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.

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.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

NEW WAVELET TRANSFORMS AND THEIR APPLICATIONS TO DATA COMPRESSION

by

INDERPREET SINGH
M.Tech., Indian Institute of Technology, Delhi, INDIA, 1993
B.E.(Hons.), Panjab University, INDIA, 1992

A Dissertation Submitted in Partial Fulfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Electrical and Computer Engineering

We accept this dissertation as conforming
to the required standard

Dr. P. Agathoklis, Supervisor, Dept. of Elect. & Comp. Eng.

Dr. A. Antoniou, Supervisor, Dept. of Elect. & Comp. Eng.

Dr. W-S. Lu, Member, Dept. of Elect. & Comp. Eng.

Dr. R. Illner, Outside Member, Dept. of Mathematics and Statistics

Dr. F. Kossentini, External Examiner, Dept. of Elect. & Comp. Eng., UBC

Supervisors: Dr. P. Agathoklis and Dr. A. Antoniou

ABSTRACT

With the evolution of multimedia systems, image and video compression is becoming the key enabling technology for delivering various image/video services over heterogeneous networks. The basic goal of image data compression is to reduce the bit rate for transmission and storage while either maintaining the original quality of the data or providing an acceptable quality.

This thesis proposes a new wavelet transform for lossless compression of images with application to medical images. The transform uses integer arithmetic and is very computationally efficient. Then a new color image transformation, which is reversible and uses integer arithmetic, is proposed. The transformation reduces the redundancy among the red, green, and blue color bands. It approximates the luminance and chrominance components of the YIQ coordinate system. This transformation involves no floating-point/integer multiplications or divisions, and is, therefore, very suitable for real-time applications where the number of CPU cycles needs to be kept to a minimum.

A technique for lossy compression of an image data base is also proposed. The technique uses a wavelet transform and vector quantization for compression. The discrete cosine transform is applied to the coarsest scale wavelet coefficients to achieve even higher compression ratios without any significant increase in computational complexity. Wavelet denoising is used to reduce the image artifacts generated by quantizing the discrete cosine transform coefficients. This improves the subjective quality of the decompressed images for very low bit rate images (less than 0.5 bits per pixel).

The thesis also deals with the real-time implementation of the wavelet transform. The new wavelet transform has been applied to speech signals. Both lossless and lossy techniques for speech coding have been implemented. The lossless technique involves using the reversible integer-arithmetic wavelet transform and Huffman coding to obtain the compressed bitstream. The lossy technique, on the other hand, quantizes the wavelet coefficients to obtain higher compression ratio at the expense of some degradation in sound quality. The issues related to real-time wavelet compression are also discussed. Due to the limited size of memory on a DSP, a wavelet transform had to be applied to an input signal of finite length. The effects of varying the signal length on compression performance are also studied for different reversible wavelet transforms. The limitations of the proposed techniques are discussed and recommendations for future research are provided.

Examiners:

Dr. P. Agathoklis, Supervisor, Dept. of Elect. & Comp. Eng.

Dr. A. Antoniou, Supervisor, Dept. of Elect. & Comp. Eng.

Dr. W-S. Lu, Member, Dept. of Elect. & Comp. Eng.

Dr. R. Illner, Outside Member, Dept. of Mathematics and Statistics

Dr. F. Kossentini, External Examiner, Dept. of Elect. & Comp. Eng., UBC

Table of Contents

List of Figures

List of Tables

List of Abbreviations

1D	One dimensional
2D	Two dimensional
bpp	Bits per pixel
BRD	Delayed branch instruction
BWT	Block wavelet transform
HVS	Human visual system
CPU	Central processing unit
CR	Compression ratio
CREW	Compression with reversible embedded wavelets
CT	Computerized tomographic (images)
DCT	Discrete cosine transform
DFT	Discrete Fourier transform
DM	Daughter module
DMA	Dual-memory access
DP	Data page pointer
DPRAM	Dynamic programmable random-access memory
DSP	Digital signal processing (processor)
EZW	Embedded zerotree wavelet (coding)
JBIG	Joint Bilevel Image Group
JPEG	Joint Photographic Experts Group
KLT	Karhunen-Loeve transform
LBG algorithm	Linde Buzo Gray algorithm
MRI	Magnetic resonant imaging
MSE	Mean squared error
NINT	Nearest integer (less than or equal to)
PSNR	Peak signal-to-noise ratio
QMF	Quadrature mirror-image filter
RAM	Random-access memory
RGB	Red-green-blue co-ordinate system

ROM	Read-only meomory
RPTB	Repeat block instruction
RPTS	Repeat sigle instruction
S+P	Said + Pearlman
SPIHT	Set partitioning in hierarchical trees
SRAM	Static random-access memory
STFT	Short-time Fourier transform
TS transform	Two-six transform
TT transform	Two-ten transform
USC	University of Southern California
VoIP	Voice over Internet protocol
VQ	Vector quantization
WHT	Walsh-Hadamard transform

Notation

- The symbols $\mathcal{Z}$, $\mathcal{R}$, and $\mathcal{C}$ denote the sets of integers, real numbers, and complex numbers, respectively.
- $\mathcal{L}^p(\mathbf{R})$ is a real inner product space of measurable functions such that

$$\int_{-\infty}^{\infty} |f(x)|^p \, dx \; < \; +\infty$$

- The classical norm of $f(x) \in \mathcal{L}^1(\mathbf{R})$ is given by

$$\|f\|_{\mathcal{L}^1} \; = \; \int_{-\infty}^{\infty} |f(x)| \, dx$$

- $\mathcal{L}^2$ norm is given by

$$\|f\|^2 \; = \; \int_{-\infty}^{\infty} |f(x)|^2 \, dx$$

- $\mathcal{L}^2(\mathbf{R}^n)$ is an inner product space of measurable square integrable n-dimensional functions.
- The classical norm of $f(x_1, \, x_2, \, \ldots, \, x_n) \in \mathcal{L}^2(\mathbf{R}^n)$ is given by

$$\|f\|^2 \; = \; \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} |f(x_1, \, x_2, \, \ldots, \, x_n)|^2 \, dx_1 \, dx_2 \, \ldots \, dx_n$$

- The series $V_0, \, V_1, \, V_2, \, \ldots$ refers to the sequence of subspaces in $\mathcal{L}^2(\mathbf{R})$.
- The symbols '$\oplus$' and '$\otimes$' refer to the tensor sum and tensor product of two subspaces, respectively.
- The symbol j denotes the quantity $\sqrt{-1}$.
- Functions of a continuous variable are indicated with round parenthesis, for example, $f(t)$ where $t \in R$.
- Functions of a discrete variable are indicated with square parenthesis, for example, $x[n]$ where $n \in Z$.
- Bold face symbols are usually used to represent vector or matrix variables.
- The quantity $\mathbf{A}^T$ denotes the transpose of $\mathbf{A}$.
- The quantity $\mathbf{A}^*$ denotes the conjugate of $\mathbf{A}$.

- The z-transform of a discrete sequence $x[n]$ is denoted as $X(z)$ and is defined as

$$Zx[n] = X(z) = \sum_{n=-\infty}^{n=\infty} x[n]\ z^{-n} \tag{0.1}$$

- The inner product of two functions in the continuous variable case is defined as

$$< f(t), g(t) > = \int_{-\infty}^{+\infty} f(t)g^*(t)dt \tag{0.2}$$

  and in the discrete variable case as

$$< f[n], g[n] > = \sum_{n=-\infty}^{n=\infty} f[n]g^*[n] \tag{0.3}$$

- The delta function $\delta[n]$ is defined as

$$\delta[n] = \begin{cases} 1 & \text{for } n = 0 \\ 0 & \text{otherwise} \end{cases} \tag{0.4}$$

- The notation $f * g$ stands for convolution of f and g
- The notation $\lfloor x \rfloor$ denotes the greatest integer not more than x (or equivalently x is rounded to the nearest integer towards $-\infty$).
- Filter coefficients are assumed to be real unless stated explicitly.
- The sampling period of discrete sequences is assumed to be one unless stated otherwise.
- A filter H has a transfer function $H(z)$. That is, roman font is used to name a particular filter while italics are used to indicate the transfer function associated with it.

Acknowledgement

I would like to express my deepest gratitude to my supervisors, Dr. Panajotis Agathoklis and Dr. Andreas Antoniou, for their valuable guidance and comments throughout my graduate study. I would also like to thank Dr. Reinhard Illner for all the valuable discussions we had on the mathematical theory of wavelets. His graduate course on wavelets was really helpful.

I would like to thank my wife, Meher, and my brother, Jasjeet, for their emotional support and encouraging me to complete this work.

In addition I would like to mention some important friends who became an integral part of my student life. They include Avijit Bhunia, Sandeep Agarwal, Ranganathan Gurunathan and Subramanian Muthu, to name a few.

Finally, I would like to thank my mom and my dad for all the sacrifices they made for me and for giving me the will to continue.

Dedication

Dedicated to my parents

Chapter 1

Introduction

Traditional image compression techniques have been designed to exploit the statistical redundancy present within real world images. The discrete cosine transform is one example of this statistical approach. Removing redundancy can only give a limited amount of compression; to achieve high memory savings, some of the non-redundant information must be removed. By using methods that closely mimic the human visual system (HVS), compression can take into account the importance of each individual coefficient and code accordingly. Psychophysicists and visual psychologists have discovered that the eye filters the image into a number of bands, each approximately one octave wide in frequency. Further, in the spatial domain, the image should be considered to be composed of information at a number of different scales. Wavelets decompose the image into multiple bands at octave frequencies similar to the multiple channel models of the HVS. New compression techniques based on the wavelet transform are getting extremely popular these days. Wavelet technology can help attain higher transmission speeds and clearer pictures than the conventional statistical methods by providing higher compression ratios and reduced computational complexity.

1.1 Historical overview

Wavelets, filter banks, and multi-resolution signal analysis, which have been used independently in the field of applied mathematics, signal processing, and computer vision, respectively, have recently converged to form a single theory. The first wavelet system was constructed by Haar [1] in 1910. The Haar wavelet system, as it is now known, uses piecewise constant functions as its basis to decompose the input signal. Although wavelet theory has many close ties to science and engineering since the early 1900s, these linkages were not discovered until the early 1980s. Until that time wavelet theory was a disjoint set of ideas that lacked a unified framework.

The vast majority of developments in the wavelet theory and applications occurred in the

mid 1980s. In 1984, the term 'wavelet' was introduced by Grossman and Morlet [2]. In 1988, a tremendous breakthrough in wavelets was brought about by Daubechies [3]. In this classic paper, Daubechies introduced a family of compactly supported wavelet systems that satisfy the property of orthogonality and perfect reconstruction. In 1989, Mallat [4] presented the theory of multiresolution analysis that later became popular as the Mallat algorithm. This algorithm provided an easy insight to engineers and physicists in applying wavelet concepts to diverse fields such as signal processing etc. In 1992, Cohen, Daubechies and Feauveau [5] established the theory of biorthogonal wavelet filters commonly known as the CDF filters. Unlike orthogonal wavelet filters (except for the trivial case of the Haar and other Haar-like transforms), biorthogonal filters allow for symmetric finitely-supported basis functions. The symmetry property can offer significant benefits in many applications, image data compression being one of them. Wavelet transforms have proven to be extremely useful for image coding as many researchers have shown (e.g., [6]-[23]). Consequently such transforms have been used extensively in many image data compression algorithms. In 1993, Shapiro [9] introduced the concept of embedded coding. His coding scheme, called *embedded zerotree wavelet* (EZW) coding, was based on wavelet transforms. Due to the obvious advantages of the embedded property in many applications, embedded coding quickly grew in popularity especially as a means of building unified lossless/lossy compression systems. In 1995, Zandi et al. [10] proposed *compression with reversible embedded wavelets* (CREW), a reversible embedded image compression system based on some ideas of Shapiro. Not long after, in 1996, Said and Pearlman [11] introduced a new coding scheme known as *set partitioning in hierarchical trees* (SPIHT), which is conceptually similar to EZW with some implementation improvements.

Apart from the embedded coding techniques which involve using highly complex coders and decoders, there are other wavelet-based techniques that became popular in certain areas of image compression. One of them is known as vector quantization (VQ). Although the concept of vector quantization has been around for decades [12], applying VQ to a wavelet transform was introduced by Barlaud et al. in 1989 [13]. Modifications to the VQ approach were later reported in various other publication [14], [15].

1.1.1 Need for compression

The need of compression emerges from the fact that any real data gathered through a data acquisition system always has some degree of correlation. The degree of correlation depends on the type of data and the amount of noise in the data. Compression is widely used in

many areas, e.g., image processing, speech coding, medicine and the Internet, to name a few. Even though data memories are getting cheaper and faster, there will be cases where compression of data is necessary. For example:

1. To store a moderately large image, say a 512×512 pixels, 24-bit color image, requires about 0.75 MBytes. A video signal typically has 30 frames per second.
2. A standard 35 mm photograph digitized at 12 μm resolution requires about 18 MBytes.
3. One second of NTSC color video entails 23 MBytes.

This shows that one can easily find situations where the current hardware is inadequate, either technically or economically. Compression techniques can reduce this gap by

- saving storage,
- saving CPU time, or by
- saving transmission time.

The requirement for compression has become even more critical with the boom in Internet. Various multimedia protocols such as H.323 and H.324 have built-in compression of audio and video signals to minimize the bandwidth per channel. Some of the Internet applications such as voice over Internet protocol (VoIP) use state of the art audio coders such as G.728 and G.729A [16] [17].

The major requirement of compression is that one should be able to quickly switch between the original and the compressed data.

1.2 Contributions of the thesis

This thesis studies reversible wavelet transforms and their application to lossless compression of speech and image signals. The ideas are drawn from all of the previously described developments, but of most direct importance are the reversible wavelet transforms proposed by Boliek *et al.* [10] and the S+P transform proposed by Said and Pearlman [11]. Many of the ideas on reversible wavelets are closely related to ideas presented in these two papers. Apart from lossless compression, wavelets have also been used for lossy compression of images. Many existing wavelet-based lossy compression techniques are based on embedded coding techniques [9], [11]. In this thesis, focus is directed towards using vector quantization [12] and wavelets for lossy compression. Some of the topics addressed in detail are as follows:

- Lossless Compression

  - a new reversible transform and comparison with existing transforms with focus on medical image compression
  - extension of the reversible transformations to the class of color images
  - practical issues associated with the implementation of transforms such as computational complexity
- Lossy Compression
  - a mixed transform technique based on the discrete wavelet transform and the discrete cosine transform with vector quantization technique
  - methods of minimizing quantization artifacts using post processing
- Practical Implementation
  - practical issues related to real-time implementation of reversible transform techniques
  - effects of block wavelet transform on compression performance and computational complexity

1.3 Thesis organization

The thesis is divided into six chapters. The first two chapters provide introductory and background material necessary to understand the rest of the thesis. The remaining chapters present a combination of fundamentals and research results in detail.

In Chapter 2, concepts underlying multirate filter banks and wavelet transforms are outlined. The chapter begins by presenting the fundamentals of multirate filter banks and wavelet-based decomposition. A special class of wavelets called reversible wavelets [10] is described. This is followed by methods of quantizing the wavelet coefficients. Two different quantization techniques, namely, scalar quantization and vector quantization, are discussed. Finally, different types of encoders are discussed. The advantages of wavelet coders over other entropy coders are also discussed.

In Chapter 3, a new reversible transform is proposed. The transform is used for the compression of medical images. The performance achieved is compared with that achieved with other existing lossless compression techniques. The transform is then extended to compress color images. A new reversible color image transformation to remove spectral redundancy among the color bands is also presented. Results obtained with the new transform are compared with results obtained with some of the existing reversible transforms.

In Chapter 4, a technique for lossy compression of images is presented. The technique

uses wavelets and vector quantization for the compression of the wavelet coefficients and DCT with scalar quantization for encoding the scaling coefficients. The results are compared with results obtained with state of the art wavelet coders such as the EZW [9] and SPIHT [11] coders. Subjective results are also given.

In Chapter 5, the proposed reversible wavelet transform is used for the compression of speech signals. Real-time implementation was done on the TMS320C30 DSP. Issues concerning the real-time wavelet transform (called *block wavelet transform*) are also discussed. Both lossless and lossy speech coding are implemented. The results are compared with results obtained with some of the existing coders.

Finally, Chapter 6 summarizes some of the more important contributions and results presented in the thesis. The chapter concludes by putting forward some suggestions and directions for future work.

Chapter 2

Overview of Wavelet-Based Compression Methods

2.1 Introduction

Coding is the process of finding a representation of a given signal. It is usually used to either find a representation with less redundancy so that fewer bits are required to encode the given signal, or to add redundancy in order to facilitate error detection/correction of the signal transmitted over noisy channels. The former coding approach is also called *compression*. With the boom in multimedia and high-resolution video containing a high degree of redundancy, compression has gained widespread attention in both research and industry.

There are essentially two basic kinds of compression schemes: lossless and lossy. In the case of lossless compression, one is interested in reconstructing the data exactly, without any loss of information. Lossless compression is often used for compressing text files and medical images. Lossy compression, on the other hand, can tolerate errors in the image as long as the quality of the signal is 'acceptable' after compression. Although the term 'acceptable' is subjective, an acceptable approximation of the signal is one that is, for practical purposes, visually indistinguishable from the original signal.

The general structure of an image compression system is as shown in Figure 2.1. In this diagram $x[i,j]$ represents the original image. The transform is applied to the original image to reduce the spatial or spectral correlation or both, to generate the transformed image $y[k,l]$. The transformed image is quantized to obtain a quantized image $y_q[k,l]$. Quantization is used to discard any coefficients deemed to be insignificant. This results in loss of information content from the image and thus the compression is lossy. In the case of lossless compression, no quantization is performed since no loss of information can be

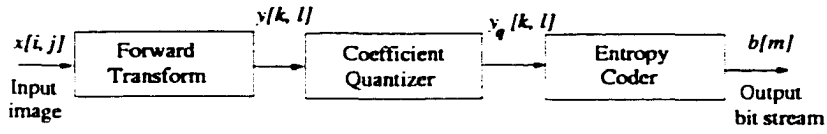


Figure 2.1. *General structure of transform-based image compression system.*

tolerated.

The quantized image is encoded to obtain the final compressed bit stream $b[m]$. There are various types of encoders and *entropy encoder* is one of them. Entropy is a measure of the amount of information in an object. For example, a speech signal with periods of silence will have lower entropy than a signal with no periods of silence. Entropy can be expressed in bits. In this form, it is generally referred to as *information content*. Every non-random signal has some degree of redundancy or correlation in itself. By an appropriate prediction algorithm, the redundancy can be removed thus reducing the entropy of the signal as well (entropy is directly proportional to information content). As reduction in entropy reduces the average number of bits per sample of the signal, it results in the overall compression of the signal. An encoder using this concept of reducing entropy to achieve compression is an entropy encoder. The entropy is reduced by using the fact that a symbol with higher probability of occurrence should use fewer bits compared to a symbol with lower probability of occurrence. At the receiver, the coded bit stream is decoded using an equivalent entropy decoder followed by an inverse transformation to obtain the decompressed image.

In the sections that follow, each of these two or three stages (for lossless or lossy compressions, respectively) in a compression system are discussed in detail.

2.2 The transformation stage

The aim of compression is to remove redundancy from the original data so that the same information (or slightly less) can be coded in a fewer bits. Correlated data is characterized by the fact that one can, given a part of the data, fill in the missing part. Several types of correlation exist. They can be characterized as follows:

- Spatial correlation: One can often predict the value of a pixel in an image by looking at the neighboring pixels.
- Spectral correlation: One can predict one frequency component by looking at the neighboring frequency components. This means that a spectral transform (Fourier transform, for example) of a signal is smooth.

- Temporal correlation: In a digital video, most pixels of two neighboring frames change very little in the time direction (e.g., the background).

Generally a transformation remove either spatial and/or spectral correlation in an image. Techniques that remove both spectral and spatial correlation are called hybrid coding techniques.

2.2.1 Spatial decorrelating transforms

A spatial decorrelating transform removes the correlation from an image by using the neighboring pixel values. A *predictor* is used to estimate the current value of the pixel based on previous pixels. Different types of predictors have been reported in the literature. A compression scheme based on this approach is called predictive coding. The current pixel value is estimated based on the neighboring pixels and the type of predictor being used. The difference between actual and estimated value is called the prediction error (or error residual). For lossless compression, the prediction error is stored without any loss using entropy coding and is used to obtain the final bit stream. On the other hand, if the compression is lossy, the error residual is quantized and then coded to obtain the bit stream. The predictor can also be adaptive in which case its coefficients change with the type of signal being coded. The Joint Photographic Experts Group, also known as JPEG, uses a variety of lossless predictors [18] for predicting the current pixel value based on the neighboring pixels. Generally the predicted value of the current pixel is estimated by using the past pixel values in the rows and columns. The more the number of neighboring pixels used for prediction, the higher is the order of the predictor. A further possibility is to delay the encoding of a pixel until the 'future trend' of the signal can be observed, and then take advantage of this trend. This is called delayed coding [19].

The Joint Bilevel Image Group, also known as JBIG [20], uses a set of predictors to estimate the current pixel value. Although JBIG is generally for the coding of binary images, it can efficiently code images with resolution up to 4 bits/pixel. However, beyond that, its performance tends to suffer compared to other transform-based methods. The main advantage of the predictive coding technique is that it involves very few computations compared to spectral-correlation and hybrid coding methods. Various applications where predictive coders are used can be found in [21], [22].

2.2.2 Spectral-decorrelating transforms

In spectral-decorrelating transforms, the input image is partitioned into blocks of pixels and each block of pixels is processed separately. The pixels in each block are transformed in the frequency domain (spatial frequency in rows and columns) using a linear-orthonormal transformation. A linear transformation is an information preserving process. The main objective of a transformation is to produce statistically independent (or at least uncorrelated) transform coefficients so that they can be coded independently of each other with good efficiency. Another objective is energy compaction, which means concentrating the energy to a few coefficients and thereby making as many coefficients as possible small enough that they need not be transmitted. Most of the spectral transforms achieve these objectives reasonably well. The most commonly known transforms are the discrete Fourier transform (DFT), the Walsh-Hadamard transform (WHT), the Karhunen-Loeve transform (KLT) and the discrete cosine transform (DCT) to name a few [19]. Although the KLT achieves the theoretical limit of performance of any decorrelation technique (in other words, its coefficients are uncorrelated), the DCT provides a good balance between performance and computational complexity. Hence it is most widely used, and in certain circumstances, its performance can come close to that of the KLT [8]. Recently, subband coding has also been used for image compression. This involves filtering the image through a bank of filters. The output of each of the filters has information within the passband of the filter. The advantage of subband coding is that the quantization of each of the subbands can be done based on human perception, thus improving the subjective quality of the decompressed image. A specific family of transforms used to obtain a type of subband decomposition of signals, called *wavelets*, has received considerable attention in recent years [3]. Wavelets can be very closely related to subband filters, with some additional properties, that make them favorable for image compression [23]. One of the main features of wavelets is their *time-frequency representation*. Wavelets have higher frequency resolution at lower frequencies and lower frequency resolution at higher frequencies. This is attributed to the nonuniform bandwidths of the filters used at each spatial resolution. Figure 2.2(a) shows the time-frequency representation for uniform filter banks. The frequency resolution is constant at different frequencies. The same also holds for time resolution. This is the characteristic of the short-time Fourier transform (STFT). Consider a signal $x(t)$ and assume that it is stationary over an arbitrary window function $h(t)$ of limited duration, centered at time location τ. The Fourier transform of the windowed signals $x(t)\, h^*(t - \tau)$ yields the STFT

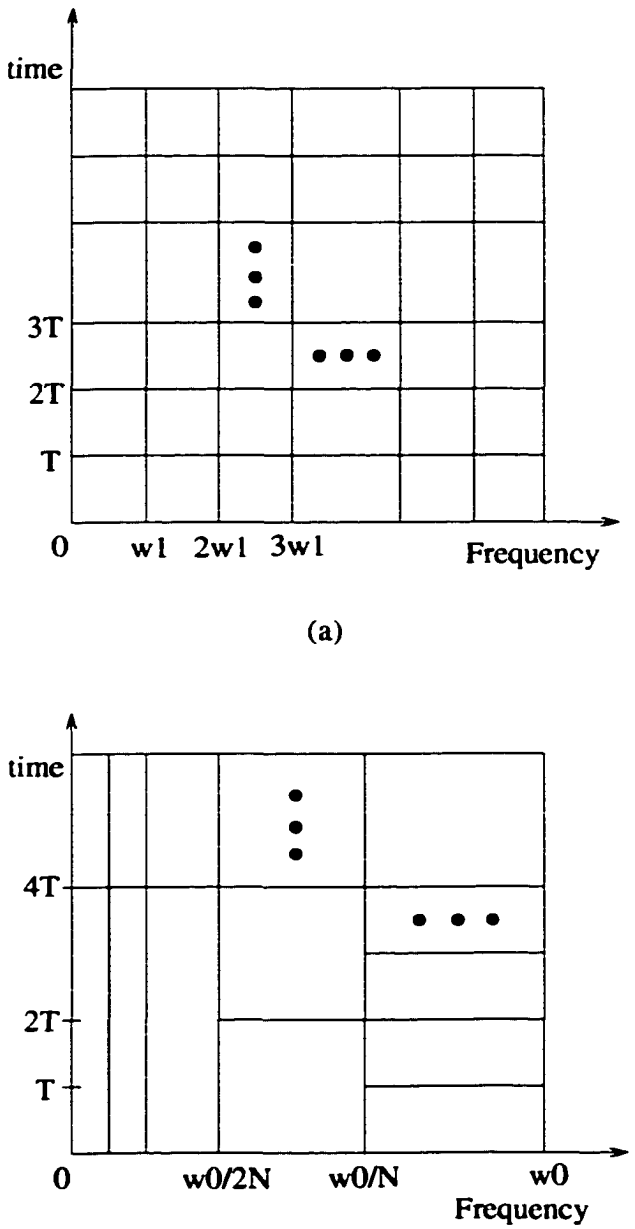


Figure 2.2. *(a) Uniform bandwidth filter banks. (b) Wavelet filter banks.*

as

$$\text{STFT}(\tau, f) = \int_{-\infty}^{\infty} x(t)\, h^*(t-\tau)\, e^{-2j\pi ft}\, dt \tag{2.1}$$

which maps the signal onto a two-dimensional function in the time-frequency plane. However, the analysis depends critically on the choice of the window $h(t)$. Figure 2.2(b) shows the time-frequency representation for nonuniform filter banks (e.g., wavelets). The frequency resolution is higher at lower frequencies and lower at lower frequencies. On the other hand, time resolution is higher at higher frequencies and lower at higher frequencies. The transformations that use both frequency and spatial correlation in an image fall in the the category of so called ***hybrid transforms.*** In the next section, we discuss the basic theory of wavelets and provide details on the current trends.

2.2.3 Hybrid transforms: Wavelets

Time-frequency representation, also called ***multiresolution representation***, is a general method for constructing orthonormal bases, developed by Mallat and Meyer [24]. Intuitively, multiresolution slices the space of square integrable functions $\mathcal{L}^2$ into a nested sequence of subspaces V_i, where each V_i corresponds to a different scale. The multiresolution is completely determined by the choice of a special function called the *scaling function.*

2.2.4 Definition of multiresolution analysis

Multiresolution analysis provides a natural framework for understanding the wavelet basis, and for the construction of new examples. The concept of multiresolution approximation of a function was introduced by Mallat [4] and provides a powerful framework to understand wavelet decomposition and reconstruction. Before the theory of multiresolution analysis is presented, it is worthwhile to discuss the concept of orthonormal basis. This concept is widely used in the development of multiresolution analysis.

2.2.4.1 Orthonormal basis

A family of elements $\{u_n\}$, $n \in \mathbf{I}$, ($\mathbf{I}$ countable index set), is called orthonormal if $\forall n \in \mathbf{I}$, $\|u\| = 1$ and

$$< u_i,\ u_j > \ = \delta_{ij}\ ,\ \ \forall i,\ j \in \mathbf{I} \tag{2.2}$$

where $< u_i,\ uj >$ is the *inner product* of u_i and u_j as given in Equation 0.3. Then $\{u_n\}_{n\in\mathbf{I}}$ is called an orthonormal set. An orthonormal set is *complete* if it satisfies the property

of *completeness.* An orthonormal set of functions, $\{u_n\}$, $n \in \mathbf{I}$, ($\mathbf{I}$ countable index set) belonging to $\mathcal{L}^2(\mathbf{R})$ space, is said to be complete if there exists no functions different from zero in $\mathcal{L}^2(\mathbf{R})$ which is orthogonal to all functions u_n. A complete orthonormal set is also called an orthonormal basis (or standard basis). Background theory on completeness can be found in [25].

As a consequence of this property of completeness, for any arbitrary function u in $\mathcal{L}^2(\mathbf{R})$, if $< u, u_n > \ = 0$ for all $n \in \mathbf{I}$, u must be zero. In this case, u can be represented as

$$u = \sum_{n \in \mathbf{I}} < u, u_n > u_n \tag{2.3}$$

and

$$\|u\|_H^2 = \sum_{n \in \mathbf{I}} | < u, u_n > |^2 \tag{2.4}$$

Equation 2.3 is the generalized form of the Fourier series with any arbitrary orthonormal basis $\{u_n\}$, $n \in \mathbf{I}$.

Examples

The functions e^{ikx}, $k \in \mathbf{I}$ form a complete orthogonal set in $\mathcal{L}^2[0, 2\pi]$. Another example of orthonormal basis in $\mathcal{L}^2(\mathbf{R})$ is

$$f_{m,n}(x) = 2^{-m/2} f(2^{-m}x - n) \tag{2.5}$$

where

$$f(x) = \begin{cases} 1, & 0 \leq x < 1/2 \\ -1, & 1/2 \leq x < 1 \\ 0, & \text{otherwise} \end{cases}$$

2.2.4.2 Scaling function and the mother wavelet

A multiresolution analysis consists of a sequence of successive approximation spaces V_j (closed subspaces) that satisfy the relation

$$\cdots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \cdots \tag{2.6}$$

with

$$\overline{\bigcup_{j \in \mathbf{I}} V_j} = \mathcal{L}^2(\mathbf{R}) \tag{2.7}$$

and

$$\bigcap_{j \in \mathbf{I}} V_j = \{0\} \tag{2.8}$$

If P_j is the orthogonal projection operator onto V_j, then

$$\lim_{j \to -\infty} P_j f = f \qquad \forall f \in \mathcal{L}^2(\mathbf{R}) \tag{2.9}$$

An additional requirement for the multiresolution analysis is

$$f(t) \in V_j \Leftrightarrow f(2^j t) \in V_0 \tag{2.10}$$

In other words, all the spaces are scaled versions of the control space V_0. Moreover,

$$f(t) \in V_0 \Rightarrow f(t - n) \in V_0 \text{ for all } n \in I \tag{2.11}$$

In summary, the five conditions for a set of subspaces $V_j \subset \mathcal{L}^2(\mathbf{R})$ to be suitable for multiresolution analysis are

1. $\cdots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \cdots$
2. $\overline{\bigcup_{j \in \mathbf{I}} V_j} = \mathcal{L}^2(\mathbf{R})$,
   $\bigcap_{j \in \mathbf{I}} V_j = \{0\}$
3. $f(t) \in V_j \Leftrightarrow f(2t) \in V_{j-1}$
4. There exists $\phi \in L^2$ such that $V_0 = \{f \mid f(t) = \sum_i \alpha_i \phi(t - i)\}$,
   ϕ is also known as the *scaling function*.
5. $\{\phi_n(t) = \phi(t - n)\}$ is an orthonormal basis of V_0.

