42] O. Herrmann, "On the approximation problem in nonrecusive digital filter design," *IEEE Trans. Circuit Theory*, vol. 18, pp. 411-413, 1971.

[43] T. Nguyen and H. Strang, *Wavelets and Filter Banks*. Wellesley, MA: Wellesley-Cambridge Press, 1996.

[44] J. Kovacevic and M. Vetterli, "Nonseparable multidimensional perfect reconstruction filter banks and wavelet bases for $\mathbf{R}^n$," *IEEE Trans. Information. Theory*, vol. 38, pp. 533-555, 1992.

[45] G. Knowles, "VLSI architecture for the discrete wavelet transform," in *Electronics. Letter*, vol. 26, 1990, pp. 1184-1185.

[46] G. Knowles and A. S. Lewis, "VLSI architecture for 2-D Daubechies wavelet transform without multipliers," *Electronics letter*, vol. 27, pp. 171-173, 1991.

[47] "*Aware Wavelet Transform Processor (WTP) Preliminary*," Aware. Inc, Cambridge, M. A. 1991.

[48] T. Nishitani and K. K. Parhi, "VLSI Architectures for Discrete Wavelet Transform," *IEEE Trans. on VLSI systems*, vol. 1, pp. 191-202, 1993.

[49] M. J. Irwin, R. M. Owens, and M. Vishwanath, "VLSI Architectures for the Discrete Wavelet Transform," *IEEE Trans. on circuits and systems II*, vol. 42, pp. 305-316, 1995.

[50] A. Grzeszczak, M. K. Mandal, S. Panchanathan, and T. Yeap, "VLSI Implementation of Discrete Wavelet Transform," *IEEE Transactions on VLSI*, vol. 4, pp. 421-433, 1996.

[51] J. Fridman and E. S. Manolakos, "Discrete Wavelet Transform: Data Dependence Analysis and Synthesis of Distributed Memory and Control Array Architectures," *IEEE Transactions. on Signal Processing*, vol. 45, pp. 569-586, 1997.

[52] J. Fridman and E. S. Manolakos, "Distributed Memory and Control VLSI Architectures for the 1-D Discrete Wvaelet Transform," presented at *IEEE VLSI Signal Process VII*, 1994.

[53] A. Aho, R. Sethi, and J. D. Ullman, *Compilers: Principles, Techniques, and Tools*. M.A.: Addison-Wesley, 1986.

[54] A. Dembo and D. Malah, "Statistical design of analysis/synthesis systems," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 36, pp. 328-341, 1989.

[55] C. Cadwell, T. Lookabaugh, and M. Perkins, "Analysis/synthesis systems in the presence of quantization," presented at *Proc. IEEE Int. Symp. Circ. and Syst*, 1989.

[56] T. Kronander, "New criteria for optimization of QMF banks to be used in an image coding system," presented at *Proc. IEEE Int. Symp. Circ. and Syst*, 1989.

[57] J. Biemond, D. E. Boekee, and P. H. Westerink, "Scalar Quantization Error Analysis for Image Subband Coding Using QMF's," *IEEE Trans. on Signal Processing*, vol. 40, pp. 421-428, 1992.

[58] J. Kovacevic, "Eliminating correlated errors in subband/wavelet coding systems with quantization," presented at *Proc. Asilomar Conf. Signal Syst., Comput.*, 1993.

[59] R. A. Haddad and N. Uzun, "Modeling and Analysis of Quantization Errors in Two Channel Subband Filter Structures," presented at *SPIE Visual Communications and Image Processing*, 1992.

[60] R. A. Haddad and N. Uzun, "Cyclostationary modeling, analysis and Optimal Compensation of Quantization Errors in Subband Codecs," *IEEE Trans. on Signal Process.*, vol. 43, pp. 2109-2119, 1995.

[61] D. Y. Chan, Y. B. Chen, and J. F. Yang, "Fast and Low Round-off Implementation of Quadrature Mirror Filters for Subband Coding," *IEEE Trans. on Circuit and Systems for Video Tech.*, vol. 5, pp. 524-532, 1995.