These shall be referred as the five axioms of multiresolution analysis. If any orthonormal basis satisfies all of the above conditions, it could be a valid wavelet basis for multiresolution analysis.

The basic tenet of multiresolution analysis is that whenever collection of closed subspaces satisfies axioms 1-5 of multiresolution analysis, then there exists an orthonormal wavelet basis $\{\psi_{j,k}; j, k \in \mathbf{Z}\}$ of $L^2(\mathbf{R})$, $\psi_{j,k}(x) = 2^{-j/2}\psi(2^{-j}x - k)$, such that, for all f in $L^2(\mathbf{R})$,

$$P_{j-1}f = P_j f + \sum_{k \in \mathbf{Z}} < f, \psi_{j,k} > \psi_{j,k} \tag{2.12}$$

where P_j is the orthogonal projection onto V_j. Please refer to [3] for details.

Now, if we call W_j the orthogonal complement of V_j in V_{j-1}, i.e.,

$$V_{j-1} = V_j \oplus W_j \tag{2.13}$$

then W_j contains the *details* necessary to go from V_j to V_{j-1}. Iterating Equation 2.13 gives

$$V_{j-1} = W_j \oplus W_{j+1} \oplus W_{j+2} \oplus \cdots \tag{2.14}$$

By virtue of axioms 2 and 3 for multiresolution analysis, this implies that

$$L^2(\mathbf{R}) = \oplus_{j \in \mathbf{Z}} W_j, \tag{2.15}$$

a decomposition of $L^2(\mathbf{R})$ into mutually orthogonal subspaces. In other words, a given resolution can be attained by a sum of added details.

If a function ϕ defines an orthonormal basis of V_0 such that it satisfies the five conditions for multiresolution analysis, it is called the *scaling function* of the wavelet basis. Thus, if $V_0 \subset V_{-1}$, then

$$\phi(x) = \sum_{k \in \mathbf{I}} c_k \, \phi(2x - k) \tag{2.16}$$

where $\phi(x)$ is the *scaling function*. Integrating both sides of Equation 2.16 with respect to (w.r.t) x, we obtain

$$\int_{-\infty}^{\infty} \phi(x)dx = \int_{-\infty}^{\infty} \sum_{k \in \mathbf{I}} c_k \, \phi(2x - k)dx \tag{2.17}$$

$$= \sum_{k \in \mathbf{I}} c_k \int_{-\infty}^{\infty} \phi(2x - k)dx \tag{2.18}$$

Assuming $\phi \in \mathcal{L}^1(\mathbf{R}) \cup \mathcal{L}^2(\mathbf{R})$ such that $\int_{-\infty}^{\infty} \phi(x)dx = 1$, and substituting $2x - k$ with x', Equation 2.18 can be written as

$$1 = \sum_{k \in \mathbf{I}} c_k \int_{\infty}^{-\infty} \phi(x')dx'/2 \tag{2.19}$$

Substituting $\int_{-\infty}^{\infty} \phi(x)dx = 1$ in 2.19 above, we get

$$\sum_k c_k = 2 \tag{2.20}$$

Also, if $\phi(x)$ satisfies the *orthogonality* condition which states that,

$$< \phi(x), \phi(x - m) >= 2\delta_{0m} \; \forall \; m \in \mathbf{I} \tag{2.21}$$

and also satisfies the *dilation equation* given in Equation 2.16, it can be shown that

$$\sum_k c_k \, c_{k-2m} = 2 \, \delta_{0m}, \forall m \in \mathbf{I} \tag{2.22}$$

where

$$\delta_{0m} = \begin{cases} 1, & m = 0 \\ 0, & \text{otherwise} \end{cases}$$

Another condition which governs the choice of the c_k's is the *approximation* condition, which is as follows: If p is a natural number, then

$$\sum_k (-1)^k k^m c_k = 0 \tag{2.23}$$

for $m = 0, 1, 2, \ldots, p-1$.

It can be shown that the number p characterize the smoothing function $\phi(x)$ such that polynomial functions of degree $p-1$ or less, are linear combinations of $\phi(x)$ and its integer translates [26]. Moreover, the higher the value of p, the greater the number of nonzero c_k's in the orthonormal basis [27].

If the scaling function ϕ is given by Equation 2.16 and the wavelet basis is defined by Equation 2.12, then it can be shown [3] that the *mother wavelet* $\psi(x)$ is given by

$$\psi(x) = \sum (-1)^n c_{1-n} \phi(2x - n) \tag{2.24}$$

and

$$\psi_{j,k}(x) = 2^{-j/2} \psi(2^{-j}x - k) \tag{2.25}$$

If P_j is the projection on V_j and Q_j is the projection on W_j, then

$$P_{j-1} f = P_j f + Q_j f \tag{2.26}$$

where

$$Q_j f = \sum_{k \in \mathbf{I}} < f, \psi_{j,k} > \psi_{j,k}(x) \tag{2.27}$$

Thus,

$$f(x) = \sum_{j,k \in \mathbf{I}} < f, \psi_{j,k} > \psi_{j,k} \tag{2.28}$$

Therefore, we can store the information in the form of wavelet coefficients $< f, \psi_{j,k} >$, for different values of j and k. These inner products are called the wavelet coefficients of the function $f(x)$. So, if we have an approximation of a signal at the resolution corresponding to V_j, then a better approximation is obtained by adding the details corresponding to W_j.

This amounts to a weighted sum of wavelets at that scale. Equation 2.12 describes this relationship where $< f, \psi_{j,k} >$ are the wavelet coefficients (or weights) of the mother wavelet $\psi(x)$ at the scale j. Thus, by iterating this idea, a square integrable signal can be seen as the successive approximation or weighted sum of wavelet basis $\{\psi_{j,k}; j,\ k \in \mathbf{Z}\}$ at finer and finer scales (with some of these .

2.2.4.3 Calculation of one dimensional wavelets coefficients

In this section, the weighting coefficients for the orthonormal D_4 wavelet basis are derived [3]. By using the approximation and orthogonality conditions on the wavelet coefficients together with the summation condition, then from Equations 2.20, 2.22 and 2.23, we get a set of equations as

$$\sum_{k=0}^{3} c_k = 2 \tag{2.29}$$

$$\sum_{k=0}^{3} c_k\, c_{k-2m} = 2\,\delta_{0m} \tag{2.30}$$

$$\sum_{k\in I} (-1)^k\, k^m\, c_k = 0 \text{ for } m = 0,\ 1 \tag{2.31}$$

Expanding the above equations, we get

$$c_0 + c_1 + c_2 + c_3 = 2 \tag{2.32}$$

$$c_0 c_2 + c_1 c_3 = 0 \tag{2.33}$$

$$-c_1 + 2c_2 - 3c_3 = 0 \tag{2.34}$$

$$c_0 - c_1 + c_2 - c_3 = 0 \tag{2.35}$$

$$c_0^2 + c_1^2 + c_2^2 + c_3^2 = 2 \tag{2.36}$$

Solving the first four equations simultaneously, we get

$$c_0 = \frac{1 + \sqrt{3}}{4}$$

$$c_1 = \frac{3 - \sqrt{3}}{4}$$

$$c_2 = \frac{3 + \sqrt{3}}{4}$$

$$c_3 = \frac{1 - \sqrt{3}}{4}$$

It can be shown that Equation 2.36 is dependent on the rest of the four equations. Squaring both sides of Equations 2.32 and 2.35 and adding, we get

$$2(c_0^2 + c_1^2 + c_2^2 + c_3^2) + 4(c_0c_2 + c_1c_3) = 4$$

By using Equation 2.33, we get $c_0^2 + c_1^2 + c_2^2 + c_3^2 = 2$ which is Equation 2.36.

Clearly increasing the value of p in Equation 2.23 increases the number of nonzero wavelet coefficients and thus smoothes the scaling function [27]. For example, if $p = 2$, then one can reconstruct exactly any factor which is a sum of linear equations. If $p = 3$, then any quadratic curve can be decomposed and reconstructed exactly using those wavelet coefficients. But, the number of nonzero coefficients increase as the value of p is increased, thus adding to the computational complexity.

2.2.4.4 Signal decomposition and reconstruction by using wavelets

The decomposition of a signal can be performed by using an algorithm introduced by Mallat [4] called the *cascade algorithm*. This algorithm decomposes the signal by taking the projections on the set of subspaces $\{W_j\}$ generated by the *scaled mother wavelet* $\{\psi_{j,n}\}$. Clearly, as we increase the value of j, we shift to coarser scales. If

$$f(x) = \sum_n a_n^0 \, \phi_{0,n}(x) \tag{2.37}$$

where $a_n^0 = < f, \phi_{0,n} >$, then $f \in V_0$.

Moving to coarser scales V_j, $j > 0$ discards some fine information contained in the wavelet components. Note that the set of subspaces $\{W_j\}$ are orthogonal to each other, i.e.,

$$W_i \cap W_j = \begin{cases} W_i, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{2.38}$$

In this way, no information of the signal is lost in the decomposition. In general, if we transform from $V_0 \rightarrow V_1$, i.e., $P_1 : V_0 \rightarrow V_1$, then

$$P_1 f = \sum_n a_n^0 \, P_1(\phi_{0,n}(x)) \tag{2.39}$$

$$= \sum_n a_n^0 \sum_l < \phi_{0,n}, \phi_{1,l} > \phi_{1,l} \tag{2.40}$$

However, from 2.16

$$< \phi_{0,n}, \phi_{1,l} > \quad = \quad \frac{1}{\sqrt{2}} \; c_{n-2l} \tag{2.41}$$

(2.42)

Therefore,

$$P_1 f \quad = \quad \frac{1}{\sqrt{2}} \; \sum_n a_n^0 \; \sum_l c_{n-2l} \; \phi_{1,l} \tag{2.43}$$

$$= \quad \frac{1}{\sqrt{2}} \; \sum_l \left(\sum_n c_{n-2l} \; a_n^0 \right) \; \phi_{1,l} \tag{2.44}$$

Moreover as P_1 is the projection of subspace V_0 onto V_1, $P_1 f \in V_1$. Hence,

$$P_1 f = \sum_l a_l^1 \phi_{1,l} \tag{2.45}$$

where $a_l^1 =< f, \; \phi_{1,l} >$, for all $l \in \mathbf{Z}$. Comparing Equations 2.44 and 2.45, we get

$$a_l^1 \; = \; \frac{1}{\sqrt{2}} \; \sum_n c_{n-2l} \; a_n^0 \tag{2.46}$$

Generalizing this to any transformation $P_j : V_{j-1} \; \rightarrow \; V_j$, we get

$$a_l^j \; = \; \frac{1}{\sqrt{2}} \; \sum_n c_{n-2l} \; a_n^{j-1} \tag{2.47}$$

where $a_l^j =< f, \; \phi_{j,l} >$. As the subspace V_j is a coarser approximation to V_{j-1}, the transformation matrix given in Equation 2.47 is called the lowpass filter matrix or the L-matrix.

In order to get the wavelet components, i.e., $Q_1 : V_0 \; \rightarrow \; W_1$, we can write

$$Q_1 f \quad = \quad \sum_n a_n^0 \; Q_1(\phi_{0,n}(x)) \tag{2.48}$$

$$= \quad \sum_n a_n^0 \; \sum_l \; < \phi_{0,n}, \psi_{1,l} > \; \psi_{1,l} \tag{2.49}$$

However, from 2.24

$$< \phi_{0,n}, \psi_{1,l} > \quad = \quad \frac{1}{\sqrt{2}} \; (-1)^n \; c_{1-n+2l} \tag{2.50}$$

Therefore,

$$Q_1\left[\phi_{0,n}(x)\right] = \frac{1}{\sqrt{2}} \sum_n a_n^0 \sum_l (-1)^n c_{1-n+2l}\, \psi_{1,l}(x) \tag{2.51}$$

$$= \sum_l \left[\frac{1}{\sqrt{2}} \sum_n (-1)^n c_{1-n+2l}\, a_n^0 \right] \psi_{1,l}(x) \tag{2.52}$$

$$= \sum_l b_l^1\, \psi_{1,l}(x) \tag{2.53}$$

where

$$b_l^1 = \frac{1}{\sqrt{2}} \sum_n (-1)^n c_{1-n+2l}\, a_n^0 \tag{2.54}$$

Generalizing, we get

$$b_l^j = \frac{1}{\sqrt{2}} \sum_n (-1)^n c_{1-n+2l}\, a_n^{j-1} \tag{2.55}$$

As the subspace W_j contains the details that were missing in V_j relative to V_{j-1}, the transformation matrix given in the above equation is called the highpass filter matrix or the H-matrix. Combining Equations 2.47 and 2.55, we get

$$a^j = \mathrm{L}a^{j-1} \tag{2.56}$$

$$b^j = \mathrm{H}a^{j-1} \tag{2.57}$$

In most practical applications, a_n^0 is *assumed* to be same as the discrete input sequence $f(n)$ where $f(n) = f(x)|_{x=n}$ for $n \in \mathcal{I}$. In other words, the input sequence is assumed to be identical to the finest scale of wavelet decomposition. Please refer to Appendix A for more details.

For signal reconstruction from the wavelet components and the signal at coarsest scale, we use

$$P_{j-1}f = P_j f + Q_j f = \sum_l a_l^j \phi_{j,l}(x) + \sum_l b_l^j \psi_{j,l}(x) \tag{2.58}$$

Also

$$P_{j-1}f = \sum_l a_l^{j-1} \phi_{j-1,l}(x) \tag{2.59}$$

where

$$a_l^{j-1} = < P_{j-1}f, \phi_{j-1,l} > \tag{2.60}$$

Substituting value of $P_{j-1}f$ from Equation 2.58, we obtain,

$$a_l^{j-1} = \sum_l a_l^j < \phi_{j,l},\ \phi_{j-1,l} > \ + \ \sum_l b_l^j < \psi_{j,l},\ \phi_{j-1,l} > \tag{2.61}$$

Using the fact that for any two functions f and $g \in \mathcal{L}^2$, $< g, f >=< g, f >^*$, Equation 2.61 can be written as

$$a^{j-1} \ = \ L_j^* \ a^j \ + \ H_j^* \ b^j \tag{2.62}$$

Decomposition using wavepackets

Another approach of decomposing the signal is based on the theory of *wavepackets* [3]. In this case, the stream of data is first decomposed into lower-order scaling coefficients and higher-order wavelet coefficients. Each of the higher-order wavelet coefficients is further decomposed into lower-order and higher-order coefficients. Thus, if the subspace V_0 is split into orthogonal subspaces V_1 and W_1 as given by

$$V_0 \ = \ V_1 \ \oplus \ W_1 \tag{2.63}$$

then,

$$W_1 \ = \ W_1^1 \ \oplus \ W_1^2 \tag{2.64}$$

In Equation 2.64, W_1^1 corresponds to the coarse details in W_1 while W_1^2 contains the high-resolution components of W_1.

2.2.4.5 Extension of wavelet basis to two dimensions

The orthonormal wavelet basis in $\mathcal{L}^2(\mathbf{R}^2)$ can be constructed by starting with a basis in $\mathcal{L}^2(\mathbf{R})$. Let $\psi_{j,k}(x) \ = \ 2^{-j/2} \ \psi(2^{-j}x \ - \ k)$ be the orthonormal basis in $\mathcal{L}^2(\mathbf{R})$. Its two-dimensional equivalent is generated by taking the tensor product

$$\psi_{\mathbf{j_1},\mathbf{k_1}\ \mathbf{j_2},\mathbf{k_2}}(x_1, x_2) \ = \ \psi_{j_1,k_1}(x_1) \ \psi_{j_2,k_2}(x_2) \tag{2.65}$$

There exists another better method in which dilation of the resulting orthonormal basis controls both the variables simultaneously. In this approach, one considers the tensor product of two one-dimensional multiresolution analysis rather than two one-dimensional wavelet basis. Thus, if

$$\mathbf{V_0} \ = \ V_0 \ \otimes \ V_0 \tag{2.66}$$

and

$$F \in \mathbf{V_j} \Leftrightarrow F(2^j x - n_1, 2^j y - n_2) \in \mathbf{V_0} \ \forall\, n_1,\ n_2 \in \mathbf{Z} \tag{2.67}$$

then $\mathbf{V_j}$ forms a multiresolution ladder in $\mathcal{L}^2(\mathbf{R}^2)$ satisfying

$$\cdots \mathbf{V_2} \subset \mathbf{V_1} \subset \mathbf{V_0} \subset \mathbf{V_{-1}} \subset \mathbf{V_{-2}} \cdots \tag{2.68}$$

$$\overline{\bigcup_{j \in \mathbf{I}} \mathbf{V_j}} = \mathcal{L}^2(\mathbf{R^2}), \tag{2.69}$$

$$\bigcap_{j \in \mathbf{I}} \mathbf{V_j} = \{0\} \tag{2.70}$$

Thus

$$\phi_{\mathbf{j};\mathbf{n_1},\mathbf{n_2}}(x, y) = \phi_{j,n_1}(x)\, \phi_{j,n_2}(y) \tag{2.71}$$

$$= 2^{-j}\, \phi(2^{-j}x - n_1)\phi(2^{-j}y - n_2), \quad n_1, n_2 \in \mathbf{I} \tag{2.72}$$

$$\mathbf{V_{j-1}} = V_{j-1} \otimes V_{j-1} \tag{2.73}$$

$$= (V_j \oplus W_j) \otimes (V_j \oplus W_j) \tag{2.74}$$

$$= V_j \otimes V_j \oplus [(W_j \otimes V_j) \oplus (V_j \otimes W_j) \oplus (W_j \otimes W_j)] \tag{2.75}$$

$$= \mathbf{V_j} \oplus \mathbf{W_j} \tag{2.76}$$

where $\mathbf{W_j}$ consists of three parts given by $\psi_{j,n_1}(x)\, \phi_{j,n_2}(y)$ for $(W_j \otimes V_j)$, $\psi_{j,n_2}(y)\, \phi_{j,n_1}(x)$ for $(V_j \otimes W_j)$, and $\psi_{j,n_1}(x)\, \psi_{j,n_2}(y)$ for $(W_j \otimes W_j)$. This leads to the three wavelets

$$\psi^{\mathbf{h}}(x, y) = \phi(x)\, \psi(y) \tag{2.77}$$

$$\psi^{\mathbf{v}}(x, y) = \phi(y)\, \psi(x) \tag{2.78}$$

$$\psi^{\mathbf{d}}(x, y) = \psi(x)\, \psi(y) \tag{2.79}$$

where h, v, and d stand for horizontal, vertical and diagonal components.

Filtering can be done on *rows* and *columns* in a two-dimensional array, corresponding to the horizontal and vertical directions in images, for example. If a^0 is an $N \times N$ array, then applying Equation 2.47 to the rows of the image results in an array $a^{1/2}$ of size $N/2 \times N$. Applying Equation 2.55 also results in an array of size $N/2 \times N$, say $b^{1/2}$. The transformation is then applied to the columns of the array $a^{1/2}$ to obtain a^1 and $b^{1,h}$ each of size $n/2 \times N/2$, respectively. The same transformation is applied to $b^{1/2}$ to obtain another two arrays, $b^{1,v}$ and $b^{1,d}$, of sizes $N/2 \times N/2$, respectively. The elements of the array a^1 are called the scaling coefficients of the image while elements of arrays $b^{1,h}$, $b^{1,v}$, and $b_{1,d}$ are called the wavelet

coefficients of the image. After obtaining the wavelet coefficients of the input image, the values of the coefficients are analyzed. All the coefficients having values less than some threshold are made equal to zero. Note that this approach is the same as was applied to the case of one-dimensional signal compression. The image is thus reconstructed from these nonzero wavelet coefficients by applying the Equation 2.62 to the rows and columns of the scaling and wavelet coefficients. Therefore,

$$a^{1/2} = L^*_{col}\, a^1 + H^*_{col}\, b^{1,h} \tag{2.80}$$

$$b^{1/2} = L^*_{col}\, b^{1,v} + H^*_{col}\, b^{1,d} \tag{2.81}$$

Finally,

$$a^0 = L^*_{col}\, a^{1/2} + H^*_{col}\, b^{1/2} \tag{2.82}$$

Clearly this approach can be extended to higher dimensions also. More interesting (non-trivial) multidimensional wavelet schemes are obtained when nonseparable subsampling is used [28]. For example, a nonseparable subsampling by 2, of a double indexed signal $x(n_1, n_2)$ is obtained by retaining only samples satisfying the equation

$$\begin{bmatrix} n_1 \\ n_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad , u_1,\ u_2 \in \mathbf{I} \tag{2.83}$$

The resulting points are located on the so called *quincunx* sublattice [28] of $\mathbf{I}^2$. The quincunx case is of interest in image processing because it treats different directions more homogeneously than the tensor product scheme discussed earlier. Instead of having two favorite directions (horizontal and vertical), the quincunx scheme treats horizontal, vertical and diagonal on the same footing without introducing any redundancies. With this brief overview of multiresolution decomposition, the concept of filter banks will be discussed with focus on multirate filter banks as they are closely related to wavelets.

2.2.5 Multirate filter banks

Multirate filter banks play an important role in the study of wavelet systems. A special class of multirate systems called quadrature mirror-image filter banks are especially significant in this regard. With these filter banks, it is not only easier to design the wavelet basis but it is also efficient in terms of implementation. The basic building blocks in a multirate digital signal processing (DSP) system are downsamplers and upsamplers.

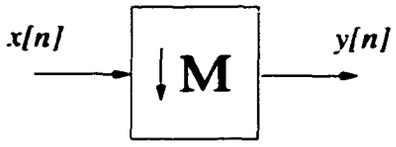


Figure 2.3. *An M-fold downsampler.*

Downsampler

An M-fold downsampler, depicted in Figure 2.3, takes an input sequence $x[n]$ and produces the output sequence

$$y[n] = x[Mn] \tag{2.84}$$

where M is an integer. The constant M is referred to as the downsampling factor. In simple terms, the downsampler discards every M samples of the input sequence. From Equation 2.84, it follows that the downsampling is a linear time-varying operation. The relationship between the input and output of the downsampler in the frequency domain is given by

$$Y(z) = \frac{1}{M} \sum_{k=0}^{M-1} X(z^{1/M} e^{-j\pi k/M}) \tag{2.85}$$

where $X(z)$ is the z-transform of the input sequence $x[n]$. Assuming that the sampling period before and after the downsampling is normalized to one, the frequency-domain relation becomes

$$Y(e^{j\omega}) = \frac{1}{M} \sum_{k=0}^{M-1} X(e^{j(\omega-2\pi k)/M}) \tag{2.86}$$

The spectrum of output of a downsampler is simply a scaled sum (i.e., average) of M shifted versions of the original input spectrum. Due to our convention of normalizing the sampling period after downsampling to one, the spectrum is also stretched. It is important to understand, however, that this spectrum stretching effect is only a consequence of the sampling period renormalization and is not caused by downsampling itself.

As is evident from Equation 2.86, downsampling can result in aliasing. That is, the downsampling operation can result in multiple baseband frequencies in the input signal being mapped to a single frequency in the output signal. If aliasing occurs, it is not possible to recover the original signal from its downsampled version. Aliasing can be avoided if $x[n]$ is a lowpass-filtered signal bandlimited to $|\omega| < \pi/M$. This condition ensures that the Nyquist

criterion is not violated. To illustrate the frequency domain effects of downsampling, we consider two examples. In the first example, the input signal is chosen so as to avoid violation of Nyquist criterion (no aliasing). The signal in the second example results in aliasing.

In the first case, suppose we apply the signal with the spectrum given in Figure 2.4(a) to the input of a two-fold downsampler. The spectrum of the downsampled signal is formed by the scaled sum of the two shifted versions of the original spectrum shown in Figure 2.4(b). Due to renormalization of the sampling period, these spectra also appear stretched. The resulting spectrum is shown in Figure 2.4(c). Because the two shifted spectra in Figure 2.4(b) do not overlap, there is no aliasing. The shape of the original spectrum is clearly discernible in the spectrum of the output signal.

In the second case, suppose we apply the signal with the spectrum given in Figure 2.5(a) to the input of a two-fold downsampler. The spectrum of the downsampled signal is formed by the scaled sum of the two shifted versions of the original spectrum in Figure 2.5(b). Again these spectra appear stretched due to renormalization of the sampling period. The resulting spectrum is shown in Figure 2.5(c). Because the two shifted spectra in Figure 2.5(b) overlap, aliasing occurs. Consequently, it is not possible to recover the original signal from its downsampled version. It is also evident that the resulting spectrum of the output signal is as shown in Figure 2.5(c). Due to aliasing, the spectrum is distorted and no longer resembles the spectrum of the original input.

Upsampler

An M-fold upsampler, depicted in Figure 2.6, takes the input sequence $x[n]$ and produces the output sequence

$$y[n] = \begin{cases} x[n/M] & \text{if } n \text{ is an integer multiple of } M \\ 0, & \text{otherwise} \end{cases} \tag{2.87}$$

where M is an integer. The constant M is referred to as the upsampling factor. The relationship between the input and output of an M-fold upsampler in the z-domain is given by

$$Y(z) = X(z^M) \tag{2.88}$$

Upsampling has a very simple and straightforward interpretation in the frequency domain. It moves the location of the sampling frequency on the frequency axis. Due to our convention of normalizing the sampling period after upsampling to one, the spectrum also seems to be

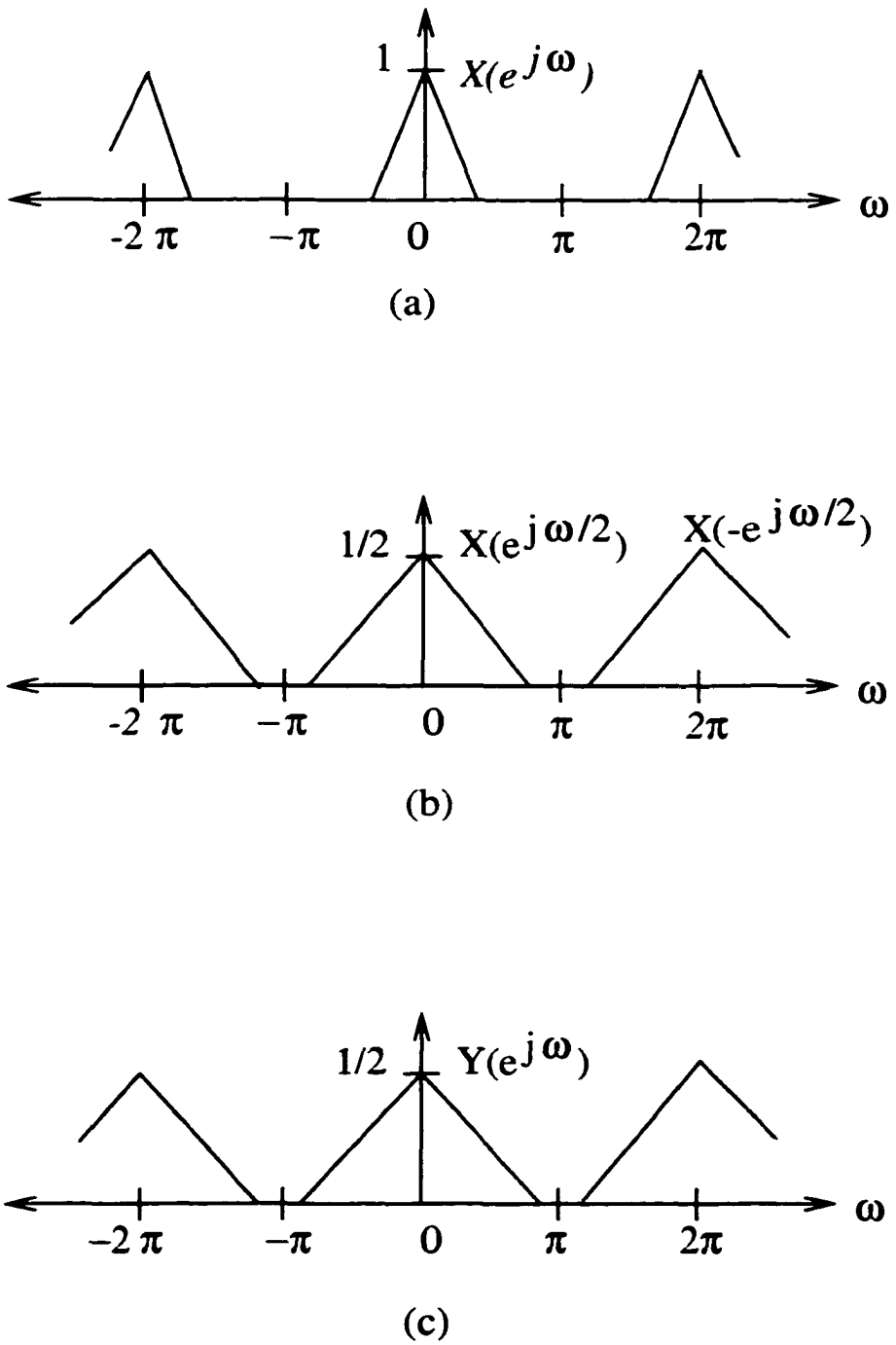


Figure 2.4. *Effects of two-fold downsampling in the frequency domain (no aliasing). (a) Spectrum of the input signal. (b) The shifted versions of the original input spectrum used to form the spectrum of the downsampled signal. (c) Spectrum of the downsampled signal.*

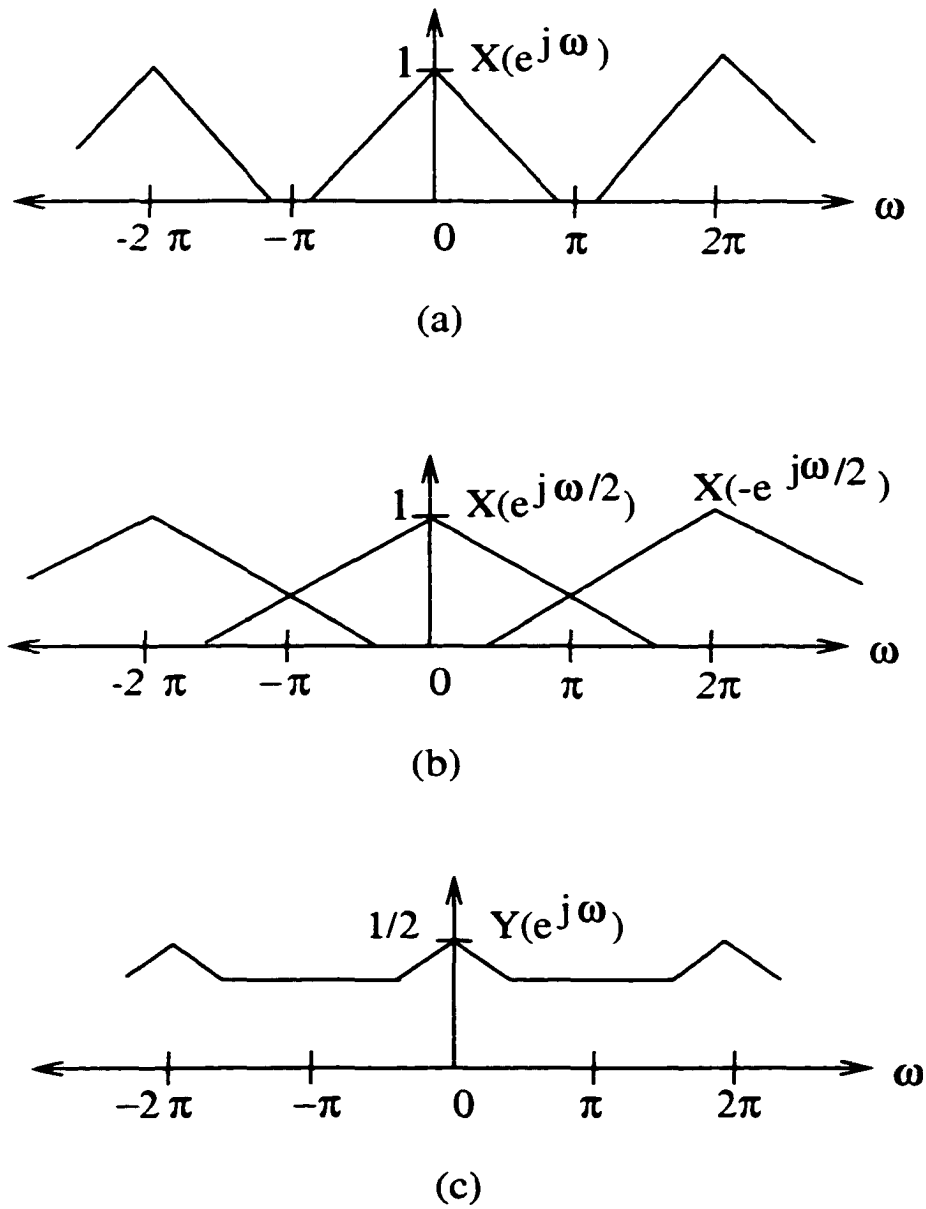


Figure 2.5. *Effects of two-fold downsampling in the frequency domain (with aliasing). (a) Spectrum of the input signal. (b) The shifted versions of the original input spectrum used to form the spectrum of the downsampled signal. (c) Spectrum of the downsampled signal.*

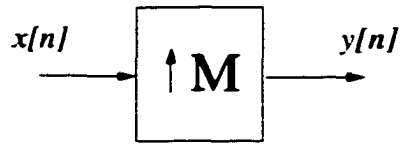


Figure 2.6. *An M-fold upsampler*

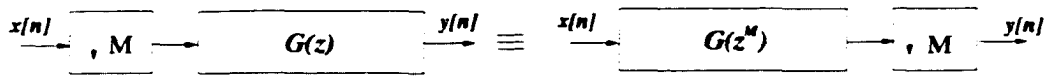


Figure 2.7. *The first noble identity.*

compressed. It is important to understand, however, that this compression effect is only a consequence of the sampling period renormalization and is not caused by the upsampling itself.

Noble identities

Often a downsampler or upsampler appears in cascade with a filter. Although it is not always possible to interchange the order of upsampling/downsampling and filtering without changing the characteristics of the system, it is possible to find an equivalent system with the order of these operations reversed, through the use of two very important relationships called the *Noble identities*. The first identity allows us to replace a filtering operation on one side of a downsampler with an equivalent filtering operation on the other side of the downsampler. This identity is illustrated in Figure 2.7 where the transfer function $G(z)$ is a rational polynomial expression. The second identity allows us to replace a filtering operation on one side of an upsampler with an equivalent filtering operation on the other side of the upsampler. This identity is shown in Figure 2.8. It is important to emphasize that these identities hold only if the transfer function $G(z)$ is a rational polynomial expression [29].

In addition to their theoretical utility, the noble identities are of great practical significance as well. For performance reasons, it is usually desirable to perform filtering operations on the side of an upsampler (or downsampler) with lower sampling rate. Using the noble

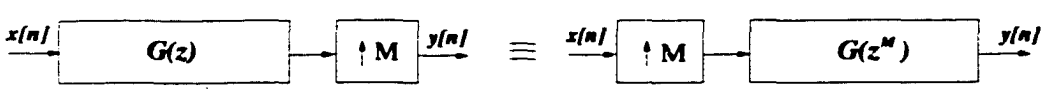


Figure 2.8. *The second noble identity.*

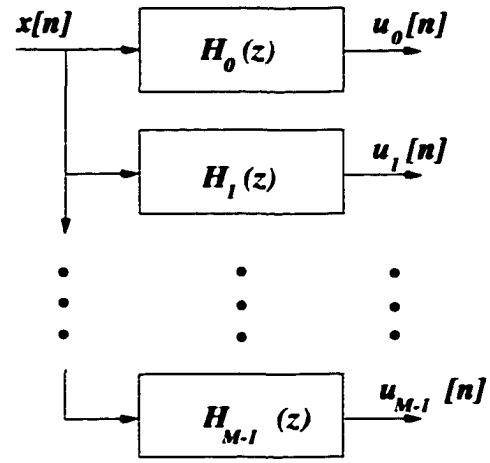


Figure 2.9. *An M-channel analysis filter bank.*

identities, we can move the filtering operations across the upsamplers (or downsamplers) and achieve improved computational efficiency.

Filter banks

A filter bank is a collection of filters having either a common input or common output. When the filters share a common input, they form what is called an *analysis filter bank*. When they share a common output, they form what is called a *synthesis filter bank*. These two types of filter banks are depicted in Figures 2.9 and 2.10. Each of these filter banks consist of M filters. The filters H_k belonging to the analysis bank are called analysis filters and the filters F_k belonging to the synthesis bank are called the synthesis filters. The signals $u_k[n]$ and $v_k[n]$ are called the subband signals. The frequency responses of the analysis/synthesis filters may be overlapping, marginally overlapping, or greatly overlapping, depending on the application.

Although M-channel filter banks have been an important topic of research, most of the past research in image compression was focussed on 2-channel filter banks. The 2-channel J-stage decomposition of a one-dimensional signal is as shown in Figure 2.11. The filters with frequency responses $H(z)$ and $G(z)$ in Figure 2.11 are the lowpass and highpass filters, respectively. The input signal $x[n]$ is decimated by 2 and passed through filters $H(z)$ and $G(z)$ to obtain scaling coefficients $s^1[n]$ and wavelet coefficients $w^1[n]$ respectively. The superscript denotes the level of decomposition. The lowpass output is passed through the same set of lowpass and highpass filters to obtain outputs $s^2[n]$ and $w^2[n]$ respectively. The

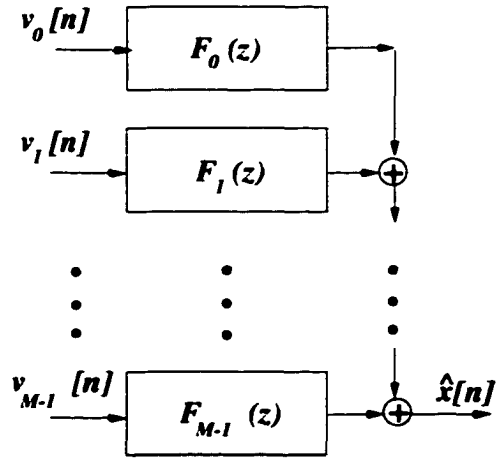


Figure 2.10. *An M-channel synthesis filter bank.*

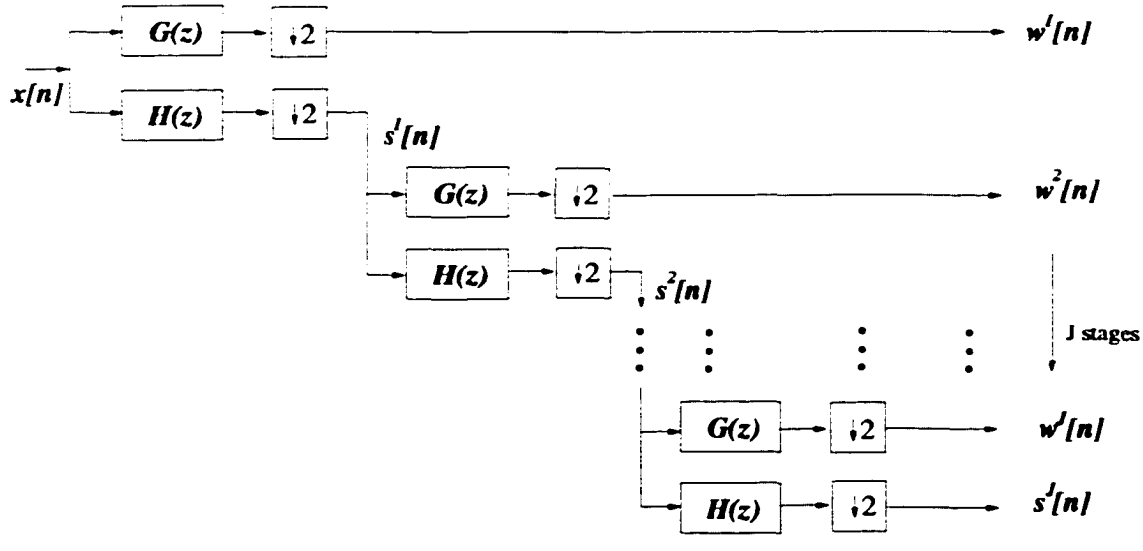


Figure 2.11. *An 2-channel J-stage wavelet decomposition.*

process is repeated J number of times to finally obtain the coarsest scale scaling coefficients $s^J[n]$ and the wavelet coefficients $w^J[n]$. These steps are reversed to obtain $x[n]$ from the scaling coefficients $s^J[n]$ and wavelet coefficients $w^J[n]$, $w^{J-1}[n]$, ..., $w^1[n]$. Because the signal is downsampled by two at every stage, the total length of the decomposed sequence (sum of lengths of output sequences from all the subbands) is equal to the length of the input sequence. The signal $s^J[n]$ is called the *approximation signal* and the signal $w^J[n]$ is called the *error signal.*

2.2.6 Correlation between multiresolution analysis and filter banks

Here we examine the correlation between multiresolution decomposition and filter banks. We consider dyadic scales for simplicity. As seen in Section 2.2.4.4, an orthogonal multiresolution decomposition of a sequence a_n^0 involves applying the filter coefficients c_k and $(-1)^k c_{1-k}$ (say d_k) to the discrete input sequence as given by Equations 2.47 and 2.55. These coefficients correspond to the analysis lowpass and highpass filters, respectively. The matrix L corresponds to the lowpass filtering convolution matrix while the H matrix corresponds to the highpass filter convolution matrix. As the rows of the L and the H matrices are shifted by two elements to the right compared to the row above it, this is equivalent to a downsampling operation by a factor of 2. The total number of scaling and wavelet coefficients after each stage of decomposition is same as that of scaling and wavelet coefficients of the input signal. Thus the dimensions of the L and H matrices are $N/2 \times N$. Analysis-synthesis filter banks that satisfy this property are called maximally decimated filters banks, and are also known as the quadrature mirror-image filter (QMF) banks [29]. The synthesis lowpass and highpass filtering operations are performed by the the convolution matrices L^* and H^*, respectively. The filter and add operation is given by Equation 2.62. The columns of the L^* and H^* matrices are also shifted by two compared to the column on their left. The frequency response of the synthesis filters is the complex conjugate of the frequency response of the analysis filters. Thus, the synthesis filters are also called the mirror image of the analysis filters. In other words, if $h[n]$ and $g[n]$ are the coefficients of the analysis filters, then $h[-n]$ and $g[-n]$ are the coefficients of synthesis filters. Perfect reconstruction filter banks are ones in which the output of synthesis filters are the same as the input filters. The Daubechies filters satisfy this property of perfect reconstruction and belong to the family of perfect reconstruction QMF filter banks (also known as the PR-QMF filter banks). Details can be found in [29].

2.2.7 Biorthogonal wavelets

Biorthogonal wavelets were introduced by Cohen et al. in [5]. In this construction, orthonormal wavelets were generalized by using two sets of functions, $\{\phi_{ij}, \psi_{ij}\}$ and $\{\tilde{\phi}_{ij}, \tilde{\psi}_{ij}\}$. The wavelet functions $\psi_{ij}, \tilde{\psi}_{ij}$ do not form orthogonal bases, but they are required to form dual bases for $\mathcal{L}_2$ such that

$$< \psi_{ij}, \tilde{\psi}_{i'j'} > = \delta_{ii'}\delta_{jj'} \tag{2.89}$$

where $\psi_{ij} = \psi(2^i - j)$. The following analysis and synthesis relationships also hold:

$$f = \sum_{ij} < f, \psi_(2^i x - j) > \tilde{\psi}(2^i x - j) \tag{2.90}$$

$$f = \sum_{ij} < f, \tilde{\psi}(2^i x - j) > \psi(2^i x - j) \tag{2.91}$$

In this case we have two dual multiresolutions, V_i, $\tilde{V}_i$, and the complement wavelet spaces W_i, $\tilde{W}_i$. It is not necessary to have orthogonality between V_i and W_i. Instead there is orthogonality between W_i, $\tilde{V}_i$ and V_i, $\tilde{W}_i$. The two multiresolutions may also coincide. Extending the filter matrices to the dual multiresolution spaces, let $\tilde{H}$ and $\tilde{G}$ correspond to the spaces $\tilde{V}_i$ and $\tilde{W}_i$, respectively (also called dual pair). For orthonormal wavelets, $\tilde{H}$=H and $\tilde{G}$=G. The advantage of adding generality is that all filters can be chosen to be finite and have other properties such as symmetry and linear phase. Generally the dual pair matrices $\tilde{H}$ and $\tilde{G}$ are used for decomposition while the matrices H and G (also called primal bases) are used for reconstruction. This is because the primal bases generally have better regularity and regularity is important for the quality of the reconstructed signal [30]. The analysis-synthesis stage for a two-channel maximally decimated biorthogonal filter bank is shown in Figure 2.12. The analysis stage consists of the filters with transfer functions $\tilde{H}(z)$ and $\tilde{G}(z)$ corresponding to the spaces $\tilde{V}_i$ and $\tilde{W}_i$ respectively. On the other hand, the synthesis stage has the filters with transfer functions $H(z)$ and $G(z)$ corresponding to the spaces V_i and W_i respectively. Clearly, if $y[n] = x[n]$, the filter banks would satisfy the condition of perfect reconstruction.

An example

The Haar wavelet is a trivial example of a biorthogonal wavelet basis. The filters are orthogonal with compact support and also satisfy the property of symmetry. More nontrivial examples of biorthogonal bases can be found in [5]. In the next section, we shall discuss various kinds of wavelets that exist in the literature. Since a vast amount of material

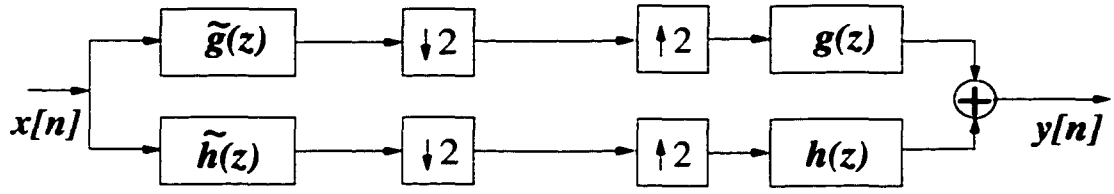


Figure 2.12. *A two-channel maximally decimated biorthogonal filter bank*

already exists on wavelets, the coverage will focus on the application of wavelets to image compression.

2.2.8 Wavelets in image coding

Wavelet transforms have been demonstrated to be viable in image compression applications. This is mainly due to the lapped nature of the transform and the computational simplicity which comes from filter-bank implementation. Lapped transforms were developed in a systematic way by Malvar in [31] to reduce the blocking artifacts introduced as a result of quantization of block transform coefficients (such as the DCT). The idea was to extend the basis functions beyond the block boundaries, creating an overlap, in order to eliminate the blocking artifact. Many articles have been presented evaluating the performance of different wavelet transforms. Wavelet transforms as applied to image compression can be classified into two main categories: separable and nonseparable wavelets [3]. Separable wavelets are the ones that are equivalent to applying a one-dimensional wavelet transform to the rows and columns of the image respectively. Nonseparable wavelets, on the other hand, are applied on a two-dimensional sublattice of the original image lattice. Due to the design of two-dimensional filters required for generating nonseparable two dimensional filters, most of the focus in image compression has been on separable wavelets. Moreover, there has not been any significant improvements to the complexity of their design. For more details on nonseparable wavelets, the reader is referred to [8]. The basis of research in this thesis is focussed on using separable wavelets for image compression.

The choice of a suitable wavelet for image compression depends on the type of image, degree of compression required and constraints on computational complexity. Before we go into recent wavelet types, it should be stressed that there have been numerous wavelet bases proposed to compress certain types of images using certain types of quantizers and encoders. Each of these blocks could significantly affect the performance of the overall compression system. Coming back to the discussion on performance of different wavelet filters, the range

of wavelets that can be utilized to evaluate the performance can be an impractically large number. For example, Villasenor *et al.* in [32] and [33], compared over 4300 candidate filter banks from the biorthogonal family and only six filter banks gave acceptable results for image compression applications with respect to compression ratio and computational complexity. Extensive analysis was also performed by Balasingham and others in [34]-[38]. After taking the computational complexity and compression ratio into consideration, the biorthogonal 9/7 filters were the overall best for most of the test images. Moreover these biorthogonal wavelets had superior performance over the equivalent orthogonal filters due to their additional property of linear phase and symmetry. This tends to smear the edges in an image equally on both sides which had better subjective performance compared to the asymmetric spreading in an edge in the case of orthogonal wavelets. Moreover, longer length filters tend to produce *ringing* around the edges in the image which has a degrading effect on the quality of the image.

In the next subsection, the effect of finite length signals is discussed. This is quite important because all of the real images are of finite length and width.

Finite length signals

The output of filters depends on past and/or future pixels in the image. Once we get close to the edge of the image, the filter inputs are not defined. There are two approaches to this problem: One is to assume the pixels beyond the boundary in the image. Second is to adapt the filters at the boundary (change their length) so that we do not need any extra pixels beyond the boundary. This is also called boundary filtering.

The first approach can use two kinds of extensions at the boundary, periodic and symmetric. In a periodic extension, the input image is tiled around itself to get an infinite tiling of the input image. This property has the undesirable disadvantage of having sudden sharp discontinuities (edges) at the boundaries of the image. In a symmetric approach, the input image is mirrored about its boundaries. The symmetric approach is more desirable because it does not introduce any discontinuity at the boundaries of the image. Due to this advantage, we will be using symmetric extensions throughout the thesis. More detailed coverage on finite signals and wavelets can be found in [39], [40] and [41].

2.2.8.1 Reversible wavelets

The most popular approach that emerged in the last couple of years is the concept of reversible wavelets. The term reversible is actually a misnomer. Ideally all wavelet trans-

forms are linear transforms and are reversible because they satisfy the property of perfect reconstruction. However, this property is lost if integer arithmetic is used for computation. Sometimes, however, it is possible to introduce finite-precision arithmetic without destroying reversibility. Such wavelets transforms are called reversible transforms. Reversible transforms are ideally suited for applications where less information is desirable when the transforms are computed using finite arithmetic. Such a requirement is often imposed in lossless and unified lossless/lossy compression. The idea behind creating a reversible transform is simple: through the clever use of quantization (e.g., rounding), modify the original transform so that it can be computed using finite-precision arithmetic while preserving reversibility. The quantization step results in the transform becoming nonlinear and is thus impossible to comply to any linear signal analysis. The degree to which the reversible transform approximates its parent transform is extremely important. If, however, the reversible transform fails to mimic the properties of the underlying linear transform, poor results will be obtained with the reversible transform. In the past, reversible transforms have been developed largely by ad hoc methods. Consequently, they were difficult to design and thus only a few reversible transforms were known. In spite of all this, a few good transforms were developed. The S-transform [42] was the first reversible transform ever developed. That was followed by the TS transform [10] and the S+P transform [11] in chronological order. Recently, a new philosophy for the design and implementation of invertible transforms, called *lifting*, was first proposed by Sweldens [43]. In the most abstract sense, lifting provides a means to generate invertible mappings between sequences of numbers. With lifting, the forward transform maps a single sequence into two or more new sequences. The inverse transform then maps the new sequences back to the original sequence. Suppose we have a sequence that we wish to transform. With lifting, the forward transform is calculated in three stages:

1. Split: the input sequence is decomposed into two or more new sequences.
2. Predict: the numbers from one sequence are used to predict the values of other sequences.
3. Scale: the final normalization is applied to each output sequence.

Figure 2.13 illustrates these three stages in a lifting scheme. The input $x[n]$ is split into even and odd samples (equivalent to decimation by 2) which is followed by prediction and update stages. Iteration of the lifting stage on output c[n] creates the complete set of wavelet transform's scaling and wavelet coefficients $c^j[n]$ and $d^j[n]$, where j is the scale of decomposition. The scaling step is used to approximate the lifting transform to the

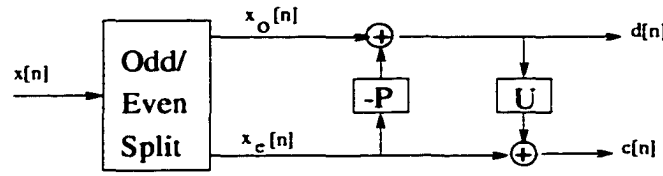


Figure 2.13. *Lifting stage: Split, predict, update.*

equivalent linear wavelet transform. The same steps can be traced backwards to obtain the input sequence from the output sequences. The steps involved in that scenario are

1. Scale
2. Predict
3. Join

As a matter of terminology, a transform calculated using this framework is called a lifted transform. The lifting steps are easily inverted *even if P and U are nonlinear or space-varying* [44]. It should be noted that for a lifting transform to yield integer wavelet and scaling coefficients, the scaling factor should be ± 1 [45]. In fact, *all* wavelet transforms can be factored into a series of lifting stages (with perhaps multiple predicts and updates per stage) [45]. Several wavelet-based integer-coefficient filter banks can be found in [46]. Calderbank et al. [47] have shown that every wavelet transform (or some subclass of filter banks) can be used to map integer input samples to integer wavelet coefficients using lifting steps. Balasingham et al. [48] compared the compression performance between integer coefficient filter banks (ICFBs) of different filter lengths. They concluded that generally ICFBs with the longest filters on the average performed the best. In particular the 17/11 ICFB had the lowest bit rate among the 8 ICFBs they had considered. They concluded that the simple 5/3 ICFBs seemed to be good candidates for practical implementation as they performed well and had low computational complexity. Unfortunately they could not compare the results with the TS transform due to a drawback in their implementation. A summarized overview of a performance comparison between existing reversible wavelet transforms can be found in [49]. Results indicated that filters of longer length caused ringing artifacts near the edges. So, even though filters of longer length had better SNRs results compared to filters of short length (such as the 5/3 filter), the latter performed better in subjective tests [49].

Recent research is focussed on categorizing wavelets based on image types. One approach is to choose certain wavelets for certain types of images as is done in [50]. The other

approach is to use adaptive wavelet transforms. Claypoole et al. [44], [51] have used the lifting scheme to design adaptive reversible wavelet transforms. Transforms were adapted based on scale or space. Generally the filter length was decreased as the predictor in the lifting scheme approached an edge in the image. This was done because longer-length filters tend to perform better in smooth regions in the image while short-length filters (e.g. Haar) performed better in an image with many edges and sudden intensity variations. There is still a need to improve the computational efficiency of these algorithms, which is an active area of current research.

2.3 Quantization stage

The process of mapping a real line into a countable discrete alphabet is called quantization. In its simplest form, each sample is individually quantized which is called as *scalar quantization*. A more powerful method consists in quantizing several samples at once, which is referred to as *vector quantization*. It should be mentioned that the quantization stage is absent in the case of lossless compression. A detailed discussion on quantization can be found in [52] and [53].

2.3.1 Scalar quantization

In scalar quantization the input data range is divided in intervals called *bins*. The output value is typically chosen to lie within each bin which contains the input value being quantized. Thus all input values lying in each bin will have the same value. In other words, one loses the resolution of the input signal to the width of one bin. In the case of a uniform quantizer, the whole data range is divided into bins of equal width. The quantized value is chosen to lie in the middle of each bin. Although uniform quantizers are suitable for most of the applications, some applications where the probability distribution is not uniform, non-uniform quantizers are also used. In non-uniform quantizers, the bins positions and widths are chosen based on an estimate of the probability distribution of the input data. Data values with higher degree of occurrence have narrower bins compared to values with lower degree of occurrence. The difference between the input signal and the quantized signal is called quantization noise. The objective of any quantization approach is to minimize the overall quantization noise. While scalar quantizers are widely used due to their easy design and computational simplicity, vector quantizers have gained wide importance recently due to faster hardware and cheap memories thus leaving vector quantization with many advantages and a few disadvantages.

2.3.2 Vector quantization

In simplistic terms, vector quantizer can be described as a generalization of a scalar quantizer in two dimensions. There are several advantages of vector quantizers over scalar quantizers. We shall restrict ourselves to two dimensions although these advantages can be extended to multiple dimensions as well.

1. Packing Gain: There is always a gain in quantizing two variables even if they are independent. The reason is that there exist better partitions of the space than the rectangular partition obtained when separately scalar quantizing each variable. The packing gain increases with dimensionality.
2. Removal of linear and nonlinear dependencies: While linear dependencies could be removed during a linear transformation, VQ also removes nonlinear dependencies. Details can be found in [8] and [54].
3. Fractional bit rate: At low bit rates, choosing between 1.0 bits/sample or 2.0 bits/sample is a rather crude choice. By quantizing several samples together and allocating an integer number of bits to the groups, fractional bit rates can be obtained.

The main drawback of VQ is its complexity, which limits the size of vectors that can be used. One solution is to structure the codebook so as to simplify the search for the best matching vector, given the input. This is achieved with tree-structured VQ. Another approach is to apply VQ to transform coefficients obtained using linear transforms (such as wavelet transforms).

Attempts have been made in combining VQ with the wavelet transform, e.g., in [55], [56].One approach is to apply tree structured VQ wavelet coefficients. Since decoded data can be progressively refined as one proceeds through the tree structure, it provides an embedded coding scheme [55]. But no importance measure to the vector symbol was proposed. Instead, each symbol was treated equally. Averbuch et al. [14] proposed a VQ-based compression technique where different codebooks were generated for different subbands. Improvements were made by quantizing the quantization error as well [14]. The codebook length was dependent on the subband and its variance. This resulted in having higher quantization for finer scales and lower quantization for coarser scales of decomposition. Recently, variable dimension VQ has also been proposed for quantization of wavelet coefficients [57]. After a dyadic wavelet decomposition of an image, the low-low subband is further divided into L *superblocks* of size $M \times N$. Each of the superblock images can be further divided in K ways into blocks of size $H_i \times V_i$, $i = 0, 1, \ldots, K-1$. A decision mechanism is used to decide how the coefficients will be grouped among the K available ways for subsequent coding.

Details can be found in [57]. VQ has also been applied to color images. This is based on perceptual VQ. Details can be found in [58] and [59].

In the next section, we shall discuss the final stage in a compression system: coding. It is only in this stage, that the coefficients are coded into bits using a coding technique, most popular of which is entropy coding.

2.4 Coding stage

The quantized transform coefficients are finally encoded into bits to obtain the compressed bit stream. Entropy coders are widely used for this purpose. Entropy coders attempt at reducing the overall entropy of the image by using fewer bits to encode more frequently occurring symbols and more bits to encode less frequently occurring symbols. Similar to the transform stage, it is reversible and thus there is no approximation problem as in quantization. After quantization, the variables take values drawn from a finite set $\{a_i\}$. The idea is to find a reversible mapping M to a new set $\{b_i\}$ such that the average number of bits/symbol is minimized. The parameters in searching for the mapping M are the probabilities of occurrence of the symbols a_i, $p(a_i)$. If the quantized variables are stationary, these probabilities are fixed, and a fixed mapping such as static-Huffman coding can be used. If the probabilities are variable, more sophisticated techniques such as adaptive Huffman or arithmetic coding can be used. Such mappings will transform fixed-length codewords into variable-length codewords, creating a variable length bit stream. If a constant bit-rate channel is used, buffering has to smooth out variations so as to accommodate the fixed-rate channel.

2.4.1 Static coding techniques

In static coding techniques, the probabilities of occurrence of each of the symbols are fixed. Each symbol is encoded based on its probability of occurrence. The decoder receives the probabilities of each symbol so that it can also duplicate the codeword generation in order to start decoding the compressed bit stream. The codewords are generated to minimize the overall entropy of the compressed stream. The most common static coding methods are Huffman coding and arithmetic coding. The main advantage of arithmetic coding over Huffman coding is its ability to compress to fractional bits/pixel. The lowest compression achieved by scalar quantization and Huffman coding is 1 bit/sample. See [52] and [60] for details.

2.4.2 Adaptive techniques

While the above approaches come close to the entropy of a stationary known source, they fail if the source is not well known or changes significantly over time. A possible solution is to estimate the probabilities on the fly (by counting occurrences of symbols at the encoder and decoder) and modify the Huffman code accordingly. This seems a little complicated at first sight, but minor modifications are required to make a static coding technique adaptive. Moreover to reduce computations due to frequent updates, updating of probabilities is done after changes cross certain threshold. Details can be found in [52] and [61]. The adaptive version of arithmetic coding is known as Q-coding [62]. Finally Ziv-Lempel coding [63] is an elegant lossless technique which uses no a priori probabilities. It builds up a dictionary of the received strings in a way that the decoder can build the same dictionary. The dictionary size is fixed and the encoder sends only the index of the matched string (in the dictionary) to the decoder. Thus Ziv-Lempel coding maps variable-size input sequences into fixed-size codewords, which is a dual of Huffman coding. The main drawback of such dictionary-based methods is when very long sequences need to be coded. No new entities can be created once the dictionary is full and the remainder of the sequence has to be coded with the current entries. Modifications to the Ziv-Lempel algorithm allow dictionary updates. Many variants of the basic technique can be found in [60]. For certain types of data that contain runs of zeros in their quantized coefficients, *run-length coding* is used. It encodes the length of the stretch of zeros, then encodes the values of the nonzero samples and then an indicator of the start of another length of zeros. This particular coding technique is used in DCT-based transform coders such as the well-known JPEG baseline coding [18].