[62] B. Gold and C. M. Rader, *Digital Processing of Signals*. New York: McGraw-Hill Book, 1969.

[63] A. V. Oppenheim, A. S. Willsky, and I. T. Young, *Signal and System*. Englewood Cliffs, N.J.: Prentice Hall, 1983.

[64] P. M. Ebert, J. E. Mazo, and M. C. Taylor, "Overflow Oscillations in Digital Filters," *Bell System Technical Journal*, vol. 48, pp. 2999-3020, 1969.

[65] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Englewood Cliffs, N.J.: Prentice Hall, 1989.

[66] A. H. Gray and J. D. Markel, *Linear Prediction of Speech*. New York: Springer-Verlag, 1976.

[67] B. Friedlander, "Lattice Filters for Adaptive Processing," presented at Proc. *IEEE*, 1982.

[68] P. P. Vaidyanathan, "Passive Cascaded-Lattice Structures for Low-Sensitivity FIR Filter Design, with Applications to Filter Banks," *IEEE Transactions on Circuits and Systems*, vol. CAS-33, pp. 1045-1064, 1986.

[69] P.-Q. Hoang and P. P. Vaidyanathan, "Lattice Structures for Optimal Design and Robust Implementation of Two-Channel Perfect Reconstruction QMF Banks," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, pp. 81-94, Jan 1988.

[70] P. Duhamel and O. Rioul, "Fast Algorithms for Discrete and Continuous Wavelet Transform," *IEEE Trans. on Information Theory*, vol. 38, pp. 569-586, 1992.

[71] O. Egger and W. Li, "Subband Coding of Images Using Asymmetrical Filter Banks," *IEEE Trans Image Processing*, vol. 4, pp. 478-485, 1995.

[72] E. Abdel-Raheem, F. El-Guibaly, and T. Yeap, "A simple approach for the design of pseudo-QMF banks," *Can. J. Elec. & Comp. Eng.*, vol. 21, pp. 153-157, 1996.

[73] V. C. Hamacher, Z. G. Vranesic, and S. G. Zaky, *Computer Organization*, 3rd ed. Toronto: McGraw Hill, 1990.

[74] A. G. Dempster and M. D. Macleod, "Use of Minimum-Adder Multiplier Blocks in FIR Digital Filters," *IEEE Trans. Circuit and System II*, vol. 12, pp. 569-577, 1995.

[75] K. K. Parhi and H. R. Srinivas, "A Fast VLSI Adder Architecture," *IEEE Journal of Solid State Circuits*, vol. 7, pp. 761-767, 1992.

[76] J. Fadavi-Ardekani, "$M \times N$ Booth encoded multiplier generator using optimized Wallace trees," *IEEE Trans. On VLSI*, vol. 1, pp. 120-125, 1993.

[77] A. R. Calderbank, I. Daubechies, W. Sweldens, and B. Yeo, "Wavelet Transforms that Map Integers to Integers," , 1996.

[78] J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Computation of Complex Fourier Series," *Mathematics of Computation*, vol. 19, pp. 297-301, 1965.

[79] C. S. Burrus, *Efficient Fourier Transform and Convolution Algorithms*. Englewood Cliffs, N.J.: Prentice Hall, 1988.

[80] I. J. Good, "The Interaction Algorithm and Practical Fourier Analysis," *J. Royal Stat. Soc.*, vol. B-20, pp. 361-372, 1958.

[81] S. Winograd, "On Computing the Discrete Fourier Transform," *Mathematics of Computation*, vol. 32, pp. 175-199, 1978.

[82] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, pp. Jan, 1974.

[83] Z. Wang, "Fast algorithms for the discrete W transform and for the discrete Fourier transform," *IEEE Trans. Acoustic, Speech and Signal Processing*, vol. ASSP-32, pp. 803-816, 1984.

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

[85] W. C. Miller and B. D. Tseng, "On computing the discrete cosine transform," *IEEE Trans, Comput.*, vol. C-27, pp. 966-968, 1978.