2.4.3 Embedded coders

Recently, embedded coders have been used for encoding wavelet coefficients. In embedded coders, coefficients are coded in decreasing order of importance. The coefficients are transmitted in the order of importance. The decoder can truncate the embedded bit stream at any point and obtain the reconstructed image. It can be shown that for an embedded stream obtained using a wavelet transform, truncating the stream at any point would yield a reconstructed image that would have the best quality (in terms of MSE) for that particular bit rate.

The most popular embedded coders for wavelet coefficients are called *embedded zero-tree wavelet (EZW) coders.* The first zero-tree coder called layered zero coding(LZC) was first proposed by Taubman et al. in [64]. Shapiro successfully developed a good practical EZW

coder [9], which was later improved by Said and Pearlman in their work of set partitioning in hierarchical trees (SPIHT) [11]. The SPIHT scheme was recently extended from 2-D to 3-D coding as well [65]. The general framework of an EZW coder is based on the fact that if the wavelet coefficients in coarser scales are insignificant, then the corresponding coefficients at higher scales will also be insignificant. If this occurs, the coarser scale coefficient and all its descendants is coded as a *zero tree*. By predicting the absence of significant information across scales, the EZW scheme can avoid encoding individual coefficient positions explicitly. This saves bits and leads to excellent compression. It was shown in [66] that compression performance in an embedded coder is directly proportional to the number of zero trees in the image. The EZW scheme can also be applied to lossless coders by replacing the wavelet transforms with reversible wavelet transforms. The bits obtained by using the zero-tree algorithm are finally compressed using adaptive coding techniques such as Huffman coding or arithmetic coding [9]. Although the embedded coders have superior performance than other coders, their main drawback is sensitivity to bit errors. Even a single bit error in the stream can affect the performance of the whole image because of the embedded nature of the algorithm. Embedded techniques have also been applied to vector quantized wavelet coefficients. Silva et al. [67] proposed a successive approximation vector quantization for wavelet coding where each vector is coded progressively by reducing the error vector from the last approximation. The developed algorithm can encode vectors in terms of their importance and more refinement can be done upon significant vectors. It performs better than EZW. It is competitive with the LZC but has computational limitations when codebook size increases. The fundamental problems associated with the combination of embedded wavelet coding and VQ can be summarized as follows:

- Proper definition of vector significance: In embedded coders using scalar quantization, the significance is based on whether the value of the pixel is above or below a certain threshold [9]. This is not the case when embedded coding is applied on vectors of wavelet coefficients.
- Development of successive vector refinement procedure: When embedded coding is applied on a per pixel basis, the pixel is refined in the so called *subordinate pass* [9]. A similar refinement procedure is required for refining a vector of wavelet coefficients.
- Complexity of VQ: Vector quantization is computationally intensive compared to scalar quantization and hence its complexity affects the real-time performance of a VQ based embedded coder.

Details on different approaches to obtain embedded bit streams using VQ can be found

in [15], [55] and [56]. Recently a VQ-based SPIHT coder, called vector SPIHT (VSPIHT) has been proposed by Mukherjee et al. in [57]. The VSPIHT encoder demonstrates better compression ratios than the scalar SPIHT coder. In the next chapter, we shall propose a wavelet-based reversible transform and apply it to compression of medical images.

2.5 Conclusions

This chapter has given a brief overview of various lossless and lossy compression techniques with more focus on wavelet based techniques. Embedded zero-tree wavelet coding techniques have proved to give overall better performance than any other techniques. Recently, reversible wavelet transforms have also been developed to transform integer input samples to integer wavelet coefficients. This property was used to obtain lossless compression using wavelets. Although longer-length wavelet filters tend to provide better compression than the short-length filters for smooth images (less edges), short wavelets overall perform the best for a wide class of images. Short-length filters perform better on images with sharp edges such as computer-generated images. Vector quantization improves the compression over scalar quantization but tends to increase the complexity of the algorithm. Many techniques to reduce the complexity of VQ, such as tree-structured VQ have been reported in literature. Complexity is usually reduced at the cost of compression performance.

Chapter 3

Lossless Data Compression Using an Integer-Arithmetic Wavelet Transform

3.1 Introduction

Lossless compression is used in applications that do not allow loss of information in the compression-decompression process. Medical image archiving is one such application. Various types of transforms that provide lossless compression are available in the literature [68]. Recently, the concept of 'reversible' wavelets was introduced which yields lossless compression of data [10]. In this chapter, a lossless compression technique based on the idea of reversible wavelets is presented. A new integer-arithmetic transformation is developed that leads to more efficient lossless compression than previous techniques. The compression technique proposed consists essentially of two parts, the integer-arithmetic transformation and the coding of the wavelet coefficients. The latter is carried out using a so-called adaptive order 1 arithmetic coder [60]. This compression technique was tested on various gray-scale and color images. A new reversible integer-arithmetic color image transformation is also proposed to reduce correlation among the three color bands of the image for superior performance.

3.1.1 Preliminaries

The discrete wavelet transform of an n-input signal decomposes the signal into a set of orthogonal frequency subbands. At each stage, the input signal is split into a 'coarse' component and a 'detail' component. The coarse component is obtained by passing the signal through a lowpass filter and the detail component is obtained by passing the signal

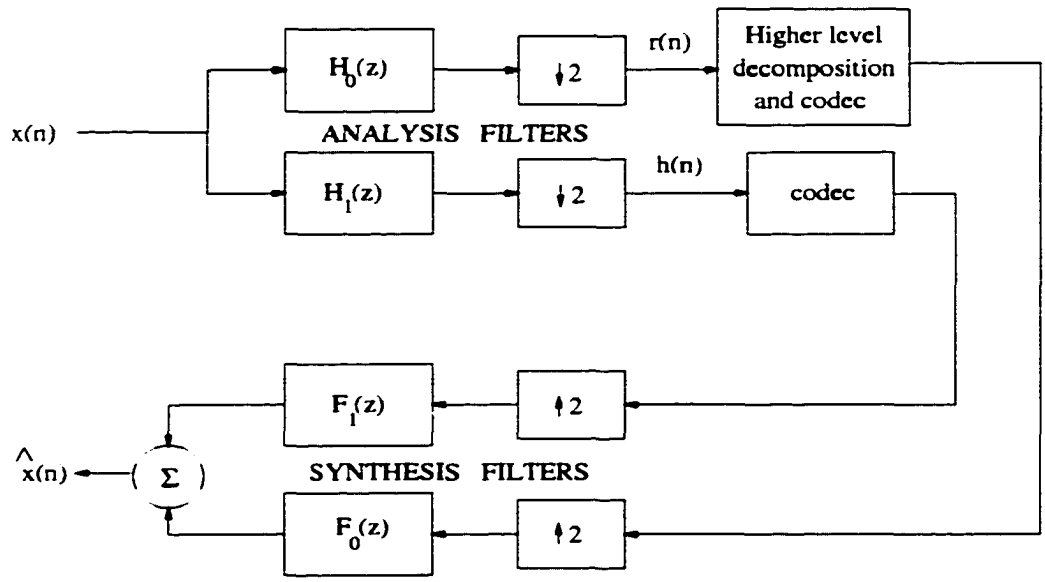


Figure 3.1. *Block diagram of analysis and synthesis wavelet filters.*

through a highpass filter. Both the lowpass and highpass filters are orthogonal and hence there is no information overlap between their outputs. To avoid any overlap, the output of both filters is decimated by a factor of 2. If $H_0(z)$ ($H_1(z)$) is the transfer function of the lowpass (highpass) filter used in the decomposition stage and $F_0(z)$ ($F_1(z)$) is the corresponding filter transfer function in the reconstruction stage, then the block diagram of the whole system assumes the form shown in Figure 3.1. For perfect reconstruction, it is required that $x[n]$ and $\hat{x}[n]$ be identical to within a scaling factor and a constant delay [69]. Wavelet filters which satisfy the property of orthogonality and alias-free reconstruction are of interest because they lead to the perfect reconstruction of $x[n]$. In the following examples, three such linear filters are given. The nonlinear transform approach to be used is based on these filters.

Examples:

a) **Haar filters:** These are the simplest of all types of filters with orthogonality of order zero. The transfer functions of filters H_0 and H_1 are given by

$$H_0(z) = \frac{1}{\sqrt{2}}(1 + z^{-1}) \tag{3.1}$$

$$H_1(z) = \frac{1}{\sqrt{2}}(1 - z^{-1}) \tag{3.2}$$

b) **TS filters:** The transfer functions of the lowpass and highpass filters of lengths two and

six, respectively, of the so called two-six (TS) transform are given by

$$H_0(z) = \frac{1}{\sqrt{2}}(1 + z^{-1}) \tag{3.3}$$

$$H_1(z) = \frac{1}{8\sqrt{2}}(-z^2 - z + 8 - 8z^{-1} + z^{-2} + z^{-3}) \tag{3.4}$$

c) **TT filters:** The transfer functions of the lowpass and highpass filters of the two-ten (TT) transform are given by

$$H_0(z) = \frac{1}{\sqrt{2}}(1 + z^{-1}) \tag{3.5}$$

$$H_1(z) = \frac{1}{128\sqrt{2}}(3z^4 + 3z^3 - 22z^2 - 22z^1 + 128 - 128z^{-1} + 22z^{-2} + 22z^{-3} - 3z^{-4} - 3z^{-5}) \tag{3.6}$$

In order to obtain lossless compression using a wavelet transform, wavelet coefficients with integer values need to be used to avoid quantization errors. Thus exact replication of the input data can be achieved. A possible approach to obtain integer-arithmetic wavelet transformations was proposed in [10] and more recently in [70], using the floor functions. In this paper, the floor function $\lfloor x \rfloor$ will be used to denote the integer part of x. For the Haar basis given in Equations 3.1 and 3.2, the corresponding integer-arithmetic nonlinear wavelet transform is given by

$$r(i) = \left\lfloor \frac{x(2i) + x(2i+1)}{2} \right\rfloor \tag{3.7}$$

$$h(i) = x(2i) - x(2i+1) \tag{3.8}$$

for $i = 0, 1, \ldots, N/2-1$ and $x(i)$ is the ith sample of the input data. Also $r(i)$ and $h(i)$ are the lowpass and highpass coefficients of the input image, respectively. When the Haar filter is applied to a finite-length vector $\mathbf{x}$, no additional boundary conditions are required to compute all the wavelet coefficients. This is due to the fact that only two elements of vector $\mathbf{x}$ are needed to calculate one element of each of vectors $\mathbf{r}$ and $\mathbf{h}$. The inverse transformation is given by

$$x(2i) = r(i) + \left\lfloor \frac{h(i)+1}{2} \right\rfloor \tag{3.9}$$

$$x(2i+1) = r(i) - \left\lfloor \frac{h(i)}{2} \right\rfloor \tag{3.10}$$

for $i = 0, 1, \ldots, N/2-1$.

The corresponding transformation for the TS filters is given in [10].

3.2 Proposed wavelet transform using integer-arithmetic

In the previous section, various linear exact reconstruction filters were presented and it was shown how the Haar filters can be used to obtain an integer-arithmetic transformation. In this section, the integer-arithmetic wavelet transform obtained from the TT filter basis is given. This leads to the transformation

$$r(i) = \left\lfloor \frac{x(2i) + x(2i+1)}{2} \right\rfloor \tag{3.11}$$

$$h(i) = c(i-2) + x(2i) - x(2i+1) \tag{3.12}$$

where

$$c(i) = \left\lfloor \frac{3r(i) - 22r(i+1) + 22r(i+3) - 3r(i+4)}{64} \right\rfloor$$

for $i = 0, 1, 2, \ldots, N/2-1$. We assume here that the transform coefficients, $r(i)$, and the input values, $x(i)$, are in terms of an L-bit fixed-point representation, where L is the wordlength. In other words, if each pixel of the input image is represented in terms of 16 bits, the lowpass-filtered image is also represented by 16 bits. On the other hand, $h(i)$ is represented in terms of a larger wordlength than $x(i)$ but its histogram is more localized and hence results in a lower entropy and higher compression. The above approach was extended to two dimensions using separable wavelet filters where Equations 3.11 and 3.12 are successively applied to the rows and columns of the image using the well-known pyramid algorithm [4].

Attention has to be given to the boundaries of the image. It can be shown that for an image of size $N \times N$ where three levels of pyramid decomposition are being used, it is required to start with at least an $(N+32) \times (N+32)$ image. To show this consider that in order to calculate every $h(i)$ in Equation 3.12, we require $r(i-2)$, $r(i-1)$, $r(i+1)$, and $r(i+2)$. This implies that four extra samples, $x(-4)$, $x(-3)$, $x(-2)$ and $x(-1)$, are needed to calculate $h(0)$. For a two-stage decomposition, we need eight extra samples and thus, sixteen samples are required for a three-stage decomposition. The same argument also applies to the last element $h(N/2-1)$. Thus, in total 32 extra samples (16 on either side) are needed for a three-stage decomposition process. In this thesis, the pixel values $x(i,j)$ with $i, j < 0$ or $i, j > N$ are obtained by flipping the image along its horizontal and vertical boundaries (also known as symmetric extension of the image). Note that the additional computations required due to the extensions of the image at boundaries, involve

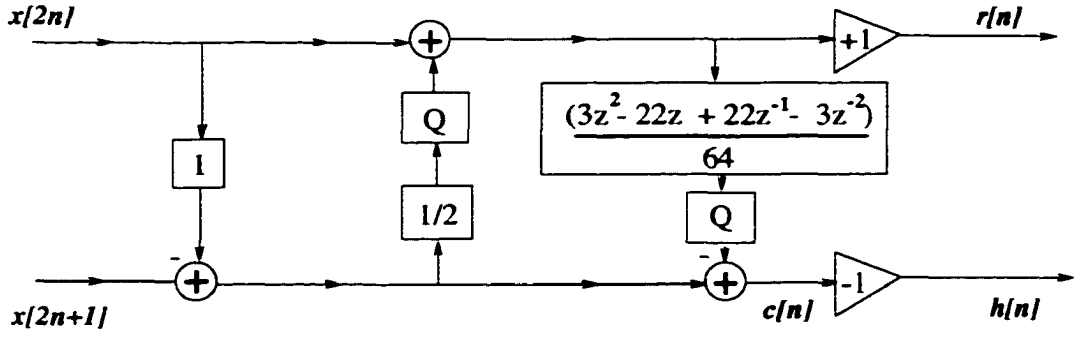


Figure 3.2. *Lifting stages for the TT transform (Q denotes quantization).*

computing extra samples for $r(i)$, only. The computation of $r(i)$ requires only integer shift-add operations. Hence, the additional computations due to symmetric extension are negligible compared to the overall computations involved.

The inverse transformation is given by

$$x(2i) = r(i) + \left\lfloor \frac{h(i) - c(i-2) + 1}{2} \right\rfloor \tag{3.13}$$

$$x(2i+1) = r(i) - \left\lfloor \frac{h(i) - c(i-2)}{2} \right\rfloor \tag{3.14}$$

where

$$c(i-2) = \left\lfloor \frac{3r(i-2) - 22r(i-1) + 22r(i+1) - 3r(i+2)}{64} \right\rfloor$$

for $i = 0, 1, 2, \ldots, N/2 - 1$. As in the forward transformation, Equations 3.13 and 3.14 are successively applied to both dimensions to reconstruct the original image.

3.2.1 Reversible two-ten transform in lifting scheme

The lifting scheme given in Figure 2.13 provides another means of designing and implementing reversible wavelets. The operations given by Equations 3.11 and 3.12 can be put in the lifting framework as given in Figure 3.2. As can be seen, we have two prediction stages and one update stage. The scaling factors are ± 1 thus making the transform reversible. The above steps can be reversed to obtain the original signal as shown in Figure 3.3

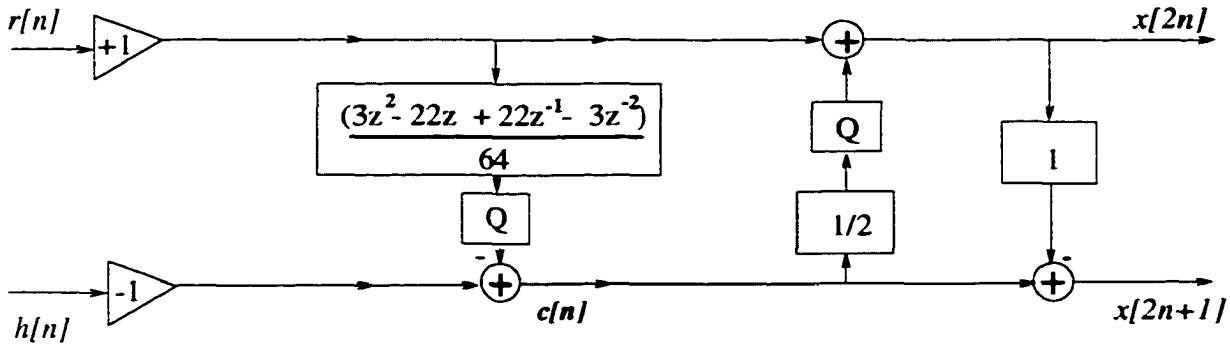


Figure 3.3. *Obtaining the original signal from the TT wavelet coefficients using lifting (Q denotes quantization).*

3.2.2 Coding of wavelet coefficients

After obtaining the wavelet coefficients, there is a need to efficiently code them. This involves a proper scanning of the coefficients followed by a compression scheme such as Huffman or arithmetic coding. The coefficients are scanned proceeding from the coarsest to the finest scale in a commonly used pattern as given in [9] which is called here *scale-based scanning*. After scanning the wavelet coefficients, the wavelet coefficient matrix is encoded using the adaptive arithmetic-coding algorithm in [60] to get the final compressed image. This approach leads to good results for the following two reasons:

1. If scale-based scanning is used, the model used in arithmetic coding can predict pixel values efficiently because each subband has elements of the same order of magnitude (except at very sharp edges). Thus, for instance, the finest scale high-high subband will mostly comprise zeroes and the arithmetic coder gives the maximum compression.
2. The scale-based scanning helps in an efficient lossy reconstruction in the mean-square error sense [9].

3.2.3 Algorithm for the coder

Efficient compression is achieved by applying the following algorithm to the input image:

Step 1 Apply Equations 3.11 and 3.12 to the rows and columns of the warped image, respectively.

Step 2 Repeat Step 1 for as many stages of decomposition as required.

Step 3 Scan the wavelet coefficients proceeding from the coarsest to the finest scale.

Step 4 Code the wavelet coefficients using arithmetic coding.

3.2.4 Algorithm for the decoder

The image can be reconstructed by applying the following algorithm:

Step 1 Decode the wavelet coefficients using arithmetic decoding of the compressed file.

Step 2 Reconstruct the original matrix from the stored scanned matrix.

Step 3 Apply Equations 3.13 and 3.14 to the columns and rows of the four coarsest scaled images, respectively.

Step 4 Repeat Step 3 for all stages of decomposition.

3.3 Compression of medical images

3.3.1 Existing methods

Various other lossless compression techniques have been reported in the literature. The standard one is the lossless Joint Photographic Expert Group (JPEG) scheme using predictive coding [71]. Recently, improved linear predictors based on the JPEG scheme have also been designed for the lossless compression of medical images [72].

3.3.1.1 Lossless JPEG predictive coding

The JPEG standard uses a predictive coding technique for compression based on lossless global prediction. The predictor uses up to three pixels for obtaining an estimate of the current pixel value. The one used in JPEG4 is being used here for the sake of comparison. If $\hat{X}$ is the predicted value of the current pixel X, then the prediction is based on

$$\hat{X} = A + D - C \tag{3.15}$$

where A, C, and D are shown in Figure 3.4. The prediction error, which is the error between the actual and predicted pixel values, is given by

$$E(i,j) = X(i,j) - \lfloor \hat{X}(i,j) \rfloor \tag{3.16}$$

The first row and first column of the original image are decorrelated by taking the difference between adjacent pixels. The prediction error and the decorrelated first row and column are coded using arithmetic coding.

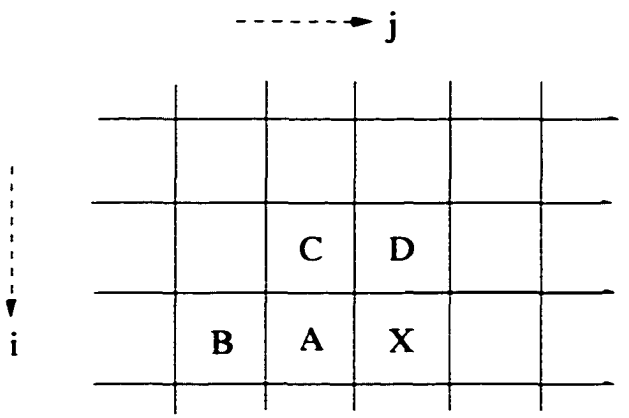


Figure 3.4. *The sample prediction neighborhood.*

3.3.1.2 Improved predictive coding

Recently in [72] new linear predictors have been designed for medical image compression. A modified estimator using up to five pixels was proposed for prediction and a number of predictors based on the proposed differentiation rules were presented. A typical predictor, $D3$ from [72], is given by

$$\hat{X} = 0.75B - 0.5C + 0.75D \tag{3.17}$$

where B, C, and D are given in Figure 3.4. As in the previous case, the prediction error and the decorrelated first row and first column are coded using arithmetic coding.

3.3.1.3 Transform-based methods

The performance of the proposed technique is also compared with other transform-based methods such as two-dimensional (2-D) lossless DCT [69], two-dimensional wavelet transform (Daubechies D_6) [73], hybrid transform [74], and the well-known GZIP (based on the Ziv-Lempel algorithm) [63].

The two-dimensional lossless DCT method involves two-dimensional block DCT along with quantization. The block diagram is given in Figure 3.5. In the two-dimensional lossless DCT method, the image is divided into blocks of 8 by 8 pixels and the two-dimensional DCT is applied. The DCT coefficients are then quantized and coded along with the quantization error using arithmetic coding. Lossless compression using the Daubechies D_6 [73] wavelet

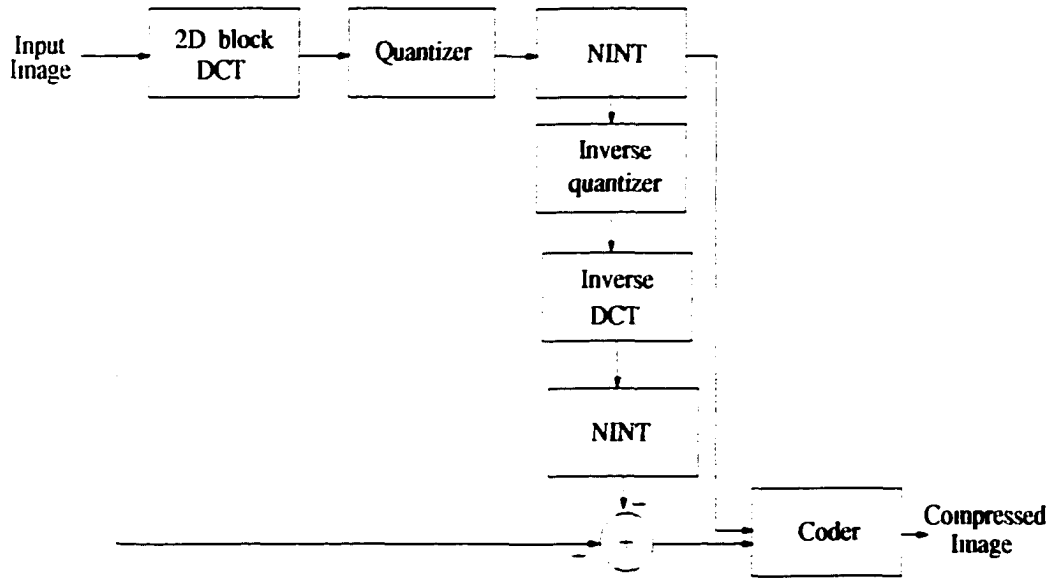


Figure 3.5. *Two-dimensional block DCT for lossless compression.*

basis has also been considered. In this method, the wavelet coefficients are first rounded to the next integer (NINT) and are quantized depending on the scale of resolution. Because the coarser scales have more information content than the finer scales, wider quantization bins has been selected for the finer scales. The image is reconstructed from these quantized coefficients and the quantization error is obtained. The quantized coefficients and the quantization error are coded using arithmetic coding. The quantization levels are then adjusted so as to maximize the overall compression ratio.

The hybrid transform algorithm applies one-dimensional DCT to each row and the difference between adjacent points in each column. The coefficients are then quantized and coded along with the quantization error using arithmetic coding.

3.3.2 Results and comparison

Different wavelet-based algorithms were applied to MRI and USC images and the results are compared with results obtained with existing techniques. Results of wavelet-based methods are also compared with results obtained with 'GZIP', the 2-D DCT and the 'hybrid' transform. The USC images are of size 256× 256 pixels with 8 bits per pixel whereas the MRI images are of size 256× 256 pixels and 16 bits per pixel. Table 3.1 shows the results obtained by using nonlinear TS transform, GZIP (LZ-77), 'hybrid' transform, 2-D DCT, wavelet transform and the proposed nonlinear TT transform on USC images. The 'hybrid'

Table 3.1. *Lossless compression ratios of USC images*

Test Image	GZIP	Hybrid DCT	2-D lossless DCT	D_6 wavelet	Nonlinear wavelets	
					TS filter	TT filter
Lena	1.20	1.40	1.33	1.63	1.51	1.75
Man1	1.29	1.58	1.40	1.64	1.55	1.74
Man2	1.25	1.51	1.35	1.43	1.27	1.56
Couple	1.47	1.56	1.47	1.63	1.45	1.75
Camera	1.33	1.35	1.28	1.46	1.25	1.53
Plane	1.23	1.44	1.49	1.46	1.24	1.56
Average	1.26	1.47	1.39	1.54	1.38	1.65

transform algorithm is based on applying 1-D DCT to each row in the image and a one-dimensional differentiator to adjacent points in each column. The 2-D DCT method involves 2-D 'block' DCT along with quantization. The block diagram is given in Figure 3.5. Results were also compared by using the wavelet transform alone. The block diagram showing the implementation of the wavelet transform for lossless compression is the same as for lossless DCT except that the wavelet transform is used instead of the two-dimensional block DCT. The Daubechies D_6 wavelet transform was used for compression. Bigger quantization bins were used for the finest scale as compared to the coarsest scale. Different quantization bins were selected and the overall compression ratios measured. The bin widths which gave the highest compression ratio were selected. Different quantization bins were selected and the overall compression ratios measured. As can be seen, the proposed nonlinear TT transform leads to the highest compression ratio. Note that the wavelet coefficients were all stored in integer format. Various 256× 256, 8-bit/pixel, MRI images and 512×512, 12-bit/pixel, CT images were compressed using the proposed technique and the techniques discussed in the previous section. The test images are shown in Figure 3.6. The performance of the various techniques is compared with respect to compression ratios and computation time.

Table 3.2 shows the compression ratios obtained for these four test images using integer-arithmetic Haar and TS transforms, GZIP (LZ-77), and the 'hybrid' transform. From Table 3.2, it can be seen that the proposed method using the nonlinear TT filter leads to the highest compression ratios. The better performance of the TT filter over the Haar and TS filters can be explained based on the fact that the higher order of the TT filter leads to a smoother dual wavelet function [3]. This leads to wavelet coefficients with lower entropy

Table 3.2. *Compression ratios for medical test images after pre-processing*

Algorithm type		Test Images				
		MRI Images			CT Images	
		Image 1	Image 2	Image 3	Image 1	Image 2
GZIP [63]		1.30	1.18	1.20	1.33	1.80
Hybrid DCT [74]		1.25	1.16	1.15	1.36	1.55
2-D lossless DCT [69]		1.63	1.50	1.48	1.44	1.54
Predictive Coding	JPEG4 [71]	1.43	1.25	1.27	1.46	1.89
	Modified JPEG(D3) [71]	1.46	1.30	1.33	1.42	1.82
D_6 wavelet		1.44	1.39	1.39	1.78	1.82
Nonlinear wavelets	Haar approach [10]	1.52	1.45	1.46	1.89	2.07
	TS approach [10]	1.58	1.50	1.51	1.91	2.05
	TT approach	1.71	1.60	1.59	2.00	2.12

for the TT transform than the coefficients obtained using the Haar and TS transforms. In support of this argument, the entropy values of various test images are calculated and given in Table 3.3. Clearly the nonlinear TT filter gives the lowest entropy as compared to the nonlinear Haar and nonlinear TS filters. This is supported by plotting the histograms of the wavelet coefficients of the MRI Image 1 after three stages of decomposition (see Figure 3.7). Clearly the nonlinear TT filter-based transform has the maximum number of zero-valued coefficients. Similar results were obtained for other MRI and CT images. Figure 3.8 gives a plot of compression ratios (bits per pixel) for various compression algorithms. As can be observed, the proposed method performs better than all the existing compression techniques.

The computation time required for compression using the nonlinear Haar, nonlinear TS, and nonlinear TT transforms has been calculated and is presented in Tables 3.4 and 3.5. As expected, the nonlinear Haar wavelet transform took the least CPU time. Arithmetic coding of the wavelet coefficients was the fastest for the TT approach, which can be explained by the lower entropy of the wavelet coefficients obtained using the nonlinear TT filter. Since the CPU time for arithmetic coding is much greater than the time required to compute the wavelet transform, the total computation time was the least for the nonlinear TT wavelet transform.

Note that the D_6 wavelet transform gave better compression ratios than the TS approach. However, the D_6 transform requires 6 floating-point multiplications for a wavelet

Table 3.3. *Entropies of test images for the Haar, TS and TT filter-based approaches*

Image	Haar approach	TS approach	TT approach
MR image 1	5.00	4.82	4.49
MR image 2	5.22	5.04	4.76
MR image 3	5.19	5.03	4.79
CT image 1	5.95	5.93	5.63
CT image 2	6.11	6.20	6.02

Table 3.4. *CPU time (in seconds) for test images using Haar and TS approaches*

Image	Haar filter			TS filter		
	Transform	Coding	Total	Transform	Coding	Total
† MR image 1	0.78	4.03	4.81	1.11	3.92	5.03
† MR image 2	0.79	3.99	4.78	1.13	3.75	4.88
† MR image 3	0.85	3.94	4.79	1.11	3.82	4.93
‡ CT image 1	3.47	23.12	26.59	4.55	22.87	27.42
‡ CT image 2	3.32	27.42	30.74	4.39	27.27	31.66

† 256× 256 image

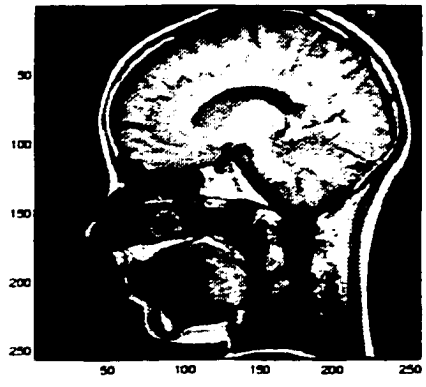
‡ 512 × 512 image

coefficient at each stage as opposed to the nonlinear TS transform which requires only shifts and additions and the nonlinear TT transform which requires 2 integer multiplications, shifts and additions. Thus, the D_6 wavelet transform involves more computations than the nonlinear TT and TS transforms.

3.3.2.1 Reason for the better performance of the nonlinear TT filter

The reason for the better performance of the nonlinear TT filter can be attributed to the smoothness (regularity) of the dual-wavelet function [3]. If ψ is the wavelet function and $\tilde{\psi}$ is the dual-wavelet function, then it is stated in [3] that

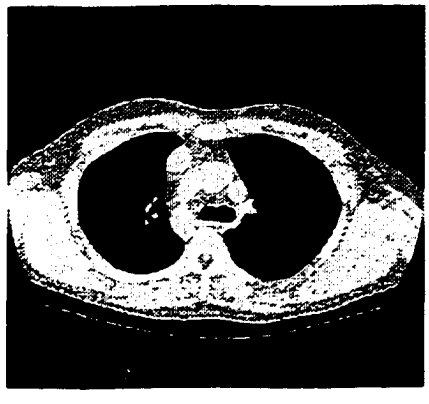
- for the same number of vanishing moments for ψ, the scheme with more regular $\tilde{\psi}$ is likely to yield the best performance,
- increasing the regularity of $\tilde{\psi}$, even at the expense of the number of vanishing moments for ψ, may lead better results.



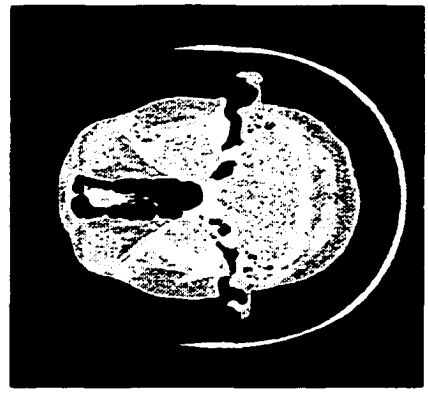
MR image 1

MR image 2

MR image 3

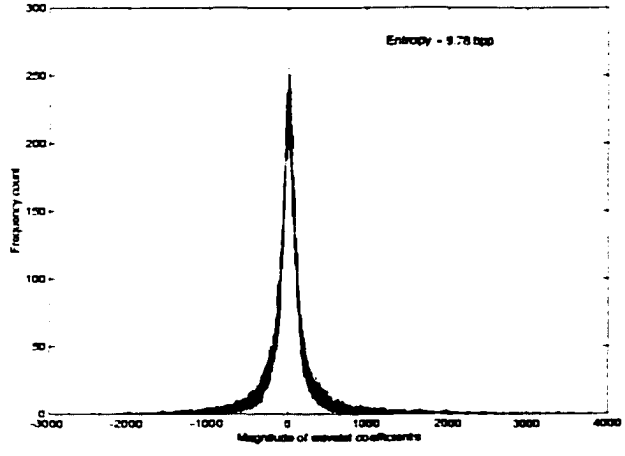


CT image 1

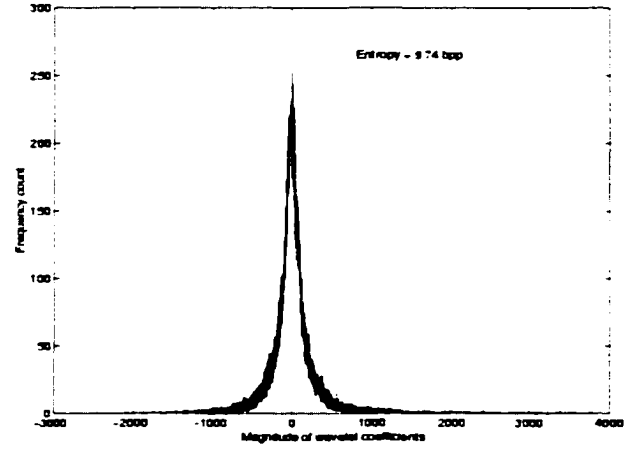


CT image 2

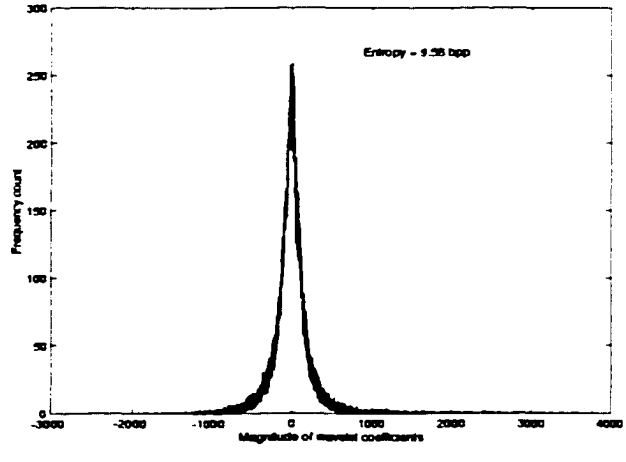
Figure 3.6. *Various medical test images.*



Histogram of Haar wavelet coefficients



Histogram of TS wavelet coefficients



Histogram of TT wavelet coefficient

Figure 3.7. *Histograms of the Haar, TS and TT wavelet coefficients.*

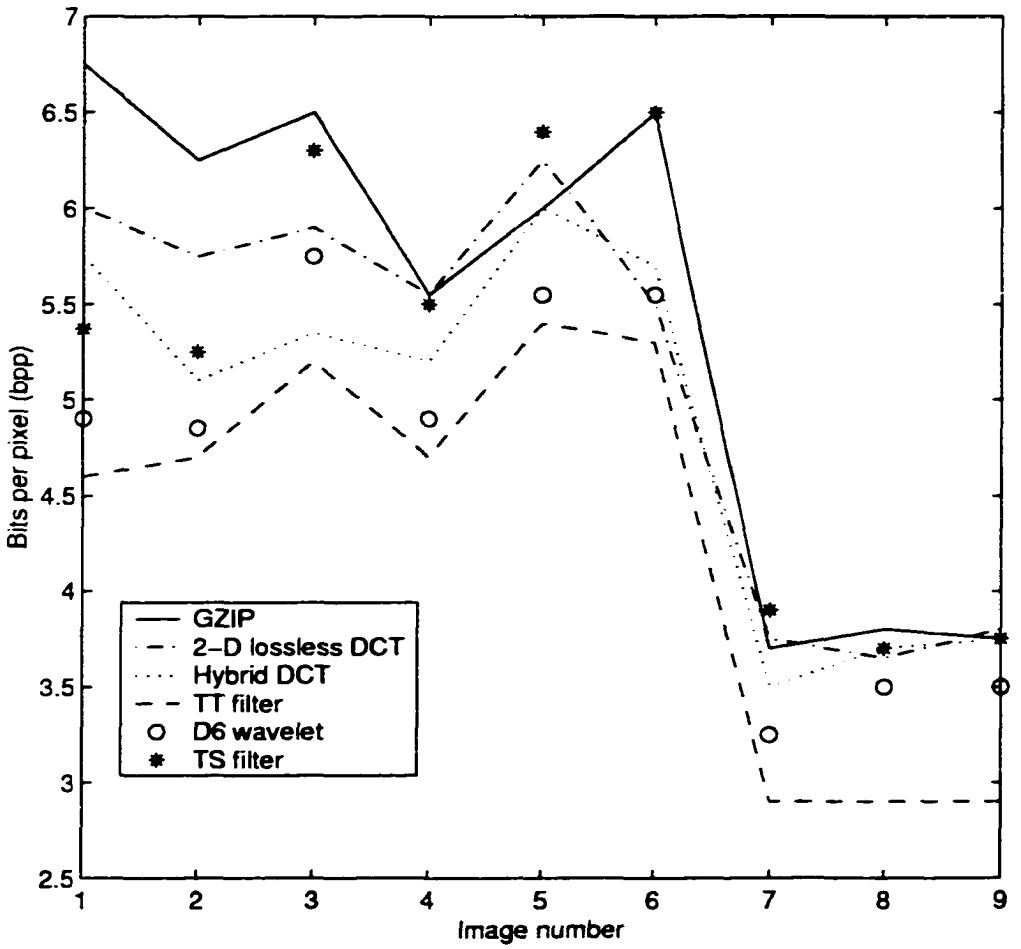


Figure 3.8. *Comparison of compression performance (in bits per pixel) using different algorithms for various USC and MRI images.*

Table 3.5. *CPU time (in seconds) for test images using the TT approach*

Image	TT filter		
	Transform	Coding	Total
† MR image 1	1.32	3.45	4.77
† MR image 2	1.29	3.44	4.73
† MR image 3	1.28	3.63	4.91
‡ CT image 1	5.35	21.22	26.57
‡ CT image 2	5.83	24.69	30.52

† 256× 256 image

‡ 512 × 512 image

Table 3.6. *Filter coefficients for various linear biorthogonal filters*

Filter type	Filter coefficients	
	analysis	synthesis
TS filter	$\frac{1}{\sqrt{2}}$[1,1]	$\frac{1}{8\sqrt{2}}$[-1, 1, 8, 8, 1, -1]
TT filter	$\frac{1}{\sqrt{2}}$[1,1]	$\frac{1}{128\sqrt{2}}$[3, -3, -22, 22, 128, 128, 22, -22, -3, 3]
2/14 filter	$\frac{1}{\sqrt{2}}$[1,1]	$\frac{1}{128\sqrt{2}}$[-5, 5, 44, -44, -201, 201, 1024, 1024, 201, -201, -44, 44, 5, -5]

In our case, the TT dual-wavelet function is much more regular than the TS dual-wavelet function thus supporting our claim for better performance of the nonlinear TT filter. Moreover, there is hardly any improvement in the regularity (smoothness) of the dual-wavelet function for filters of longer length than that of the highpass TT filter. As a result, the total entropy of the wavelet coefficients after multiscale decomposition was less for the TT filter (longer kernel) than the TS and Haar filters (shorter kernels). It was also noted that increasing the filter length of the highpass filter above 10 does not increase the smoothness of the dual wavelet function but instead increases the strength of the side peaks, as shown in Figure 3.9. Hence it can be concluded that the TT filter is the optimum in the series of reversible integer arithmetic wavelets. The filter coefficients for the TS, TT and 2/14 filters are given in Table 3.6 [3]. The plots of the dual-wavelet function for the TS and TT filters are given in Figures 3.10 and 3.11 respectively.

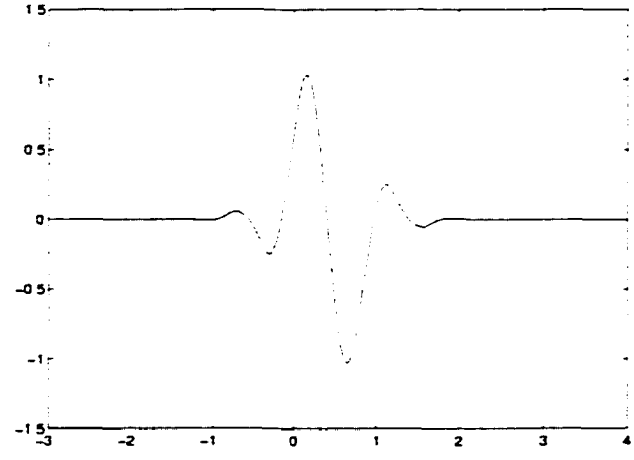


Figure 3.9. *Dual-wavelet function for the 2/14 filter.*

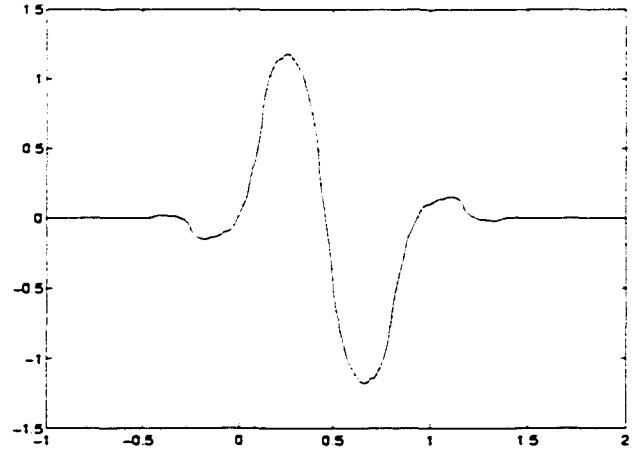


Figure 3.10. *Dual-wavelet function for the TS filter.*

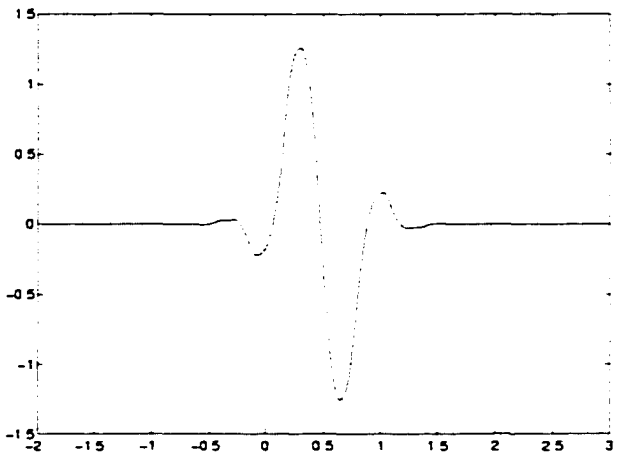


Figure 3.11. *Dual-wavelet function for the TT filter.*

3.4 An integer-arithmetic color-coordinate transformation for color image compression

In the previous section, the performance of the reversible TT transform was compared with that of other existing transforms. The remaining sections of this chapter will deal with the compression of color images. In this section, a new color image transformation, RGB to $Y'I'Q'$, a modified version of RGB to YIQ transformation, is proposed and it will be referred to as the transformation method. This transformation is based on integer arithmetic involving only integer additions and shift operations. The $Y'I'Q'$ image is then transformed to the wavelet domain by using a modified reversible integer-arithmetic wavelet transformation similar to one proposed in [76], [10]. The RGB to YIQ mapping is defined by [77]

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.30 & 0.59 & 0.11 \\ 0.60 & -0.28 & -0.32 \\ 0.21 & -0.52 & 0.31 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.18}$$

For compatibility with integer arithmetic hardware, the YIQ coefficients are quantized (rounding/truncation) thus resulting in quantization error. For the system to be lossless, the integer YIQ coefficients as well as the quantization error need to be compressed. In order to avoid these quantization errors, a new transformation based on integer arithmetic is proposed. The transformation is completely reversible and hence lossless. The forward

transformation RGB to $Y'I'Q'$ is given by

$$Y' = \left\lfloor \frac{\left\lfloor \frac{R+B}{2} \right\rfloor + G}{2} \right\rfloor \tag{3.19}$$

$$I' = R - B \tag{3.20}$$

$$Q' = \left\lfloor \frac{R+B}{2} \right\rfloor - G \tag{3.21}$$

where $\lfloor . \rfloor$ denotes the floor function. The inverse transformation is given by

$$R = Y + \left\lfloor \frac{Q+1}{2} \right\rfloor + \left\lfloor \frac{I+1}{2} \right\rfloor \tag{3.22}$$

$$G = Y - \left\lfloor \frac{Q}{2} \right\rfloor \tag{3.23}$$

$$B = Y + \left\lfloor \frac{Q+1}{2} \right\rfloor - \left\lfloor \frac{I}{2} \right\rfloor \tag{3.24}$$

It is observed that the Y coefficient (luminance) given by Equation 3.18 is very close to the Y' coefficient given by Equation 3.19. The similarity between Y (luminance in YIQ mapping) and Y' (luminance in $Y'I'Q'$ mapping) was noted by calculating the difference between the two matrices. The root-mean-square criterion (RMS) was used for comparison. The average relative error (in percent) was also calculated for different test images. The results are given in Table 3.4. The Y and Y' for a typical color image, Image 1, are given in Figures 3.12(a) and 3.12(b), respectively. As can be observed, there is hardly any visual difference between the two images. Thus, as in the YIQ model, the Y' coefficients in the $Y'I'Q'$ model are consistent with the sensitivity of the human visual system. It can also be observed that the RGB to YIQ transformation requires 9 floating-point multiplications and 8 floating-point additions whereas the RGB to $Y'I'Q'$ requires 5 integer additions and 2 shift operations.

3.5 Compression of color images

The original color images were stored in RGB format. Each of the three primary-color images used 8 bits per pixel (bpp) and was 256×256 pixels in size. Three different color image representations, the RGB, the YIQ and the proposed $Y'I'Q'$, were used for comparison.

(a)

(b)

Figure 3.12. *Comparison between luminance components of the existing YIQ and the proposed $Y'I'Q'$ models.*

Table 3.7. *Comparison between Y (luminance in YIQ mapping) and Y' (luminance in $Y'I'Q'$ mapping)*

Test Image	RMSE	Average Relative Error (in %)
Image 1	8.30	6.0
Image 2	8.27	7.0
Image 3	4.59	3.0
Image 4	3.37	2.0
Image 5	4.21	2.5

After converting each image to the required representation, the nonlinear wavelet transform was applied successively to the columns and rows of the image. The procedure was slightly modified for the YIQ representation. The floating-point Y, I, and Q coefficients were first quantized to the nearest integer. The nonlinear wavelet transform was then applied to the quantized coefficients. The quantization error as well as the wavelet coefficients of the quantized YIQ coefficients were then coded using arithmetic coding. Table 3.8 shows the compression performance in terms of bits per pixel (bpp) for five typical hard-to-compress color images. The results from the proposed nonlinear wavelet transform based on TT filters are compared with results obtained using the existing Haar and TS filter based transforms. The results are also compared with results obtained with the commonly used third-order two-dimensional global predictor in [72], which is of the form

$$X = 0.75A - 0.50B + 0.75C \tag{3.25}$$

where X, A, B, and C are given in Figure 3.4. It is observed that the proposed $Y'I'Q'$ color image format gives the best performance among all the image formats discussed. Also, the TT filter based approach gives better performance than the Haar and TS filter based transform. In general, the reversible wavelet based transform coding proved a better compression technique than the existing predictive coders. It should also be noted that although the RGB to YIQ transformation (based on real number arithmetic) gives better lossy compression than the standard RGB format, the same is not valid in the lossless domain. The main reason for this anomaly is that the real RGB to YIQ transformation is not a reversible transformation in integer arithmetic, thus leading to quantization errors. As far as the computational complexity is concerned, the total computation time is the sum of the time taken to perform the wavelet transformation and the time taken to code

Table 3.8. *Comparison of compression performance (bpp) among different lossless compression techniques*

Test Image	Format	Reversible wavelets			Predictive coding
		Haar	TS	TT	
Image 1	RGB	4.68	5.03	4.93	6.15
	YIQ	5.12	4.79	4.73	5.76
	$Y'I'Q'$	4.68	4.62	4.54	5.23
Image 2	RGB	6.00	6.11	6.52	6.59
	YIQ	5.91	5.79	5.26	6.52
	$Y'I'Q'$	5.35	5.09	4.62	6.03
Image 3	RGB	7.55	7.36	7.02	7.92
	YIQ	7.22	7.22	6.88	6.92
	$Y'I'Q'$	7.22	7.12	6.22	6.83
Image 4	RGB	6.92	6.96	6.75	6.53
	YIQ	6.56	6.15	5.26	6.29
	$Y'I'Q'$	6.38	6.15	5.13	6.21
Image 5	RGB	6.09	6.15	5.92	6.75
	YIQ	6.10	6.00	5.51	6.52
	$Y'I'Q'$	6.00	5.89	5.22	6.10

the coefficients. Although it took less time to compute the wavelet transform using the Haar and the TS approach as compared to the proposed method, the coding time for the TT approach was the least. This is because the wavelet coefficients obtained from the TT-filter based integer-arithmetic wavelet transform had the lowest entropy. Table 3.9 gives a comparison of the total computation times (in seconds) for different transform coders. The time was calculated based on the number of clock cycles required for the computation and coding of wavelet transform coefficients (with three stages of decomposition) on a SUN SPARC-5 workstation. As expected, the TT filter based approach took to least time for coding while the predictive coder took the maximum time of all the coders.

3.6 Conclusions

A reversible wavelet transform based on the TT transform was proposed. The transform compared with other transforms of its class, namely, the Haar and the TS transform. Vari-

Table 3.9. *Total coding times of color images ($Y'I'Q'$) for different coding techniques*

Test Image	Reversible wavelets			Predictive coding
	Haar	TS	TT	
Image 1	8.7	8.8	8.4	8.3
Image 2	8.9	9.0	8.4	10.8
Image 3	9.0	8.3	8.3	11.1
Image 4	8.9	8.7	7.5	10.4
Image 5	7.4	7.1	7.0	12.1

ous algorithms were used to compress a wide class of medical images. The results obtained indicate that the TT transform performs the overall best in its class of transforms. Preliminary results obtained with the S+P transform indicate its inferior performance relative to the TT transform.

A reversible transformation was also proposed for the compression of color images. The transformation transforms pixels from the color space RGB to a new color space $Y'I'Q'$. This transformation removes redundancy from the spectral bands and also maps integer input values to integer output values. The results obtained were compared with results obtained with the JPEG compression standard.

Chapter 4

A Mixed Transform Technique for Lossy Image Compression

4.1 Introduction

Subband transform coding based on wavelets has shown very promising results in terms of bit rate vs. distortion [9], [11], [30]. This is due to the capability of wavelets to represent signals with the least number of nonzero coefficients. A clear advantage has been shown when some other orthogonal basis (such as the DCT) is applied on the wavelet coefficients [30]. For example, results indicate [22] that higher coding gains can be obtained when such a multi-transform (also called a *mixed transform*) technique is applied to images. Many types of transforms have been used in mixed transform techniques apart from the DCT. For example, [22] uses a combination of subband, DFT, and Walsh transforms for efficient coding of non-stationary speech signals. However, this technique has a major disadvantage, namely, there is a need to select weighting coefficients to minimize the overall entropy of the signal. This is a multidimensional optimization problem and turns out to be even harder when applied to images. Because of these constraints, we use the highly efficient DCT with subband coding for image compression. The subband decomposition splits the whole image into a hierarchy of bandlimited components. Each of these components has a different statistical characteristic, and by designing a quantizer for each band optimally, the quantized data entropy can be drastically reduced without significant degradation in quality. Scalar quantization and independent encoding of signal samples is computationally simple but statistical dependencies between samples cannot be removed. On the other hand, vector quantization (VQ) can approach the rate distortion limit as the vector dimensionality becomes large. This quantization step is the only irreversible transformation and hence results in loss of information. The quantized coefficients are later coded using arithmetic

coding.

Another issue worth discussing is performance evaluation of the compressed images. Peak signal-to-noise ratio (PSNR) has been widely used as a subjective measure of image quality [78]. But this criterion is not that well suited when one talks about color images. The main reason is that the human eye responds differently to different colors [78].

The PSNR is not a good criterion for the evaluation of image quality in the red-green-blue (RGB) co-ordinate system. It can be shown that an image with lower PSNR might have a better subjective performance than another image with higher PSNR. Because, of this, many other co-ordinate systems have been proposed that provide better match with the response of the human eye [78]. One of the important ones is the uniform chromaticity scale (UCS) system. It has been shown that there is a close correlation between error in this system and subjective evaluation of color images.

In this chapter, the concept of mixed transform techniques is considered. Biorthogonal wavelets are used in the subband decomposition and the DCT is used for the coarsest scale subband coefficients. The quantization step is divided into two parts. Vector quantization is used for the wavelet coefficients and scalar quantization for the wavelets+DCT coefficients from the coarsest scale. Vector quantization is based on fixed codebook size vectors as discussed in [14]. Scalar quantization is applied to the wavelet+DCT coefficients mainly because there are well-known quantization tables for DCT coefficients [18]. After encoding, the quality of the compressed image is judged using objective and subjective evaluations. Objective evaluations were obtained using the mean-square error in the RGB system and in the $U*V*W*$ system [78]. Subjective evaluations were carried out by showing the images to a number of independent observers and averaging their responses on a scale of 1 to 10.

The chapter is organized as follows. In the next section, a brief overview of fixed-length vector quantizer is given. Later, the proposed compression technique is given in Section 4.3 followed by discussion on performance criteria for the evaluation of color images. Section 4.6 presents compression results using the proposed technique and are compared with the results obtained with the existing JPEG technique and other methods.

4.2 Overview of vector quantization

4.2.1 Introduction

According to Shannon's rate-distortion theory [79], one can always obtain a better performance by coding vectors instead of scalars. Vector quantization has begun gaining interest

with the advent of the codebook design algorithm introduced by Linde, Buzo and Gray [12] called the LBG algorithm. VQ has been found to be an efficient coding technique for speech as well as images. However, it requires high computational complexity. In this section, first vector quantization is defined as an image coding technique and then the optimality criteria for vector quantizers are discussed. Later, the LBG algorithm is presented and some important factors affecting the codebook design are discussed.

4.2.2 Definition of vector quantization

Vector quantization can be defined as a mapping Q of the K-dimensional Euclidean space R^k into a finite subset Y of R^k, i.e.,

$$Q : R^k \rightarrow Y, \tag{4.1}$$

where $Y = y_1, y_2, \ldots, y_N$ is the set of reproduction vectors, and N is the number of vectors in Y. VQ can also be defined as a combination of two functions: coding and decoding. Coding C is defined as a mapping of R^k into the index set J, and decoding D is defined as the mapping of J into the output set Y where $J = 1, 2, \ldots, N$. Thus,

$$C : R^k \rightarrow J \quad \text{and} D : J \rightarrow R^k \tag{4.2}$$

An input vector $\mathbf{x}$ is quantized by selecting the nearest reproduction vector from the codebook according to a distortion criterion $d(\mathbf{x}, \mathbf{y}_i)$

$$Q(\mathbf{x}) = \min^{-1}[d(\mathbf{x}, \mathbf{y}_i)] \quad i = 1, 2, \ldots, N \tag{4.3}$$

This process is done by the encoder which then sends the address of the reproduction vector nearest to x over the channel. The decoder receives the address of the reproduction vector and simply performs a table lookup operation on the same copy of the codebook that the encoder uses. Thus, VQ is used for applications that require very fast decoding.

If the codebook length is N, i.e., the number of codebook vectors is N and each vector consists of k pixel elements, then the rate of the quantizer (in bits per pixel) is given by

$$R = \frac{1}{k} \log_2 N \quad \Rightarrow N = 2^{Rk} \tag{4.4}$$

From the above relationship, it is clear that the codebook size N, and hence the complexity of the VQ, increases exponentially with the rate and block size. In order to reduce the search complexity, the codebook size must be kept small. This puts limits on the utilization of the source coding theorem which states that the larger the value of k, the closer we approach to rate distortion function. Figure 4.1 shows the block diagram of a basic memoryless vector quantizer.

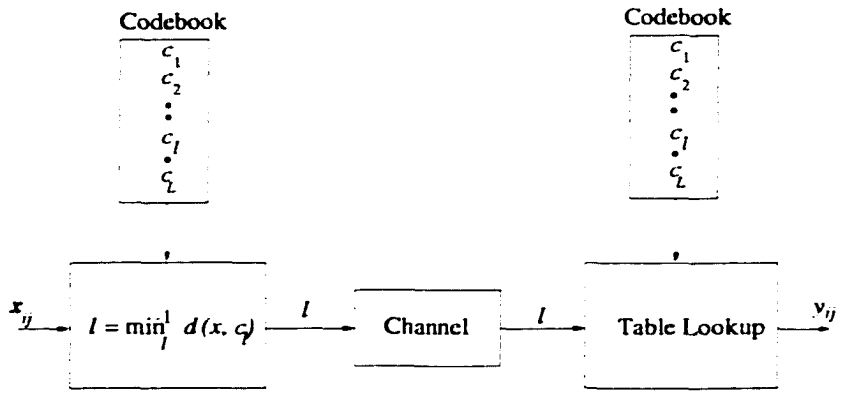


Figure 4.1. *A basic memoryless vector quantizer*

4.2.3 Optimal vector quantization

A vector quantizer is considered to be optimal if it minimizes the average distortion $E[d(\mathbf{x}, \mathbf{y}]$, where $\mathbf{x}$ is the input vector (with uniform probability distribution function) and $\mathbf{y}$ is the best reproduction vector matched to it. There are two necessary conditions for a quantizer to be optimal [80] :

Condition 1 : Given a codebook $Y = y_i : i = 1, 2, \ldots, N$ and a specific decoder D, the encoder encodes a vector $\mathbf{x}$ by selecting the codeword which yields the minimum distortion, given by

$$Q(\mathbf{x}) = \min^{-1} d(\mathbf{x}, y_i) \quad i = 1, 2, \ldots, N \tag{4.5}$$

By applying this condition, the encoder partitions the input space into several partitions S_i, where the vectors that are encoded to the same codeword are placed in the same partition. Such partitions, according to a minimum distortion rule, are called Voronoi or Dirichlet partitions [81].

Condition 2 : Given the Voronoi partitions of the input space, we need to find the codebook that minimizes the overall distortion. To minimize the overall distortion, the distortion contributed by each cell needs to be minimized. This could be done by assigning the generalized centroid of each partition as a codeword in the codebook

$$\text{Cent}(S_i) = \min E(d(\mathbf{x}, y) \mid \mathbf{x} \in S_i) \quad \text{for all } y \in R^k \tag{4.6}$$

The generalized centroid of partition S_i under a distortion measure $d(.,.)$ is unique if the distortion is strictly convex [82]. Since the source is assumed to be ergodic, the conditional

expectation may be replaced by a simple sample average as

$$\mathrm{Cen}(S_i) = \frac{1}{\| S_i \|} \sum_{x_i \in S_i} x_i \tag{4.7}$$

where $\| S_i \|$ denotes the cardinality of the partition.

The two conditions in Equations 4.5 and 4.6 are the key of the codebook design algorithm. The codebook is iteratively optimized for the old encoder and then a minimum distortion encoder is used for the new codebook.