[86] P. Duhamel, "Implementation of 'Split-Radix' FFT algorithm for complex, real and real-symmetric data," *IEEE Trans. Acoust., Speech, and Signal Process.*, vol. ASSP-34, pp. 285-295, 1986.

[87] F. A. Kamangar and K. R. Rao, "Fast algorithms for the 2D-discrete cosine transform," *IEEE Trans. Comput.*, vol. C-31, pp. 899-906, 1982.

[88] N. Nasrabadi and R. King, "Computationally efficient discrete cosine transform algorithm," *Electronics Letter*, vol. 19, pp. 24-25, 1983.

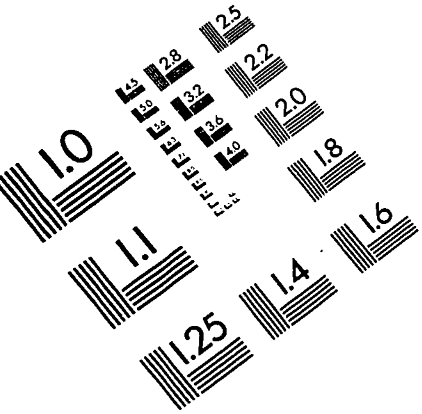
[89] K. R. Rao and P. Yip, "Fast DIT algorithms for DCT and DST," *Circuits, Systems and Signal Process.*, vol. 3, pp. 387-408, 1984.

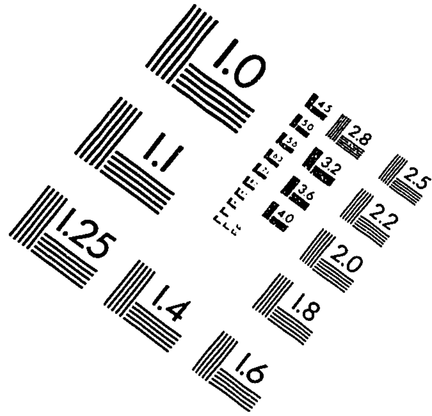
[90] K. R. Rao and P. Yip, "DIF algorithms for DCT and DST," presented at *Intl. Conf. on Acoust., Speech, and Signal Process.*, Tampa, F.L., 1985.

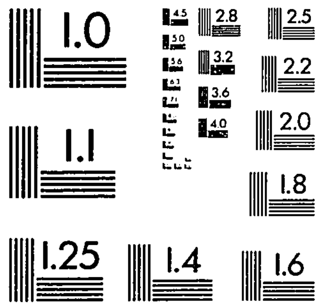
[91] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*. San Diego: Academic Press, Inc., 1990.

[92] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Boston: Kluwer Academic Publisher, 1992.

IMAGE EVALUATION TEST TARGET (QA-3)

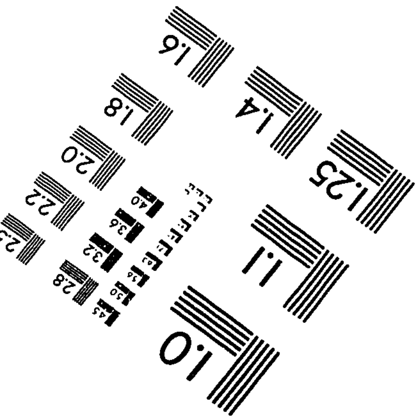


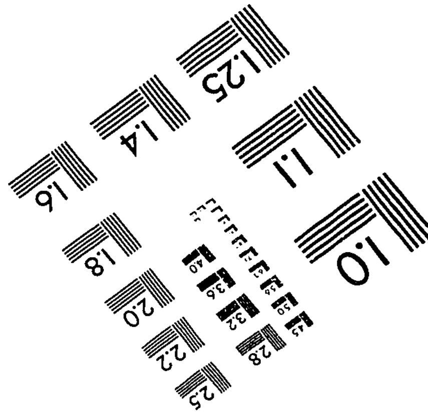




150mm

6″





APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989