4.2.4 The LBG algorithm

The two necessary conditions mentioned above lead naturally to an iterative algorithm for optimal VQ design called the LBG algorithm [12]. The algorithm works either with a probability density function or with the training set (t-set) of sample vectors from the true distribution. The LBG algorithm for an unknown distribution training sequence is as follows:

1. Let N be the number of codewords in the codebook and let the distortion threshold $\epsilon \geq 0$. Assume that $\hat{A}_0$ as the initial codebook which has N codewords, a training sequence $(x_j; j = 0, 1, \ldots, n-1)$ with all vectors in the optimal codebook, $\hat{x}(i)$; $i = 0, 1, 2, \ldots, N-1$, set to zero, and $D_{-1} = \infty$ (D_i is the average distortion at iteration i). Also assume that m is the number of iterations.
2. Given $\hat{A}_m = (y_i; i = 1, 2, \ldots, N)$, find the minimum distortion partition $P(\hat{A}_m) = (S_i; i = 1, 2, \ldots, N)$ of the training sequence such that $x_j \in S_i$ iff $d(x_j, y_i) \leq d(x_j, y_l)$, for all l. Compute the average distortion

$$D_m = \frac{1}{n} \sum_{j=0}^{n-1} \min_{y \in \hat{A}_m} d(x_j, y) \tag{4.8}$$

3. If $(D_{m-1} - D_m)/D_m \leq \epsilon$, stop the iteration with $\hat{A}_m$ as the final codebook, otherwise continue.
4. Find the optimal codebook $\hat{x}[P(\hat{A}_m)] = \hat{x}[(S_i); i = 1, 2, \ldots, N]$ for $P(\hat{A}_m)$ where $\hat{x}(S_i)$ is the centroid for the partition cell S_i.
5. Set $\hat{A}_{m+1} = \hat{x}(S_i)$, increment m to $m + 1$, and go to Step 2.

Since D_0 is finite, the distortion either decreases or remains the same at each iteration of the algorithm. The distortion threshold ϵ determines the number of iterations before the algorithm is terminated. If $\epsilon = 0$, the algorithm will not terminate unless it reaches a fixed point (i.e. applying the algorithm to a codebook C yields codebook C). For $\epsilon > 0$, the

algorithm must stop. For a training set of finite size n and for convex problems, this occurs in a finite number of steps.

The LBG algorithm does not assure convergence to the globally optimal codebook. However, depending on the size of the training set, it provides us with a set of fixed points which is arbitrarily close to the set of fixed points for the true distribution. Thus, the larger the size of training set, the closer are the codewords to those obtained with a true distribution. The designed codebook depends on the choice of an initial codebook. Different choices of initial codebooks produce locally optimal codebooks. The locally optimal codebook that gives the least average distortion is deemed to be the globally optimal one.

4.2.5 Codebook initialization

The performance of the final codebook could depend strongly upon the initial codebook chosen, especially for sources with multiple minima. Choosing an initial codebook that leads to the best final codebook is still an open problem. Some of the techniques studied in the past to generate an initial codebook are given below:

1. One possible way to initialize the LBG algorithm is by using a codebook previously designed for some other purpose. Alternatively, one might initialize with evenly spaced codewords in the vector space.
2. *Random Initialization*: In this technique, N codebook vectors are chosen at random for the t-set. This method was first used in the k-means clustering problem. In practice, the random initialization is implemented by choosing evenly spaced elements in the training sequence (e.g. $\mathbf{x}_1$, $\mathbf{x}_{k+1}$, $\mathbf{x}_{2k+1}$, $\ldots$, $\mathbf{x}_{(N-1)k+1}$, where $\mathbf{x}_i$ is the training vector, m is the number of training vectors, N is the number of codewords desired, and $k = m/N$).
3. *Product Codes*: Let us assume that we have a collection of codebooks C_i, $i = 0, 1, \ldots, m-1$, each containing N_i vectors of dimension K_i, and having a rate $R_i = \log_2 N_i$ bits per vector. Then, the product codebook C is defined as the collection of all $N = \prod_i N_i$ possible concatenations of m codewords taken successively from the m codebooks C_i. The dimension of the product codeword is $k = \sum_{i=0}^{m-1} k_i$. Thus, for example, a k-dimensional vector quantizer of rate R bits per vector can be formed using a scalar quantizer k times in succession, with rate R/k.
4. *Splitting Technique*: In this technique, the final codebook is constructed recursively from a smaller codebook having the same dimension. First, the optimum 0-rate code is found by finding the centroid of the entire training set. Then, a codebook of size

two is formed by splitting the first codeword into two. In order to ensure that the distortion will not increase, the original codeword is kept in the new codebook and the new codeword is formed by scaling the energy of the first. Having formed an initial codebook of size two, the LBG algorithm is then used to design an optimal codebook of size two. Using the same technique, an initial codebook of size four can be formed by two optimal codewords. Then, an optimal codebook having four codewords is designed. For an initial codebook of size N, this process is repeated $\log_2 N$ times.

5. *The Pairwise Nearest Neighbor (PNN) Algorithm* : The PNN clustering algorithm was introduced by Equitz [83] as an alternative to the LBG algorithm for vector quantizer design. It generates codebooks from a training set in approximately 5 percent of the time required by the LBG algorithm. The PNN algorithm starts by assigning a separate cluster for each training vector. Then, it merges two clusters at a time until the desired codebook size is achieved. When clusters are merged, the centroid is used to represent the vectors of the merged cluster. Once the clusters have more than one member, things become more complicated. A tradeoff is made between merging close clusters and the number of vectors affected by the merging process.

Even though the PNN algorithm is to be considered unacceptable because it usually does not generate an optimal codebook, it appears to be an excellent alternative to random initialization for the LBG algorithm. By using it as an initializer for the LBG algorithm, the total coding error is reduced and the LBG algorithm converges in fewer iterations.

4.2.6 Compression of images using vector quantization

In the previous subsection, some idea of various methods of choosing an initial set of vectors was provided. In this section, we shall discuss the method of applying vector quantization to image compression. Brief results obtained by applying the vector quantization alone, are also given.

Before the vector quantization is applied, the codebook needs to be initialized. In our approach, we initialize the codebook based on random selection. Once the size and length of the codebook is decided, the codebook vectors are updated based on the already described LBG algorithm. The compression algorithm is terminated when the mean-square error is brought below a certain predefined threshold value. The algorithm can be described in terms of a step-by-step process as follows:

Step 1 : Start with a predefined length and width for the codebook.

Step 2 : Initialize the codebook vectors using a random selection scheme.

Step 3 : Update the codebook vectors using the LBG algorithm.

Step 4 : Calculate the objective function (mean-square error in our case).

Step 5 : Repeat steps 3 and 4 until the value of the objective function reduces below a predefined threshold ϵ.

Step 6 : Output the index of the codebook vectors and the codebook vector table.

4.3 A new mixed-transform technique for low bit-rate image coding

The proposed technique is based on the wavelet transform of the original image. The resulting wavelet coefficients are compressed using VQ based on the Linde, Buzo and Gray (LBG) algorithm [12] while DCT is applied to the low-frequency scaling coefficients. This transform technique is referred to as the *MiXeD-Transform* (MXDT) technique. For the wavelet transform, a biorthogonal basis is used because it leads to better performance than an orthogonal basis [30]. The low-frequency content of the image is confined in the scaling coefficients and block DCT is applied to them. The resulting DCT coefficients are quantized based on the scalar quantization tables of the JPEG standard. They are stored using the zig-zag scanning pattern described in [18]. The CDF-97 filter set is used because of its proven superior performance compared to that of other existing filters [46]. The transfer functions $H_0(z)$ and $F_0(z)$ of the lowpass and highpass filters, respectively, for the biorthogonal 9/7 filter are given by

$$\begin{aligned} G(z) &= 0.037828 - 0.023849z^{-1} - 0.110624z^{-2} + 0.377402z^{-3} + 0.852699z^{-4} \\ &\quad +0.377402z^{-5} - 0.110624z^{-6} - 0.023849z^{-7} + 0.037828z^{-8} \\ H(z) &= 0.064539 - 0.040689z^{-1} - 0.418092z^{-2} + 0.788486z^{-3} - 0.418092z^{-4} \\ &\quad -0.040689z^{-5} + 0.064539z^{-6} \end{aligned}$$

The theory behind the biorthogonal wavelet basis can be found in [3], [46].

4.4 Proposed MXDT coder

The block diagram of the proposed MXDT coder is given in Figure 4.2. The wavelet transform based on the cascade algorithm [4] is applied to the input image followed by a full-search VQ of the wavelet coefficients. The LBG algorithm [12] is used to optimize the codebook generation. The quantization errors in the wavelet coefficients are also quantized using the

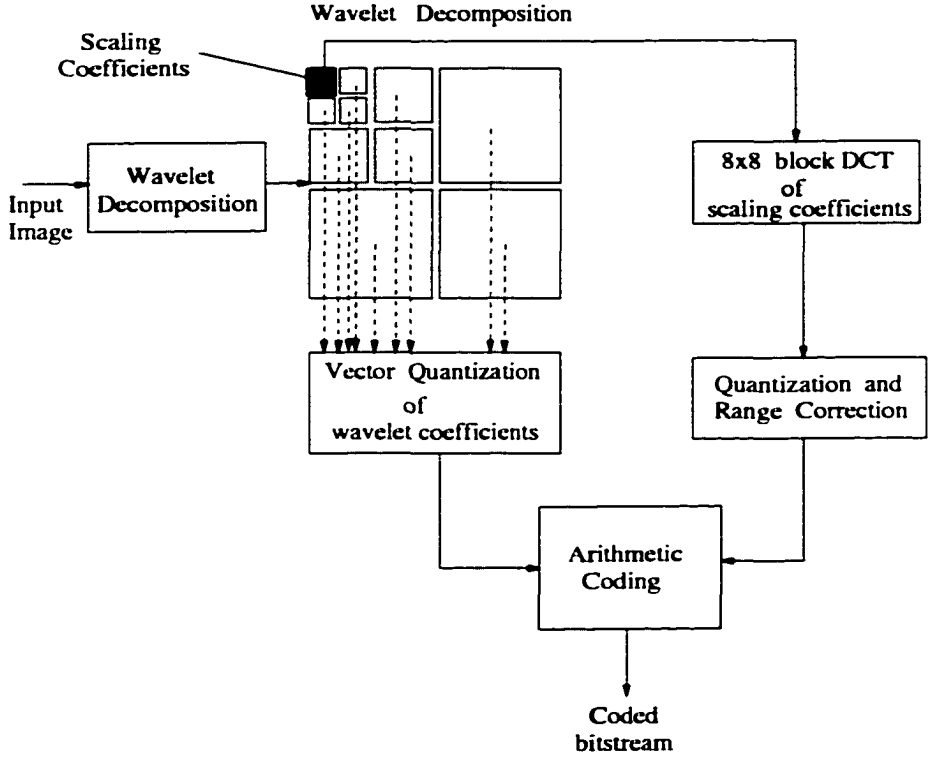


Figure 4.2. *Block diagram of the proposed MXDT coder.*

same codebook size and width. The latter step results in a significant improvement in the overall Peak signal-to-noise ratio (PSNR). Finally these codebook indices are encoded using arithmetic coding with a simple frequency-count-based probability predictor. The scaling coefficients contain almost entirely the low-frequency content, and thus, the combination of an 8×8 block DCT and scalar quantization based on the JPEG quantization tables given in [18] is used rather than VQ. The DC coefficients of each 8×8 block are subtracted from those of the first block and the resulting quantized DCT coefficients are passed through a range corrector. The range corrector performs indexing of all the available data values. For example, if the mixed transform coefficients have the values 1200, 50, −60, 20, −5, 3, −5, 3, −4, 0, 0, 0, 0, 0, 0, 0, the range corrector will check the data values and generate a map as shown in Table 4.1.

Table 4.1. *An example showing range correction of the DCT coefficients*

Data	Index	Codeword
−60	0	000
−5	1	001
−4	2	010
0	3	011
3	4	100
20	5	101
50	6	110
1200	7	111

The range of DCT coefficients in the above example is [−60, 1200] thus requiring 11 bits to code each data value. The above mapping reduces the number of bits per data value to 3. The map is coded in the compressed bit stream by sending a 1 or a 0 depending on whether a value is present in the input data or not. It was observed that this range correction can result in huge savings in memory due to the highly non-uniform distribution of the DCT coefficients.

The steps in the proposed coding algorithm are as follows:

Step 1 : Obtain the biorthogonal wavelet transform of the input image using the MIT-97 filters with three stages of decomposition.

Step 2 : Perform VQ on the wavelet coefficients followed by arithmetic coding.

Step 3 : Use an 8×8 block DCT on the scaling coefficients with scalar quantization of these mixed transform coefficients based on the JPEG quantization tables.

Step 4 : Perform range correction and arithmetic coding of the mixed transform coefficients scanned in a zig-zag scanning pattern.

4.4.1 Selection of codebook size and width

Table 4.2 shows the lengths and widths of the codebooks for different shapes and scales of the image. Shapes 0, 1, 2, and 3 refer to the scaling, horizontal, vertical, and diagonal

Table 4.2. *Codebook sizes and widths for a typical test image*

Scale	Shape	Codebook	
		Length	Width
1	1	0	–
	2	0	–
	3	0	–
2	1	128	32
	2	128	32
	3	128	64
3	1	128	16
	2	128	16
	3	128	16

component wavelet coefficients of an image, respectively.

4.4.2 Statistical analysis of the wavelet coefficients

Before we describe the algorithm, there is a need to study the statistics of each of the three shapes, i.e., Shapes 1, 2, and 3 (Shape 0 is just a coarse approximation). The statistics of the wavelet coefficients for the Lena image were studied for different scales and shapes. The following inferences were made from these results:

1. The range of coefficients and their variances grow as the resolution becomes coarser, which proves that the coefficients at lower levels are of higher importance.
2. After subtracting the average and normalizing the variances to unity, the range of coefficients become smaller and similar to all shapes in each level. This fact enables

us to *quantize different shapes with similar considerations.*

3. In the coarsest scale, Shape 0 consists of a wider range and larger coefficients and variances than the other shapes. Thus, these coefficients are quantized using only a scalar quantizer. This might lead to lower compression but it has the least size and maximum number of significant coefficients thus resulting in a higher reconstruction accuracy.

The sensitivity of the final reconstructed image quality to quantization error is more noticeable at coarser scales, necessitating larger allocation tables to coarser scales. Tests were conducted on a few test images with regard to different code lengths L and widths W. The relation of these parameters to the compression ratio has already been discussed in Section 2. In most cases, Shape 3 (diagonal) has much less detail and hence needs fewer quantization levels (wider bins) as compared to Shapes 1 and 2. Tests were also conducted by varying the codebook size and widths. It was observed that increasing the decomposition levels above three increased the compression ratio significantly but also reduced the PSNR below visually acceptable levels. As can be seen from Table 4.2, the finest scale wavelet coefficients (scale 1) are discarded completely.

4.4.3 Error correction

Applying vector quantization on wavelet coefficients does not always produce satisfactory results. The goal is to achieve higher PSNR for the reconstructed coefficients with higher compression ratios without allocating enormous tables that degrade the speed of the algorithm. One way is by using the error correction (EC) method. The quantization error of the reconstructed image is computed and the LBG quantization is applied on the error. We get an approximation of the error and add it to the previous reconstructed image. This yields better SNR for quantized coefficients and higher PSNR for the reconstructed image as well. The pictorial representation of the improved implementation based on the error correction method is given in Figure 4.3. In the remaining of this chapter, this modification is incorporated in our algorithm to achieve an improved compression performance.

A modification

There was a slight modification to the above approach that resulted in a significant improvement in the reconstructed image quality. Most of the time, two different images never have the same dynamic range. So, if one uses the codebook obtained by training from one image on the other image, compression results are very poor. So, we rescaled the second

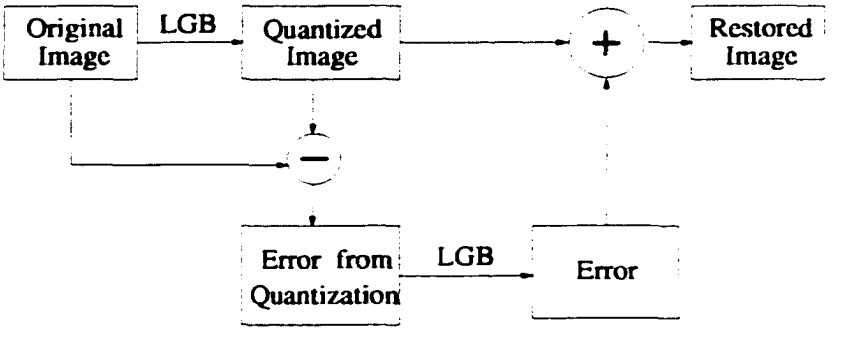


Figure 4.3. *The Error Correction method*

image so that its range is the same as that of the first image. Then, applying the codebook for coding the second image improved the image quality by almost 3 dB.

4.5 Decoders

A decoder for the proposed compression scheme can be easily obtained by simply reversing the steps of the MXDT encoder. Such a decoder can be referred to as the MXDT decoder. Although such an decoder is easy to implement, it has some disadvantages. It tends to introduce two types of artifacts that degrade the quality of the decompressed image. One type of artifacts is due to VQ of the wavelet coefficients and shows up as vertical/horizontal lines in the image, especially at sharp edges. Another type is due to JPEG quantization of the mixed-transform coefficients and shows up as spots in the decompressed image.

Although artifacts due to VQ are difficult to circumvent, much progress has been achieved in recent years in reducing artifacts due to the quantization of the DCT coefficients in a JPEG coder [84]-[87]. Below we develop an improved decoder for the proposed compression scheme on the basis of a technique described in [88] which will be referred to as decoder MXDT1.

The methods in [84], [85] require a novel decoder which completely eliminates the need of a JPEG decoder. In [87], a technique referred to as wavelet-based thresholding was used for the enhancement of JPEG decompressed images. Although there is no rigorous proof of why wavelet-based thresholding reduces blocking artifacts, an intuitive argument is as follows. Quantization in the JPEG coder tends to filter out the high frequencies from each 8×8 block based on the local statistics of that block. The only areas where the artifacts are prominent would be at the block boundaries. The sharp transitions at block boundaries are like edges in the image and block edges correspond to high frequencies in the image. Thus although the

JPEG coder filters out the high frequency components from each block, it introduces block edges which can be considered as correlated high-frequency noise. Thresholding has been applied by Donoho and others [88]- [90] for noise removal. These authors have shown that wavelet thresholding is an optimal criterion for noise removal from data with additive white noise. Although in our case the noise is correlated (block edges), the wavelet transform tends to whiten the data and hence while the error due to blocking artifacts, e, is not white, DWT$\{e\}$ might be. Although no extensive comparative study has been performed on various alternative methods with regards to removal/reduction of blocking artifacts, the wavelet-based method seems more promising than other known methods.

Decoder MXDT1 is based on applying soft thresholding on the scaling coefficients rather than the decompressed image itself. The reason is that DCT is applied on the scaling coefficients only. Soft thresholding is used because if the error is bounded, soft thresholding is optimal in the mean-square error sense [88], if smoothness is desired. Hence, a two-stage wavelet decomposition of the scaling coefficients (LL_0) is performed followed by soft thresholding of the coefficients. A two-stage wavelet decomposition would lead to seven subbands as shown in Figure 4.4. Let LL_0 be the subband corresponding to the scaling coefficients of the decompressed image and $LL_2, LH_2, HL_2, HH_2, HL_1, LH_1, HH_1$ be the subbands from a two-stage wavelet decomposition of LL_0. The scaling coefficients, LL_0', after the thresholding operation are given by

$$LL_0' = \text{IDWT}\{T_{soft}[\text{DWT}\{LL_0\}, \delta]\} \tag{4.9}$$

where IDWT is the inverse discrete wavelet transform and T_{soft} is Donoho's soft threshold given by

$$T_{soft}(x, \delta) = \begin{cases} \text{sign}(x)(|x| - \delta), & |x| > \delta \\ 0, & |x| \leq \delta \end{cases} \tag{4.10}$$

where

$$\delta = \lambda \hat{\sigma}(e) \tag{4.11}$$

is the thresholding level and $\hat{\sigma}(e)$ is the estimate of the standard deviation (std) of noise e in x. Donoho also showed that a reliable estimate of the std of noise is the std of the wavelet coefficients in the diagonal components in the finest scale. In our case, $\hat{\sigma}(e) = \text{std}(HH_1)$. The scaling factor λ in (9) is also a crucial parameter. Donoho used $\lambda = \sqrt{2\log(n)}$, where n is the image size. However, our experiments indicate that this estimate of λ results in oversmoothing of the image and brings down the PSNR drastically. On the other hand, $\lambda = 1.0$ works almost perfectly for most of our test images.

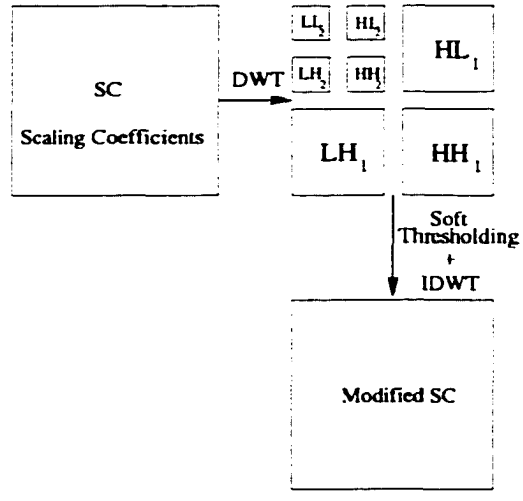


Figure 4.4. *Wavelet-based soft thresholding.*

The algorithm used in decoder MXDT1 is as follows:

Step 1 : Perform arithmetic decoding of the bit-stream to get the mixed transform coefficients and the quantized codebook vectors.

Step 2 : Apply an 8×8 block IDCT on the mixed transform coefficients to get the scaling coefficients (with quantization error).

Step 3 : Improve the visual appearance of the image using wavelet-based soft thresholding of the scaling coefficients.

Step 4 : Apply an inverse wavelet transform to obtain the decompressed image.

As will be seen in the next section, this simple modification in the decoder improves the visual image quality considerably.

4.6 Results and discussion

The proposed technique was tested with 19, 512×512, 8-bit, gray-scale facial images. Five of these images were used for training in the VQ algorithm.[1] The coder was implemented using three stages of wavelet decomposition. For the sake of comparison, results were also obtained using the EZW coder [9], the SPIHT coder [11], the wavelet+VQ coder [14], the

[1]The images can be downloaded from sequoyah.ncsl.nist.gov in the files "mugshots.txt" and "mugshots.tar.Z".

existing JPEG decoder[2], and the upcoming JPEG-2000 standard[3] using the PSNR, which is computed as

$$\text{PSNR} = 20\log\left(\frac{255}{\sqrt{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}|x_i(i,j)-x_o(i,j)|^2}}\right) \tag{4.12}$$

where $x_i(i,j)$ is the input image (8 bits per pixel) and $x_o(i,j)$ is the output image. The results for four of the 19 images are now examined. The first two images, FACE1 and FACE2, belong to the five-image training set while FACE3 and FACE4 do not. The results presented in Table 4.6 indicate that the decompressed images using the MXDT and MXDT1 decoders lead to an average PSNR of 34 dB at a compression ratio (CR) of 60:1 for images within the training set and a PSNR of 33 dB at almost the same CR for images outside the training set. The MXDT decoder gives slightly better PSNR than the MXDT1 decoder. However, as will be seen below, the MXDT1 decoder performs better in subjective evaluations. The method in [14] gave a slightly better PSNR than the MXDT and MXDT1 decoders at an average of only 45:1 compression. The only computational overhead in the MXDT decoder compared to the wavelet+VQ decoder in [14] involves computing the block DCT of the scaling coefficients. The JPEG-2000 coder performed best in terms of PSNR except for FACE3 image where the results were inferior to all the other techniques (except baseline-JPEG decoder). The performance of the EZW coder in general was a little inferior than the SPIHT coder.

The JPEG decoder performed much worse than the above schemes with an average PSNR of only 28 dB at a CR of about 60:1. Figures 4.5(a) and (b) show the original FACE1 and FACE4 images respectively. Figures 4.6(a), (b), (c), and (d) show the decompressed images using the JPEG decoder, the SPIHT decoder, the MXDT decoder, and the MXDT1 decoder, respectively. Figures 4.7(a)-(d) show the results for image FACE4.

The proposed schemes were also compared with the SPIHT technique with respect to subjective evaluation. This is a common practice because of the fact that the PSNR is not always consistent with the quality of the image [91], [92]. The images (FACE1 to FACE4) were rated on a scale of 10 by five independent observers. The original image and the JPEG decompressed image could be easily identified by the observers visually and they were rated 10 and 3, respectively. All the other decompressed images were rated within this range. The subjective tests were conducted on the computer screen with equal display

[2]A copy of the JPEG software can be downloaded from the Independent JPEG software group available via anonymous ftp at ftp.uu.net/graphics/jpeg/jpegsrc.v6a.tar.gz

[3]A software implementation of the JPEG-2000 standard (called *Jasper*) can be obtained from Image Power Inc. at http://www.imagepower.com

Table 4.3. *Comparison of compression performance of various schemes*

Technique used		Images			
		FACE1	FACE2	FACE3	FACE4
JPEG	CR PSNR (dB)	56:1 28.52	58:1 29.22	58:1 28.84	56:1 29.10
EZW	CR PSNR (dB)	58:1 33.21	61:1 36.17	58:1 36.81	59:1 33.01
SPIHT	CR PSNR (dB)	58:1 33.99	61:1 36.37	58:1 36.79	59:1 33.89
JPEG-2000 (JasPer)	CR PSNR (dB)	58:1 34.93	61:1 38.11	58:1 32.50	59:1 36.91
Wavelets+VQ	CR PSNR (dB)	42:1 34.09	43:1 35.05	45:1 35.17	41:1 33.73
MXDT	CR PSNR (dB)	58:1 33.40	61:1 34.31	58:1 34.38	59:1 33.08
MXDT1	CR PSNR (dB)	58:1 33.07	61:1 34.25	58:1 34.02	59:1 32.99

size for all the images. The original image was in the top left corner of the screen. All the decompressed images were of the same size and placed at random positions to avoid any spatial correlation among the observations. The background and lighting was also kept constant throughout the experiments. The results were then averaged out and are listed in Table 4.6. Results are not given for the EZW coder because there was no noticeable difference between the SPIHT and EZW coders in terms of subjective performance. It was observed that for images FACE2 and FACE3, the results from the MXDT1 and SPIHT decoders were visually indistinguishable. Moreover the quality of the decompressed images using the MXDT1 decoder is comparable to that obtained with the SPIHT decoder while the MXDT decoder was found to be inferior. Note that these results are not consistent with the PSNR evaluations.

A detailed study on the type of artifacts in different compression schemes is warranted. Figure 4.8(a) shows a zoomed area of the original FACE1 around the forehead. The skin marks in the original image are completely filtered out in the SPIHT coder (Figure 4.8(b)) but are present in the image using the MXDT1 decoder (Figure 4.8(c)). Figure 4.9(a) shows a zoomed portion of the original FACE4 image around the upper left corner. Figures 4.9(b) and (c) show the results obtained using the SPIHT coder and the MXDT1 decoder, respec-

Table 4.4. *Subjective evaluation of various schemes*

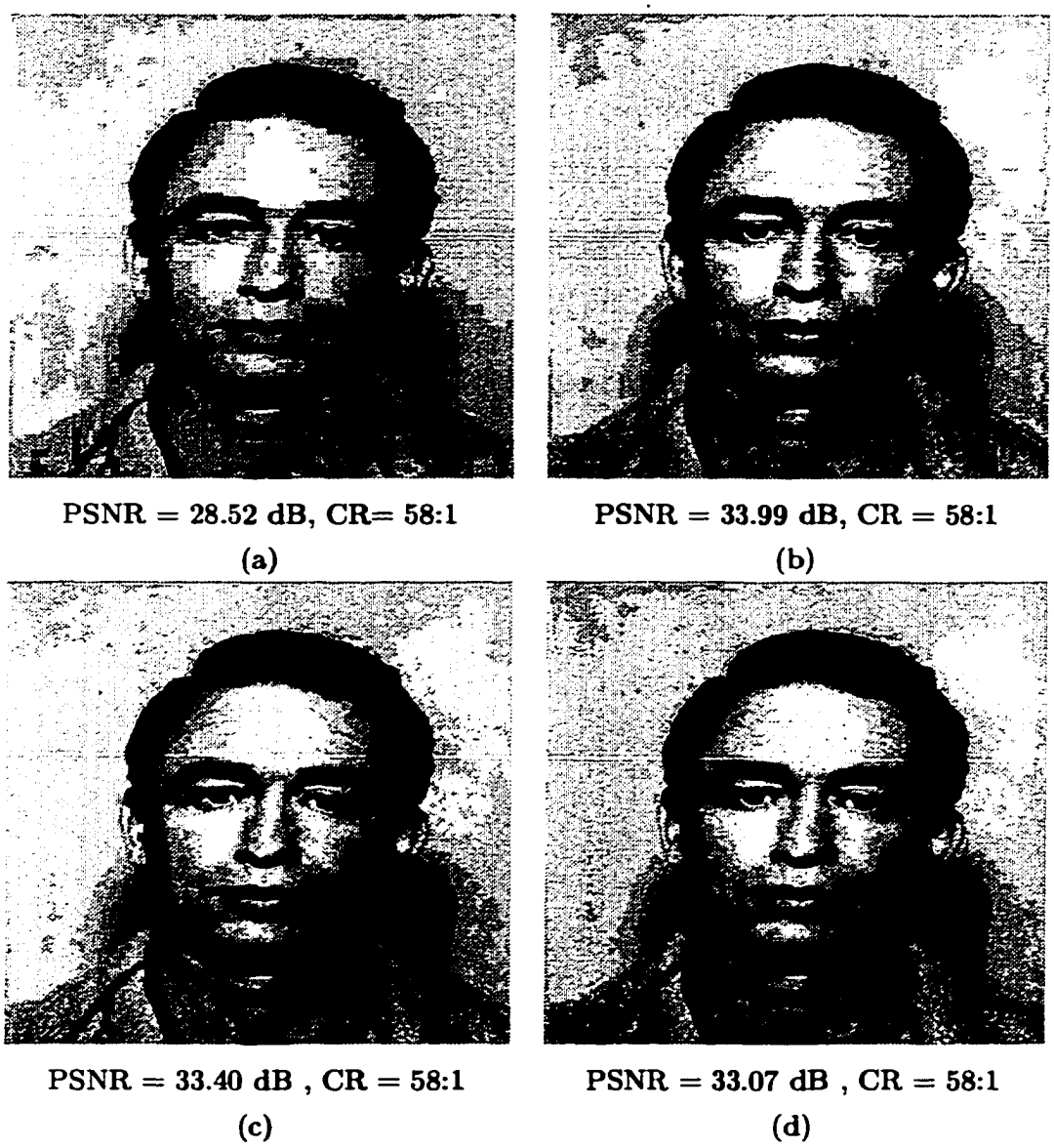
Image	Subjective			
	JPEG decoder	SPIHT decoder	MXDT decoder	MXDT1 decoder
FACE1	3.0	7.0	6.5	7.5
FACE2	3.0	7.5	6.0	7.5
FACE3	3.0	8.0	6.0	8.0
FACE4	3.0	8.0	7.0	7.0

(a) (b)

Figure 4.5. *(a) Original FACE1 Image. (b) Original FACE4 image.*

tively. As seen in Figure 4.9(c), some vertical stripes are visible next to the plate. This is due to the fact that the plate was not present in any of the images used for training. Artifacts in the SPIHT and EZW coders are similar. The artifacts cause blurring of the cheeks and hair in the face images. The JPEG scheme, as is well-known, results in blocking artifacts as can be seen in Figures 4.6(a) and 4.7(a).

The results indicate that adaptive Huffman coding is much faster than arithmetic coding at a marginally lower compression ratio. The proposed encoder and decoders require a wavelet transform, VQ, and 8×8 DCT for which hardware implementations already exist [18], [93], [94]. These implementations could be used to realize the proposed technique in hardware.



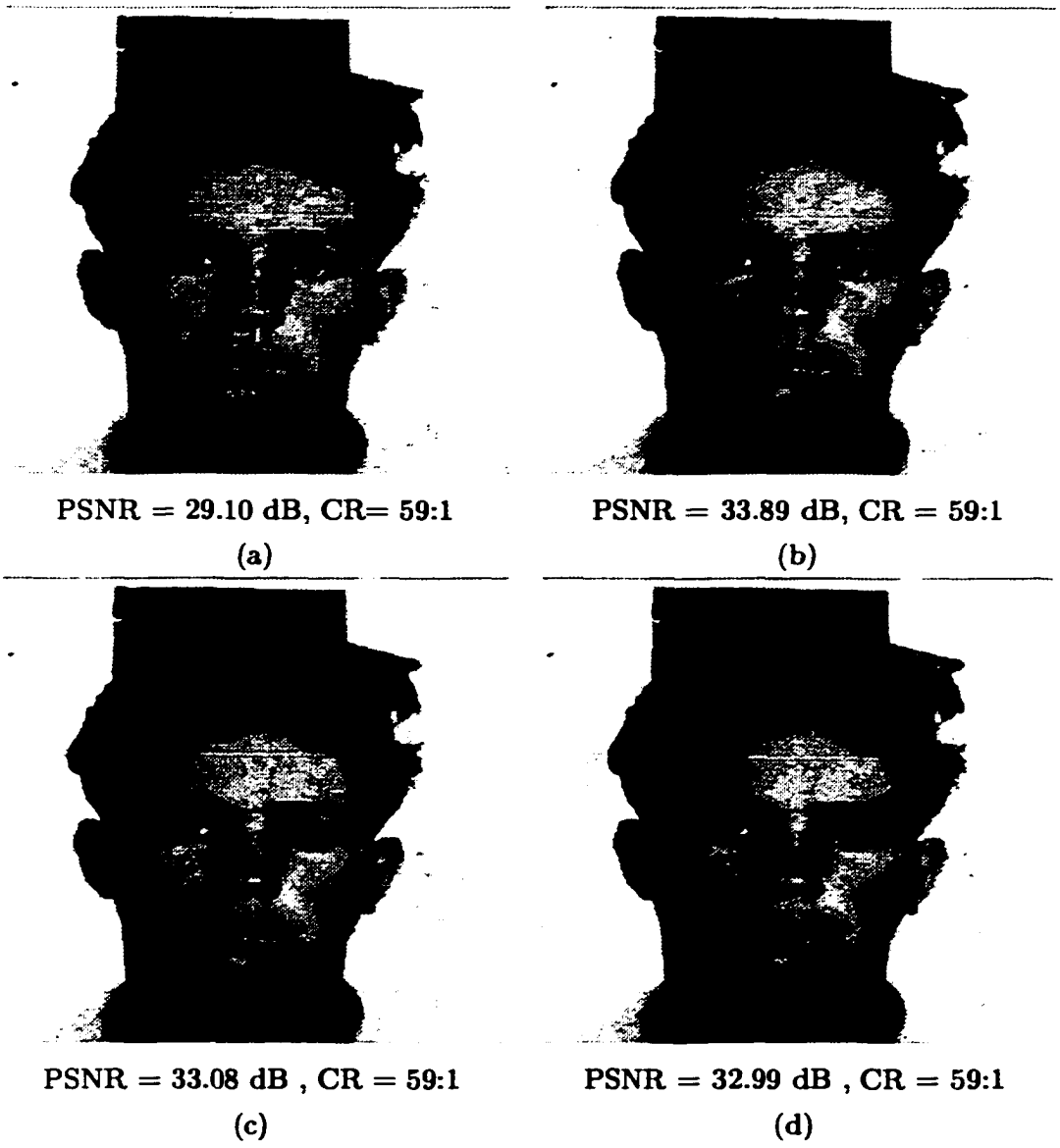
PSNR = 28.52 dB, CR= 58:1 (a)

PSNR = 33.99 dB, CR = 58:1 (b)

PSNR = 33.40 dB , CR = 58:1 (c)

PSNR = 33.07 dB , CR = 58:1 (d)

Figure 4.6. *(a) Decompressed image using the JPEG decoder. (b) Decompressed image using the SPIHT decoder. (c) Decompressed image using the MXDT decoder. (d) Decompressed image using the MXDT1 decoder.*



PSNR = 29.10 dB, CR= 59:1 (a)

PSNR = 33.89 dB, CR = 59:1 (b)

PSNR = 33.08 dB , CR = 59:1 (c)

PSNR = 32.99 dB , CR = 59:1 (d)

Figure 4.7. *(a) Decompressed image using the JPEG decoder. (b) Decompressed image using the SPIHT decoder. (c) Decompressed image using the MXDT decoder. (d) Decompressed image using the MXDT1 decoder.*

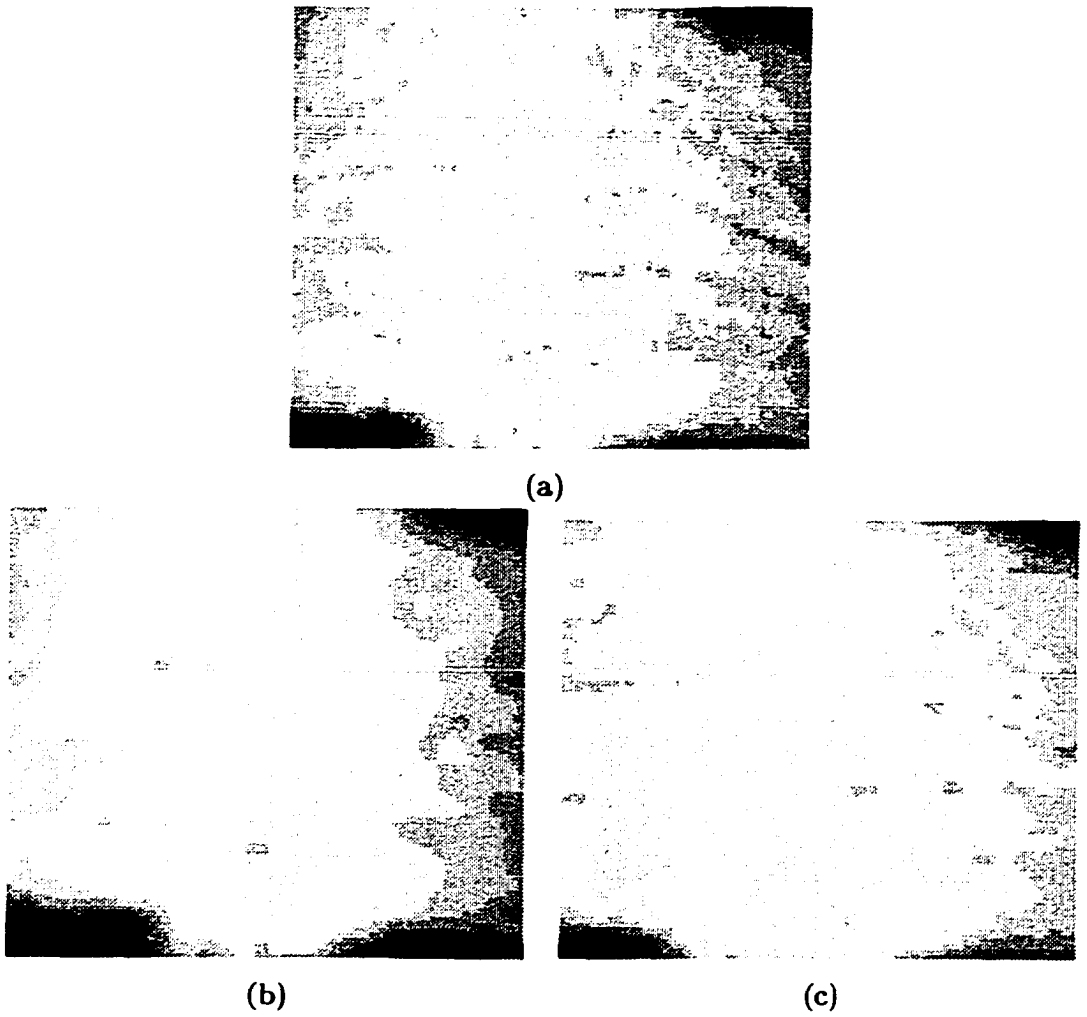


Figure 4.8. *(a) Zoomed FACE1 image. (b) Zoomed decompressed image using the SPIHT decoder. (c) Zoomed decompressed image using the MXDT1 decoder.*

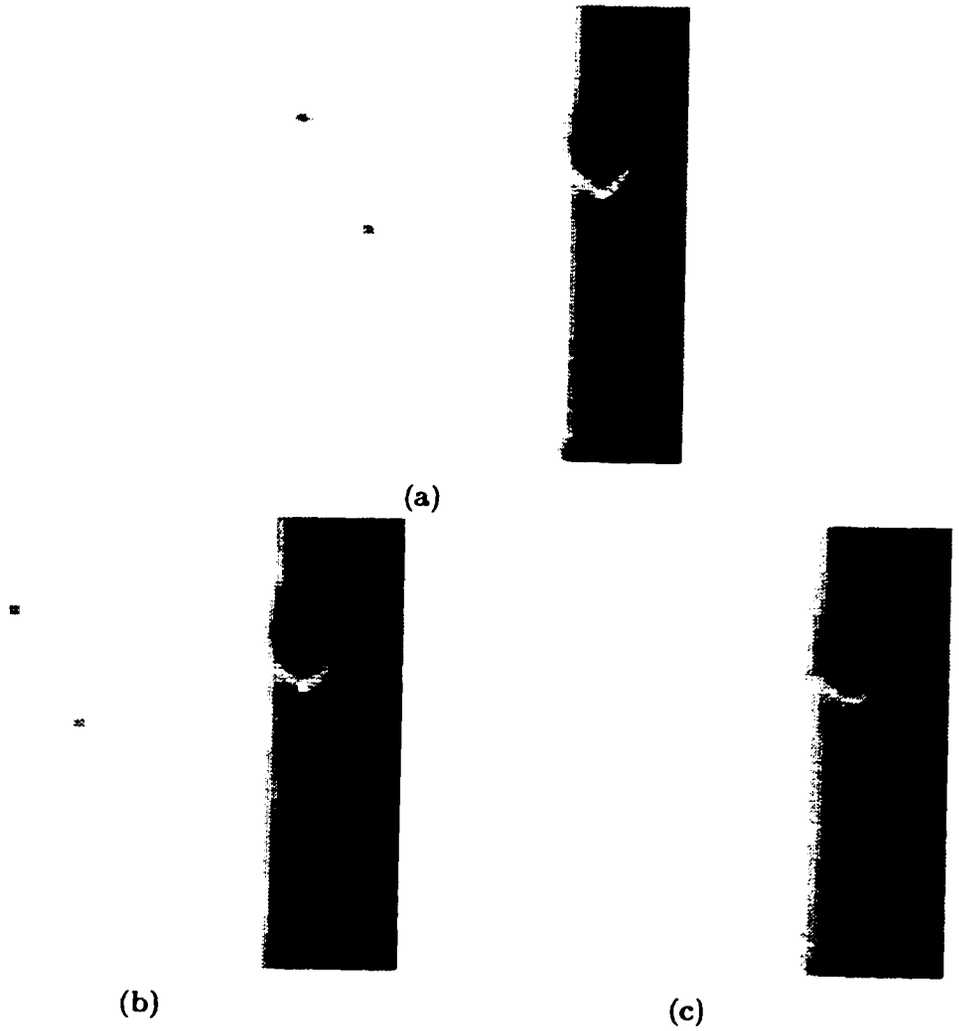


Figure 4.9. *(a) Zoomed FACE4 image. (b) Zoomed decompressed image using the SPIHT decoder. (c) Zoomed decompressed image using the MXDT1 decoder.*

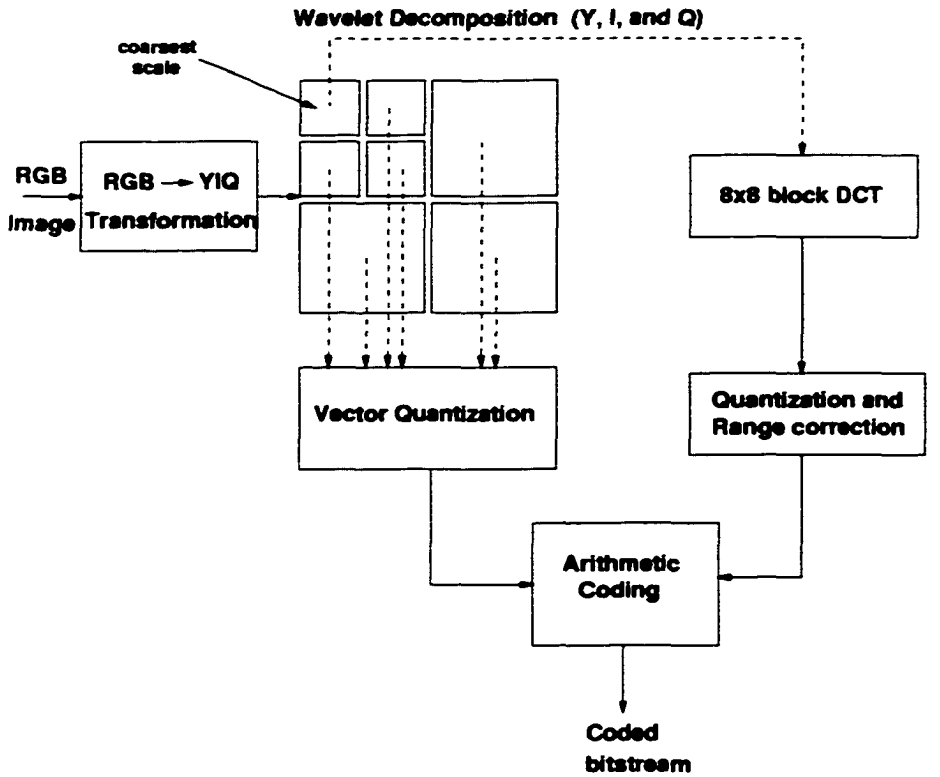


Figure 4.10. *Block diagram of the proposed wavelet-DCT mixed transform coder.*

4.7 Compression of color images

In this section, the proposed color image compression technique is described. A color image consists of three primary colors, red, green, and blue (RGB). In the NTSC color TV standard, another color model, the YIQ (luminance and chrominance) model is used for transmission. Transforming a color image to the YIQ model is an efficient way to remove the correlation among the red, green and blue color bands. Hence, the YIQ model is often used in color image compression. It can be obtained from the RGB model through the transformation

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{4.13}$$

4.7.1 Proposed color image compression technique

The block diagram for the technique, assuming a two-stage wavelet decomposition, is given in Figure 4.10.

The RGB color image is first transformed to the YIQ coordinate system using Equation 4.13. The wavelet transform based on the cascade algorithm [4] is applied to each of the Y, I, and Q components followed by vector quantization of the wavelet coefficients (except the coarsest scale). The LBG algorithm [12] is used to optimize the codebook generation. The error vectors are also quantized using the same codebook size and width. The latter step results in a significant improvement in the overall PSNR. Finally these codebook indices are encoded using arithmetic coding. A simple frequency count based probability predictor is used. No vector quantization is done on the coarsest scale because it contains almost entirely the low frequency information, and thus, the combination of DCT and scalar quantization performs better than vector quantization. An 8×8 block DCT is used on the coarsest scale followed by a scalar quantizer based on the JPEG quantization tables given in [18]. The DC coefficients of each 8×8 block are subtracted from that of the first block and the resulting DCT coefficients are finally coded using arithmetic coding.

4.7.2 Results and discussion

The proposed technique was tested using various 256×256, 24-bit color images. Most of the color images were aerial views around the coast of Vancouver island. The coder was implemented using four stages of wavelet decomposition. The technique was then compared with the existing JPEG standard. Table 4.5 shows the lengths (L) and widths (K) of the codebooks for different shapes and scales of the Y component. Shape 0, 1, 2, and 3 refer to the coarse, horizontal, vertical, and diagonal component wavelet coefficients of an image, respectively.

In most cases, Shape 3 (diagonal) has much less detail and hence needs fewer quantization levels (wider bins) as compared to Shapes 1 and 2. The values of the codebook lengths and widths given in Table 4.5 correspond to a compression ratio (CR) of 19.5 [14]. Table 4.6 compares the performance of the proposed technique with the JPEG standard for four aerial color images. The PSNR given in the table was calculated by taking the average of PSNR's for each of the R, G, and B color bands. Higher quantization of the I and Q components did not greatly affect the reconstructed image quality. In a typical example, PSNR's of 33 dB, 28 dB and 25 dB for Y, I, and Q, respectively, resulted in PSNR's of 32, 31, and 30 dB for the corresponding R, G, and B components. It should also be noted

Table 4.5. *Codebook sizes and widths for the Y (luminance component) of a typical landscape image*

Scale	Shape	Codebook	
		Length (L)	Width (K)
1	1	8	128
	2	8	128
	3	8	256
2	1	128	32
	2	128	32
	3	128	64
3	1	256	8
	2	256	8
	3	256	8
4	1	256	2
	2	256	2
	3	256	2

Table 4.6. *Comparison in compression performance between the JPEG standard and the mixed transform technique*

Image	Technique	CR	PSNR	Subjective
#1	JPEG	67:1	27.6	Marginal
	Proposed	67:1	30.7	Almost perfect
#2	JPEG	60:1	27.6	Poor
	Proposed	60:1	30.8	Good
#3	JPEG	67:1	27.1	Marginal
	Proposed	67:1	31.2	Almost perfect
#4	JPEG	68:1	29.4	Good
	Proposed	68:1	29.3	Almost perfect

Figure 4.11. *Original Image*

Figure 4.12. *JPEG compression (Compression Ratio = 67:1, PSNR = 27.6 dB).*

that in the case of Image 4 (beach image), the JPEG standard and the proposed technique gave comparable PSNR but the proposed technique had better performance in subjective evaluation. Figure 4.11 shows the original image (Image 1). Figure 4.12 shows the image compressed using JPEG standard and Figure 4.13 shows the image compressed using the proposed technique. The quality of the compressed images was also tested based on subjective evaluation of seven independent observers on a scale of 1 to 4, 4 corresponding to "almost perfect" and 1 corresponding to "poor" quality. The proposed technique gave much better results than the JPEG for very low bit rates due to the absence of any blocking artifacts. One of the major drawbacks of the proposed technique is high computational complexity because it involves optimization during the codebook generation. It was also found very hard to increase the PSNR above 33 dB for images of size less than 512×512 pixels. From Table 4.6 it can be inferred that for the same compression ratio (CR $\approx$ 65:1),

Figure 4.13. *Compression using the proposed technique (Compression Ratio = 67:1, PSNR = 30.7 dB).*

the proposed technique gave an average PSNR of 31 dB compared to 28 dB given by JPEG standard. The proposed technique performs better in subjective evaluation as well. It was noted that bigger size images yield better PSNR's at the same compression ratios. The reason is two-fold. Firstly, the vector quantizer has greater degree of freedom of selecting codebook sizes and widths in case of large size images. Secondly, the coarsest scale in a 256×256 size image has only four 8×8 DCT blocks in a four-stage wavelet decomposition. Experiments were also carried out using other image types such as facial and landscape images. Results indicate that the proposed technique gives better compression performance than the JPEG standard.

4.7.3 Conclusions

A mixed-transform technique for image compression has been proposed. The technique involves applying a wavelet transform on the input image followed by VQ on the wavelet coefficients and DCT on the scaling coefficients. An average compression ratio of 60:1 was achieved at an average PSNR of 34 dB. The results obtained with the method have been compared with results obtained with the existing JPEG standard and other wavelet coders. The proposed technique was found to perform considerably better than the JPEG standard and is comparable to other wavelet coders both in subjective evaluations and with respect to the PSNR.

A lossy color image compression technique based on wavelets and the DCT has also been proposed. The technique has been successfully applied to various color images achieving very low bit rate coding with almost no degradation in image quality. Typical results

indicate an average CR of 60-68 and PSNR of 29-32 dB. Much better results are expected if the technique is used on larger size images. The proposed technique and the existing JPEG standard have been compared and the results indicate that the proposed technique outperforms the JPEG standard in both quantitative as well as subjective evaluations.

Chapter 5

On the DSP Implementation of Wavelet Transform for Real-time Speech Compression

5.1 Introduction

Speech compression is a useful technique that facilitates the efficient use of communications channels. It has gained widespread attention due to the boom in cellular and mobile stations in limited bandwidth conditions. There have been a number of speech codecs developed for lossy coding of speech signals. Based on interdisciplinary studies of signal processing, coding theory, and psychophysics, speech signals can be effectively encoded at various transmission rates while maintaining high quality in the reconstructed signals. The performance of any speech compression algorithm is judged by the decompressed sound quality, bit-rate, computational complexity, memory requirements, and sample delay. It is difficult for an algorithm to be the best in all the above measures. Generally, for the same subjective performance, lower bit-rate coders tend to use more memory and CPU (Central processing unit) cycles than higher bit-rate coders. This is due to the increase in the computational complexity of low bit-rate coders.

For rates below 16 kbps, *analysis-by-synthesis* approach is also used. In this approach, the speech waveform is partitioned into many small segments, called speech frames, which are classified by the voiced or unvoiced speech. Voiced speech may be modeled by a pseudo-periodic sequence of impulses driving an excitation filter, an all-pole vocal tract model and a lip-radiation filter [95]. For voiced speech, code-excited linear predictive (CELP) coders model the excitation signal by selecting the 'best' waveform from an adaptive codebook containing past pitch pulses. Unvoiced speech is modeled by using random noise as an

excitation signal. The 'best' match from a codebook of excitation vectors is found using techniques such as vector quantization. Recently waveform interpolation (WI) [96] coders have been proposed which model the pitch pulse shape to encode the speech waveform. Since appropriate modeling of the excitation signal directly affects the quality of output speech, the success of the pulse coding stage depends on the closeness of the excitation signal to the true glottal signal. High-pitched speech during nasal and nasalized sounds often takes a sinusoidal form. In [97], it is shown that in addition to having large fluctuations in linear-predictor (LP) parameters, these segments are characterized by having a very low energy residual in which pitch pulses are nearly absent. To accommodate these nasal sounds, an additional term to account for smoothness in the evolution of the LP parameters, was introduced in [97]. Even though the LP filter is generally a low-order filter (order 10 or lower), as in the ITU-T G.728 16 kbits/sec speech coder [16], the search for the best match for the excitation signal from a codebook of past excitation vectors is very computationally intensive. The presence of a long-term or pitch predictor, sometimes called the adaptive codebook, is critical to coder performance at lower rates. Unfortunately, the rate required for high quality encoding of pitch filter parameters is often a large fraction of the total available coder bandwidth for low bit-rate coders [98], [99]. The application of analysis-by-synthesis techniques to pitch filter optimization can also produce high computational requirements [100].

Recently, the G.728 standard was extended further to obtain an even lower bit rate for coding of speech signals at 8 kbits/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP) also known as G.729 standard [101]. This standard proposes a technique which encodes the 16-bit linear PCM samples (sampled at 8000 Hz) to an appropriate format. There are also annexures to the G.729 standard [17], [102]-[105].

The G.729A standard [17] simplifies the computational-intensive parts of the algorithm thus reducing the complexity by approximately 20%. The G.729B standard [102] introduces the silence detector in its encoder. The G.729C standard [103] is the only floating-point implementation of the G.729 series of standards. The G.729D standard [104] brings down the bit rate to 6.4 Kb/s at the cost of degraded quality. The latest of G-series of speech coding standards is the G.729E standard [105]. It has a bit rate of 11.8Kb/s but is used for high quality speech as well as for music signals. It also preserves the background noise in the speech signals.

Subband coders have also been tested for the compression of speech signals [106]. These coders are based on selecting best and near-best bases from cosine and wavelet packet transforms. A near-best basis performs as well as the best bases in terms of the rate-

distortion measure. But these techniques also suffered in computational complexity for real-time implementation. Another variable-rate wavelet coder reported recently which incorporates both lossless and lossy speech coding is the Shorten 1.0[1] [107].

In this chapter, a wavelet transform based on pyramid decomposition [4] is used for speech coding. This decomposition results in non-uniform filter bandwidths. The wavelet filters used have the property of mapping integers to integers. In other words, integer input samples would always map to integer wavelet coefficients. This property is used to achieve lossy compression of speech all the way up to perfectly lossless mode. The wavelet transform involves no floating-point multiplications or divisions and hence is highly efficient for real-time speech compression.

In the following section various issues regarding the practical implementation of the proposed wavelet transform are considered. Next, the hardware and software used for the implementation are presented. The results based on this implementation are presented, analyzed and compared with the results obtained using the techniques based on the state-of-the-art ITU-G.728 standard [16] and Shorten-1.0.

5.2 Block wavelet transform

Generally speaking, the wavelet transform is applied on the complete input data. But situations may arise where the whole input data is not available at any particular time instant. This may be the case in real-time applications. This implies that one must resort to a 'block' wavelet transform.

Let N be the total number of samples, K be the size of each block, and L be the filter length. We need to calculate the number of wavelet coefficients that are different when using the block transform as compared to the case of an unblock transform. Also for the same level time-scale decomposition, if J is the number of levels of decomposition in the unblock wavelet transform, the number of levels needed in the block transform is also J. This really constrains the minimum size of block possible based on the number of decomposition stages required.

Theorem 1 *For a one-dimensional signal of length N, if K is the block size, L is the filter length of the highpass filter (with the Haar as the lowpass filter), then the number of wavelet coefficients that would be different in a block wavelet transform, compared to those in an unblock wavelet transform, would be $J\left\{\lfloor\frac{L-2}{4}\rfloor\left(\frac{2N}{K}-2\right)\right\}$, where J is the number of stages of wavelet decomposition and $K > 2^J$.*

[1] A newer version of this coder, namely, Shorten-2.0 also has same rate-distortion performance

Proof. Here N is the number of samples, L is the filter length of the highpass filter and K is the block size. This implies that there are N/K blocks. The wavelet coefficients between block and unblock wavelet transforms are different due to the boundary effects. There would be $2N/K$ boundaries in the block wavelet transform (two per block). On the other hand, there are only two boundaries in the case of a unblock wavelet transform (one at start and other at the end of the signal). Therefore the number of additional boundaries in the block approach are $2N/K - 2$. Each boundary would have $\lceil \frac{L-2}{4} \rceil$ wavelet coefficients that are different between block and unblock wavelet transforms. Because, this boundary effect is going to perpetrate for higher decomposition levels as well (same highpass filter), the total number of wavelet coefficients that are different in a block wavelet transform compared to an unblock wavelet transform are $J\left\{\lceil \frac{L-2}{4} \rceil \left(\frac{2N}{K} - 2\right)\right\}$.

It should also be noted that if the length of the highpass filter L equals 2, i.e., the Haar wavelet for the lowpass and highpass filters, the block and unblock wavelet transform implementations yield exactly the same values of wavelet and scaling coefficients. ■

5.2.1 Entropy coding

This section deals with the assembly programming of the adaptive order-0 Huffman coding for real-time speech compression. The program includes checking the counts for each symbol in the alphabet. The Huffman table is then built based on these numbers. The Huffman table is built using a simple, yet elegant, procedure. The individual symbols are laid out as a string of leaf nodes that are going to be connected by a binary tree. Each node has a weight, which is simply the frequency count of that symbol. The tree is then built using the following steps:

- The two nodes with the lowest weight are located.
- A parent node for these two nodes is created. It is assigned a weight equal to the sum of the two child nodes.
- The parent node is added to the list of free nodes, and the two child nodes are removed from the list.
- One of the child nodes is designated as the path taken from the parent node when decoding a 0 bit. The other is arbitrarily set to the 1 bit.
- The previous steps are repeated until only one free node is left. This free node is designated as the root of the tree.

As desired, longer length words were allocated to symbols with lower counts and shorter length codes were given to symbols with higher counts. The Huffman table is in the form

of a binary tree with each node (parent) having two branches: $child_0$ and $child_1$. The table is built based on the following structure:

- Symbol
- Symbol count
- Parent node
- $Child_0$ node
- $Child_1$ node

The symbol count is the frequency of occurrence of a particular symbol in the input file. A parent node is the node index of the node of which the present symbol/node is a child. $Child_0$ and $Child_1$ refer to the two children with decoding bit-0 and bit-1, respectively.

An example

As an example, say we need to encode the message 'AABCDAC'. The counts of symbols A, B, C, and D are 3, 1, 2 and 1 respectively.

The table looks like this:

Node	0	1	2	3
Symbol	A	B	C	D
Symbol count	3	1	2	1
Parent node	-	-	-	-
$Child_0$	-	-	-	-
$Child_1$	-	-	-	-

The least weighted nodes are 1 and 3 (corresponding to symbol B, and D. The weights are combined and put under a new node, node 4, with symbol count 2 (1+1). Parent node for B and D is hence node 4 while $Child_0$ and $Child_1$ for node 4 are node 1 and 3 respectively.

The updated table looks like this:

Node	0	1	2	3	4
Symbol	A	B	C	D	
Symbol count	3	0	2	0	2
Parent node	-	4	-	4	-
$Child_0$	-	-	-	-	3
$Child_1$	-	-	-	-	1

The above steps are repeated until the root node is reached. The final table looks like this:

Node	0	1	2	3	4	5	6
Symbol	A	B	C	D			
Symbol count	0	0	0	0	0	0	7
Parent node	6	4	5	4	5	6	-
$Child_0$	-	-	-	-	3	4	0
$Child_1$	-	-	-	-	1	2	5

Thus, the code for any symbol can be obtained by going up the tree starting from the leaf. Thus, the code for symbol A is '0', for symbol B is '101', for symbol C is '11' and D is '100'.

Thus the coded bit stream is '0010111100011' (13 bits).

Scaling of counts

There is a slight modification that was done in the code to support files of larger sizes. The values of counts for a node could overflow if the value exceeds the word limit (32 bits). To avoid overflows, the counts were scaled down by half and rounded to the nearest number, each time the 65536 samples were read. Any counts which rounded to zero were made 1 so that they still remain in the table.

Table 5.1. *CPU times (in seconds) using different software realizations*

	Implementation	Haar	TS [10]	TT [110]
DWT	C-code	0.6656	0.9625	1.259
	Optimized C-code	0.3918	0.5486	0.5728
	Assembly code	0.1626	0.2210	0.2407
IDWT	C-code	0.9504	1.0608	1.1265
	Optimized C-code	0.7453	0.8082	0.8334
	Assembly code	0.2734	0.3142	0.3245

5.3 Implementation of reversible integer arithmetic wavelet transform

The recursive pyramid algorithm was used to implement the one-dimensional discrete wavelet transform on the TMS320C30 chip [111]. Before we move to implementation issues of reversible wavelets, it is worth studying the effects of optimization and software programming techniques on the performance of the algorithm in terms of speed. This analysis is very helpful for real-time implementation.

We shall discuss the implementation of one-dimensional and two-dimensional wavelet transforms using high-level and low-level languages. Specifically, tests were run using TI's C-compiler generated executable code, the optimized compiler based executable code, and the assembly code. Pretest runs indicated that implementing in assembly language was the most feasible method for real-time implementation of wavelet transforms. The reversible wavelet transform was also extended to two dimensions.[2] It was observed that the 2D wavelet transform took almost twice the time taken by 1D wavelet transform for the same number of stages. The reason can be attributed to the transform itself. The 2D wavelet transform can be split into a 1D transform applied to the rows and columns successively. Thus, the computation time is doubled in the 2D case. Table 5.3 shows the CPU time required for a 12-stage wavelet decomposition of a set of 32768 samples using the Haar, TS, and TT wavelet transforms. Results are given for standard C code, optimized C code and assembly code for the three transforms. Figure 5.1 shows the relation between CPU time and data length for forward and reverse TT transform using the three different software realizations. The number of test samples N was 32768 and the number of stages of decomposition were fixed

[2]No real-time image compression was conducted on the DSP due to its extremely slow (16.6 million instructions per second) processor.

at 12 ($\log_2 N - 3$). The number of cycles for the 2D transform was almost twice that used for the 1D transform. Having opted for assembly programming, two versions of the 1D and 2D wavelet transform were developed, one using a block wavelet transform and the other based on an unblock wavelet transform. The algorithms were implemented with TMS320C30 DSP receiving a set of prerecorded data from a Sun workstation. This allowed a quantitative analysis of the compression performance using different wavelet bases. Further a rough estimate of the algorithm's computational complexity was obtained since the main code in this case was the same as the one for real-time implementation. A second on-line version of each algorithm was developed for the TMS320C30 DSP using the Daughter Module (DM) for input and output. The DM consists of an analog-to-digital converter (A/D) and a digital-to-analog converter (D/A). Details can be found in Section 5.4.1. These versions differed from the off-line versions by an additional set of instructions needed to achieve data format compatibility between the DM and TMS chip. With these on-line versions, the maximum sampling frequency feasible for a specific wavelet set was determined.

The nonlinear integer arithmetic based TT transforms proposed in Chapter 3 was implemented on the DSP chip. It should be noted that the C30 chip is a 32-bit floating point processor and hence the integer arithmetic based transforms did not really save the computation time for both integer and floating point point instruction took one instruction cycle for completion. The wavelet decomposition of the input speech samples is a non-uniform subband decomposition. A four-stage wavelet decomposition was carried out for lossy speech compression while a six-stage wavelet decomposition (pyramid scheme) was used for lossless speech compression. The reason for doing this was that for lossless coding, for a block size of 4096 samples, six stage of wavelet decomposition removed most of the correlation among the samples. For lossy coding, decomposing more than 4 stages would imply splitting the range of frequencies from zero to $F_s/32$ (F_s is the sampling frequency). For $F_s = 8$ KHz, this would imply a range of zero to 250 Hz. It has been observed that the human ear cannot recognize frequencies less than about 200 Hz. Hence, there is no need of applying more than 4 stages of decomposition. For lossy compression, the quantization levels used are given in Table 5.2. The finest scale was completely ignored due to the fact that all the formants in a human speech are generally at frequencies below 2 KHz.

Formants in human speech are mostly between 250 Hz and 1800 Hz. These frequencies lie in bands 2 and 3, respectively. Hence, these bands were least quantized. Bands 0 and 1 contain formants related to nasal and nasalized sounds which have their first formant around 250 Hz.

This selection of the quantization levels gave sub-optimal results for a series of test

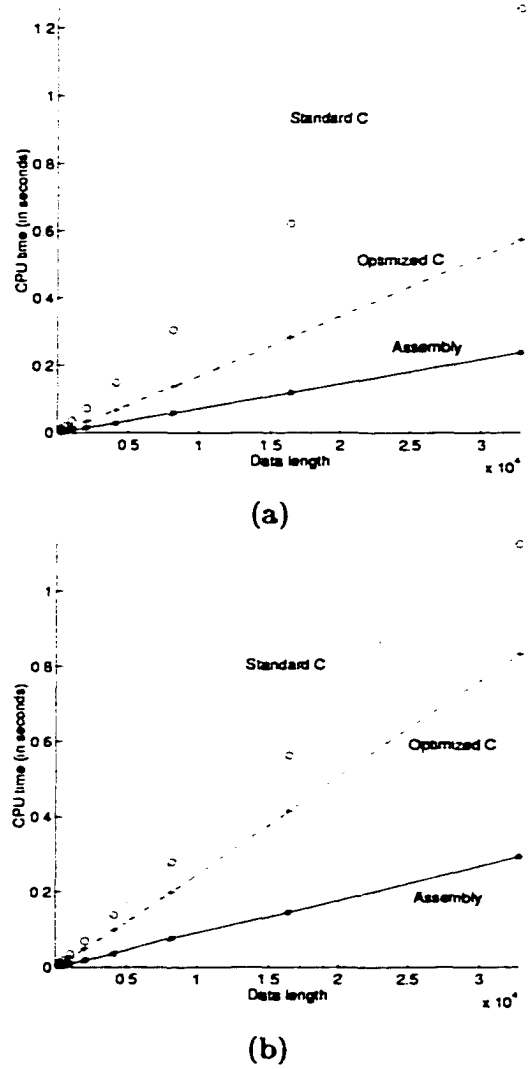


Figure 5.1. *(a) Relation between CPU time and data length for DWT based on TT transform with different software implementations(b)Relation between CPU time and data length for IDWT based on TT transform with different software implementations.*

Table 5.2. *Quantization levels for different subbands*

Band number	Frequency range	Number of quantization levels
0	[0, 250)	75
1	[250, 500)	75
2	[500, 1000)	100
3	[1000, 2000)	100
4	[2000, 4000)	0

sequences. The results are discussed in Section 5.5.

5.4 Implementation issues for the TMS320C30 DSP

An important factor in optimizing any assembly code is the detailed knowledge of the architecture of the DSP, both from a hardware and software perspective, and equally the detailed knowledge of the algorithm to be implemented.

5.4.1 Hardware description

The wavelet transform was implemented on the TMS320C30 DSP by Texas Instruments Inc. (TI) on the SDSP/C30D board developed by Loughborough Sound Images Inc. (LSI) with added analog-to-digital (A/D) and digital-to-analog (D/A) capabilities. The TMS320C30 is a 32-bit processor with an optimized architecture for computation intensive signal processing and mathematical operations using floating-point arithmetic. The 33.3 Mhz version used in this work has a 60 nanosecond instruction cycle and is capable of doing 16.7 MIPS (million instructions per second). By exploiting the full use of parallel instructions, it is theoretically possible to achieve 33.3 MFLOPS (million floating-point operations per second), i.e., one instruction every clock cycle. Along with the optimized integer and floating-point units, the DSP integrates an on-chip 4K×32-bit ROM, a 2K×32-bit on-chip RAM, two timers, a DMA controller (dual memory access), and serial input/output interfaces. The TMS320C30 uses a 32-bit data bus and a 24-bit address bus. The DMA controller can perform very fast memory access without interrupting the CPU (central processing unit).

The SDSP/C30D SBus board [112], [113] (also referred to as the DSP motherboard as opposed to the daughter module which includes the A/D and D/A converters) is a single

width SBus board, 83.8 mm×146.7 mm, connected to the SBus slot, and designed for use with SPARC/SUNOS 4.1.1 (or greater) compliant systems. The DSP board has two memory banks of zero wait states each 64K×32-bit in size. Bank 0 is used to set up reset and interrupt routines plus the code for the LSI's auxiliary library, the debugger, and the TMS320C30 executable object code. Bank 1 is mostly used for data manipulations and storage. In addition there is a block of 2K×32-bit of dual-port RAM (DPRAM) to which the host machine and the TMS processor has concurrent access capabilities (of course unless both access the same memory location). This is not true in case of SRAM where the DSP must be held until the data fetching by the host machine is complete. Finally, for greater modularity, the DSP motherboard also allows communication to other boards or devices through very efficient serial and parallel interfaces.

The daughter module [114] presents two 16-bit dual input/output (I/O) channels with a maximum input sampling rate of 200 KHz and a maximum output sampling rate of 500 KHz. Both the I/O channels are factory fitted with lowpass filters consisting of two cascaded second order, conventional Salley-Key configurations with unity dc gain. Identical resistor values give a maximally-flat Butterworth response with -24 dB/octave roll-off in the stop-band. The use of the A/D and D/A capabilities of the DM is simply a matter of setting up a number of registers, defining an interrupt routine, enabling DSP interrupts, and reading or writing the signal samples. Other characteristics of the DM include inter-channel isolation of 85 dB, input impedance of 20 kΩ, and output impedance of 0.5 Ω, being capable of driving a load down to 600 Ω to a voltage range of ∓3 V. A 16-bit A/D converter with maximum input voltage of 3 V results in an accuracy of $3/2^{15}$ V.

5.4.2 Implementing the block wavelet transform

A cross-assembly code was written by implementing the wavelet transform in C (and optimizing the generated assembly code) whereas the Huffman coder was implemented in assembly code and called in the main C code using cross-assembly features of the TMS320 support software. The wavelet transform module was implemented in C and the generated assembly code was optimized by removing some address load and branch instructions. Also block repeat and single instruction repeat (RPTS and RPTB) instructions were used wherever feasible. The wavelet transform was implemented for a variable number of stages of decomposition for the lossless mode and a fixed number of stages for the lossy mode. The block size was also a variable and could be varied between 512 and 16384 samples. Generally, large block sizes are better in terms of compression but require more memory.

This is one of the major differences between wavelet based methods and linear-prediction based methods. Attention should be given to the boundaries of the block. In most of the window-based voice coders (vocoders) such as the linear-predictive coders, some samples from the previous block are attached to the boundaries of the current block to avoid edge discontinuities. In the proposed technique, boundary artifacts are reduced by doing a symmetric extension at the block boundaries. Initially the program was written to test various reversible wavelet transforms by entering the length and coefficients of the impulse response of the lowpass and highpass filters. Later it was found out that this is creating overhead with the effect of reducing speed. So the program was changed so as to handle only a particular set of wavelet filters. At some points RPTB instructions were also avoided by repeating the code many times. This increased memory required by the program but removed the overhead of looping.

5.4.3 Implementation of the entropy coder

As was described the previous section, an adaptive order-0 Huffman coder was implemented in assembly code on the TMS320C30 DSP. The coder was based on two passes, the first one to determine the frequency of occurrence of each symbol and build the Huffman table and the second pass to actually generate the coded bitstream corresponding to the input symbol data. The coder was implemented in assembly code using the same memory bank of 64 K words. This avoided the need to reload the data page pointer (DP). The program memory was loaded in the external bank 0 SRAM while the data was loaded in the external bank 1 SRAM. Both are single wait-state memory banks. The Huffman table was stored using a data structure having symbol count, parent, and its descendants as the members. Each structure corresponds to a node in the Huffman table. The counts of the existing symbols was sent to the decoder in the header for each block of data. The Huffman decoder reads the counts and builds the Huffman table for that block. A second pass of data decodes the coded bitstream. A frequently used assembly instruction was the delayed-branch (BRD) instruction. This instruction is like the branch instruction except that it preserves the pipeline operation of the DSP.

5.4.4 Real-time implementation

Real-time implementation has some critical issues to be taken care of compared to the off-line implementation. First of all, one needs two input and two output buffers. At the input, the input data is loaded in one of the buffers while the DSP is operating on the other buffer.

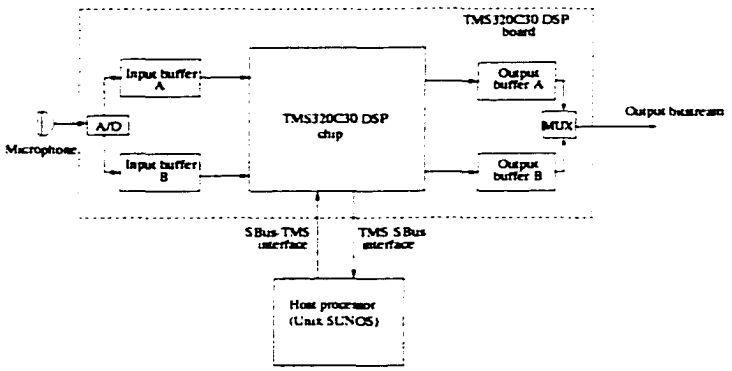


Figure 5.2. *Block diagram of the real-time implementation of a wavelet-based speech coder on the TMS320C30 DSP*

The same explanation holds for using two buffers at the output. The block diagram of the complete setup is given in Figure 5.2. The 16-bit A/D converter converts a ± 3 V analog input range to a 16-bit wide digital sample. But unfortunately the proposed method cannot support 16 bits per sample due to a much increased memory requirement to store the count of various possible symbols in the 2^{16} size alphabet. Hence only 8 MSB's were read into the TMS 64K-words SRAM bank 1. The bitstream was packed with a header for each block and sent on the channel in synchronization with the input buffers.

In other words, when the input data is being loaded into input buffer A, the bitstream is being pumped out from the output buffer A. The SUN workstation (host machine) was used to provide user interaction such as setting sampling frequency, block size, loading the object file on the TMS, starting the TMS, halting the TMS etc.

5.5 Results and discussion

The performance of the proposed coder was compared with that of two existing speech coders. The first one is the unified lossless/lossy speech coder using windowing and prediction techniques based on [107]. The lossless coder used a block prediction followed by a static Huffman coder.[3] The compression results were compared in terms of peak-signal-to-noise ratio (PSNR) and the total compressed file size. The second coder with which the performance was compared is based on the well-known G.728 standard[1] [109]. The standard though developed for 16 bits/sample speech data is also valid for the 8 bit/sample speech

[3]The software was downloaded from "http://www.ee.duke.edu/ bp/speech-software.html".

Table 5.3. *Compressed file size for the proposed method and the coder proposed in [107]*

Signal	Shorten-1.0 [107]	Proposed
Signal 1	7593	7308 (1.12 s)
Signal 2	9745	9584 (1.03 s)
Signal 3	9093	8970 (0.74 s)
Signal 4	10981	8985 (1.45 s)

data. The output bit-rate of the G.728 standard-based coder is 16 kbits/sec. This implies that for an 8 bit/sample μ-law signal [115], there is a compression ratio of 4:1. It was also observed that for an 8 bit/sample speech, the compression ratio was only 2.5:1.

Four different test speech signals of 16384 samples each were tested on the DSP. These four samples are:

- *what did you tell me*, a male voice with British accent.
- *my name is inder*, a male voice with Indian accent.
- *beijing tiananmen*, a male voice with Chinese accent.
- *the lawyers broke that will*, a female voice with American accent.

Of these four test signals, signals two and three had noise in the background while signals 1 and 4 were clear. The reason for having background noise was to compare how well the proposed method performs with respect to preserving the background noise in the signal. Table 5.3 shows the lossless compression results for the proposed technique. The results are also compared with the lossless coder in [107]. As can be seen, both the methods have variable output bit-rate because both use entropy coding based on variable length codewords. The computation times for the proposed technique is also given in the parenthesis. The sampling rate was fixed at 8 KHz. For a sampling frequency of 8 KHz, 16384 samples would require 2 seconds. From the table above, it can be inferred that only approximately 70% of the time was used in the worst case (Signal 4). The Shorten-1.0 algorithm was not implemented in real-time and thus no computation time is given here. The same is the case for G.728 algorithm as well.

Table 5.4 shows the results for lossy compression of speech signals using the proposed technique. The quantization levels were based on the allocation given in Table 5.2. Results are also compared with results obtained for the Shorten-1.0 and the G.728 LD-CELP vocoders. The computation time for the proposed technique is given in the parenthesis. The computation time for lossy compression is approximately 50% lower than the time required

Table 5.4. *Compressed file size for various vocoders. The computation times for the proposed technique are given in the parenthesis*

Signal	Shorten-1.0 [107]	G.728 [109]	Proposed
Signal 1	7593 bytes lossless	6554 bytes 36.1dB	4600 bytes 46.0dB (0.50 secs.)
Signal 2	5821 bytes 41.9dB	6554 bytes 31.7 dB	5370 bytes 40.0dB (0.52 secs.)
Signal 3	7161 bytes 49.7dB	6554 bytes 37.0dB	5100 bytes 41.0 dB (0.50 secs.)
Signal 4	7037 bytes 25.9dB	6554 bytes 36.0dB	4858 bytes 35.2dB (0.54 secs.)

for lossless compression, the reason being that lossy compression involves quantization of subband coefficients and thus the time to build the Huffman table is greatly reduced. Moreover the frequency band [2000, 4000) is not coded at all. As can be seen, the G.728 vocoder has a fixed file size of 6554 bytes. It should be mentioned that the output bit rate in the Shorten-1.0 technique can be varied between 3 bits/sample and 16 bits/sample. For test Signal 1, Shorten-1.0 did not do lossy compression even at 3 bits/sample. Shorten-1.0 could not provide higher compression on Signals 2, 3, and 4 than the one given in Table 5.4. This is because it can compress to a maximum of 3 bits per sample. It was difficult to do fair comparison from the coding results given in Table 5.4 since neither the rate nor the distortion is fixed across all the coding methods. Thus, subjective results were also carried out on the decompressed samples. It was observed that for the LD-CELP coder the decompressed sound signals had a noise in the background which was objectionable to the listener. However a post-filter could have improved its subjective quality. Shorten-1.0 and the proposed technique in general gave comparable subjective results. Both the proposed method and the Shorten-1.0 could preserve background noise in Signals 2 and 3, but Shorten-1.0 had a lower compression ratio than the proposed technique. Also, as can be seen from Table 5.4, the proposed method took an average of 0.5 s to process a block of 16384 samples. This implies that to process 8000 samples (at 8 KHz), it would take 0.24 seconds. This results

in an average MIPS rate of 4 MIPS for a 16.6 MIPS TMS320C30DSP. Thus the overall MIPS rate of both the encoder and decoder would be approximately 8 MIPS. This is because the encoder and decoder would be very similar (in the worst case). Compare that to the 17.5 MIPS (10.7 MIPS for encoder and 6.8 MIPS for decoder) required by the G.728 vocoder implemented by DSPSE Corporation on the TMS320C30 DSP.[4] The approximate sample delay is equal to length of block used in block wavelet transform (BWT). For a block size of 512 samples, this corresponds to a delay of 64 ms which is a little on the high side compared to for the delay of 40ms for the G.728 codec.

5.6 Conclusions

A wavelet-based speech coder has been proposed. Real-time implementation on the TMS320C30 DSP has been successfully carried out. The speech signals were sampled at 8 KHz at 8 bits/sample. The proposed vocoder was compared with the G.728 standard based LD-CELP vocoder and the Shorten-1.0 vocoder. Results indicate that the proposed vocoder performed best in terms of PSNR and was comparable with the Shorten-1.0 in subjective tests. Future work should include extending the proposed technique to deal with 16 bits/sample input speech signals. For 16 bit/sample data, it is expected that the block-size would have to be decreased. The MIPS rate required by the proposed technique was nearly half that of the G.728 on the TMS320C30DSP. The sample delay, on the other hand, was 64 ms in the proposed method compared to 40 ms in the G.728 standard. An adaptive technique to encode the symbols would be more suitable than a static coding technique due to the overhead involved in storing the probability of each symbol in a static coding technique. An adaptive model to choose the number of quantization levels for each subbands is also warranted.

[4]Technical details can be obtained from the company's web site at
http://www.dspse.com/programs/software/va/itu/products/g728%5Fc3x.htm

Chapter 6

Conclusions and Scope for Future Work

6.1 Overview

Applications of wavelets to compression have been considered. The first part of the thesis is focused on the use of reversible wavelets for lossless image compression. The second part applies wavelets for the compression of image databases while the last part is focussed on applying wavelets for lossless and lossy speech compression on the TMS320C30 digital signal processor. In accordance with this logical classification, the contributions of the thesis can be summarized in the three sections that follow.

6.2 Reversible wavelets for lossless image compression

Reversible wavelets have been used for lossless compression of images. Generally smooth images can be efficiently compressed using longer filters than shorter ones. Before the lifting scheme was developed, there was no structured procedure for designing reversible wavelets. The only two reversible wavelets that existed were the Haar wavelet and the two-six wavelet. We introduced the two-ten wavelet transform which we have used to compress medical images such as MRI and CT images. The results indicate that due to the smooth nature of the images, the two-ten transform gave better results than the two-six transform. It was also shown that increasing the length of filters beyond ten for the highpass filters degraded the performance of the transform. The reason for this is attributed to the fact that the regularity of the transform increases if the filter length is increased beyond 10. It was observed that the increase in filter length led to an increase in ringing at the edges of the image.

Reversible wavelets were also used for the compression of color images. Applying

wavelets to the RGB color bands resulted in poor compression due to the redundancy between spectral bands. To overcome this problem, a reversible color image transformation was proposed which removed the redundancy between the RGB bands at the expense of a small increase in computational complexity. The transformation requires only a few additions and shift operations but no multiplications. The compression performance was better than that achieved with the JPEG standard in subjective and objective evaluations.

6.3 Lossy image compression

Wavelets combined with vector quantization were used for the compression of gray-scale images. The biorthogonal MIT97 wavelet transform was used for wavelet decomposition due to its overall superior performance. Vector quantization was applied on the wavelet coefficients and the residual error to obtain lossy compression. The codewords were compressed using arithmetic coding. The scaling coefficients were not vector quantized due to the higher information content. Instead the DCT followed by JPEG scalar quantization was applied to quantize the scaling coefficients. At the decoder, the steps were reversed to obtain the decompressed image. An enhanced decoder was also proposed which would apply thresholding on the reconstructed scaling coefficients in order to eliminate any artifacts due to scalar quantization. This approach improved the subjective quality of the reconstructed images even though there was a slight degradation in the PSNR results. The results were also compared with results obtained with state-of-the-art embedded coders such as the EZW and SPIHT coders and were found to be comparable. In terms of complexity, the encoder required more computations than the decoder. However, due to the simplicity of the decoder, the proposed technique can be used for fast image retrieval, image browsing applications on the Internet and other applications. The embedded techniques have the same degree of complexity at the encoders and decoders. The proposed technique is more robust in terms of channel and bit errors compared to other state-of-the-art techniques.

6.4 Real-time compression of speech

Reversible transforms were used for real-time compression of speech signals. Both lossless and lossy compression of speech were conducted and implemented on the TMS320C30 DSP. The reversible two-ten transform was used for wavelet decomposition. The wavelet coefficients were encoded using Huffman coding. Some issues related to real-time implementation were also studied. Due to the real-time nature of the signal, the samples were stored in

double buffers which were switched at a particular rate depending on the size of the buffers. It was also observed that longer filters generally resulted in better compression ratio at the cost of an increased output sample delay. Reducing the buffer size would increase the overhead in Huffman coding. An approach for lossy compression of speech was also proposed. The results were compared with results obtained with the state-of-the-art G.728 16Kbits/sec coding standard as well as the Shorten-1.0 unified lossless and lossy speech compression technique. The results indicate that the proposed technique offers better compression performance than the G.728 and Shorten-1.0 technique in subjective and objective results. The proposed technique also preserves the background noise in speech unlike the G.728 standard.

6.5 Future research

There are still many areas that could benefit from further research. Some of these areas include the following:

- Although the reversible two-ten transform has been successfully applied for lossless compression of medical images, the performance can be improved by making the transforms adaptive. In other words, it is highly likely that improved compression performance can be achieved by segmenting the image into homogeneous sections and then applying possibly different transforms to each subregion. For such an approach, fast and effective segmentation algorithms need to be devised. The selection of a particular transform for a particular type of subregion would also need investigation.
- Yet another possibility for improving transform effectiveness is to develop spatially varying transforms. In this case, the transform adapts to the spatially varying statistics of the image. As a starting point, one might consider some recent work by Clay-Poole et al. [44] which proposes adaptive wavelet transforms for improved compression performance.
- Adaptive arithmetic coding of wavelet coefficients was used for lossless compression of images. One would be able to improve compression performance by using an embedded zerotree coding technique at the expense of increased computational complexity.
- Full search VQ was used for lossy compression of gray scale images. It would be worthwhile to compare the performance of full search VQ with tree-structured VQ and lattice quantizers.
- Although this thesis has dealt with lossy compression of gray scale images, one could extend the approach to compression of color images as well.

- The speech coding algorithm uses a uniform scalar quantizer for lossy compression of speech signals. A quantization technique based on perceptual weighting of speech signals could further enhance the subjective performance of the speech coder.
- Finally, most research effort to date has been focussed on separable transforms. Although such transforms are computationally more efficient than their non-separable counterparts, they have a disadvantage of severe directional dependencies. By using non-separable transforms, these dependencies can be reduced. This possibility should be explored.

Bibliography

[1] A. Haar, "Zur theorie der orthogonalen funktionen-systeme," *Math. Annal.*, vol. 69, pp. 331-371, 1910.

[2] A. Grossman and J. Morlet, "Decomposition of hardy functions into square integrable wavelets of constant shape," *SIAM Journal on Mathematical Analysis*, vol. 15, pp. 723-736, July 1984.

[3] I. Daubechies, *Ten Lectures on Wavelets*, SIAM, 1992.

[4] S. Mallat, "A theory for multiresolution signal decomposition; the wavelet representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pp. 674-693, July 1989.

[5] A. Cohen, I. Daubechies, and J. C. Feauveau, "Birothogonal Bases of Compactly Supported Wavelets," Tech. Rep., TM11217-900529-07, AT&T Bell Labs.

[6] M. Antonini, M. Barlaud, and P. Mathieu, "Image coding using lattice vector quantization of wavelet coefficients," *Proc. IEEE Conf. ASSP*, vol. 4, pp. 2273-2276, May 1991.

[7] T. Senoo, and B. Girod,"Vector quantization for entropy coding of image subbands," *IEEE Trans. Image Processing*, vol. 1, pp. 526-533, Oct. 1992.

[8] M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding*, Prentice-Hall, Englewood Cliffs, New Jersey, 1995.

[9] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing*, vol. 41, pp. 3445-3462, Dec. 1993.

[10] A. Zandi, J. D. Allen, E. L. Schwartz, and M. Boliek, "Compression with integer arithmetic embedded wavelets," *Data Compression Conference*, Snowbird, Utah, pp. 212-221, 1995.

[11] A. Said and W. A. Pearlman, "An image multiresolution representation for lossless and lossy compression," *IEEE Trans. Image Processing*, vol. 5, pp. 1303-1310, Sept. 1996.

[12] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. 28, pp. 84-95, 1980.

[13] A. Antonini, M. Barlaud, O. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. Image Processing*, vol. 1, pp. 205-220, April 1992.

[14] A. Averbuch, D. Lazar, and M. Israeli, "Image compression using wavelet transform and multiresolution decomposition," *IEEE Trans. Image Processing*, vol. 5, pp. 4-15, 1996.

[15] K.-C. Liang, J. Li, and C.-C. J. Kuo, "Embedded wavelet coder with multistage vector quantization," *Int. J. Imaging Systems Tech.*, vol. 8, pp. 444-449, 1997.

[16] Coding of speech at 16Kbits/sec using low-delay code excited linear predictor, *G.728 ITU-T Recommendation*, Sept. 1992.

[17] Reduced complexity 8Kbits/sec CS-ACELP speech codec, *G.729A ITU-T Recommendation*, Nov. 1996.

[18] W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*, Van Rostrand Reinhold, 1993.

[19] A. N. Netravali and B. G. Haskell, *Digital Pictures*, New York: Plenum Press, 1988.

[20] International Telegraph and Telephone Consultative Commitee (CCITT, now ITU), *Progressive Bilevel Image Compression*, ITU-T.82, (JBIG), 1993.

[21] W. B. Michael and A. Ramaswammy, "An efficient representation of non-stationary signals using mixed transforms with applications to speech," *IEEE Trans. Circuits and Systems*, vol. 42, pp. 393-401, June 1995.

[22] Y. W. Nijim, S. D. Stearns, and W. B. Mikhael, "Lossless compression of seismic signals using differentiation," *IEEE Trans. Geoscience Remote Sensing*, vol. 34, no. 1, pp. 52-56, Jan. 1996.

[23] M. Vetterli and C. Herley, "Wavelets and filter banks: Theory and design," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 40, no. 9, pp. 2207-2232, Sept. 1992.

[24] S. G. Mallat, "Multiresolution approximations and wavelet orthonormal bases of $L^2(\mathbf{RR})$," *Trans. Amer. Math. Soc.*, vol. 315, no. 1, pp. 69-87, 1989.

[25] E. Kreyszig, *Introductory Functional Analysis with Applications*, John Wiley and Sons, 1989.

[26] G. Strang , "Wavelets and dilation equations: a breif introduction," *SIAM Review*, vol. 31, no. 4, pp. 614-627, Dec. 1989.

[27] G. Strang and G. Fix, "Fourier analysis of the finite element method in Ritz-Galerkin theory," *Stud. Appl. Mathematics*, vol. 48, pp. 265-273, 1969.

[28] J. Kovacevic and M. Vetterli, "Nonseparable multidimensional perfect reconstruction banks and wavelet for $\mathcal{R}_n$," *IEEE Trans. Inform. Theory*, vol. 38, pp. 533-555, Mar. 1992.

[29] P. P. Vaidyanathan, "Multirate digital filters, filter banks, polyphse networks, and applications: A tutorial," *Proceedings of the IEEE*, vol. 78, pp. 56-93, Jan. 1990.

[30] J. Katto, and Y. Yasuda, "Performance evaluation of subband coding and optimization of its filter coefficients," *SPIE Visual Commun. Image Processing*, vol. 1605, pp. 95-106, Boston, Massachusetts, Nov. 1991.

[31] H. S. Malvar, "Lapped transforms for efficient transform/subband coding," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol 38, pp. 969-978, June 1990.

[32] J. D. Villasenor, B. Belzer, and J. Liao, "Filter evaluation and selection in wavelet image compression," *Proc. IEEE Data Compression Conf.*, Snowbird, Utah, pp. 351-360, 1994.

[33] J. D. Villasenor, B. Belzer, and J. Liao, "Wavelet filter evaluation for image compression," *IEEE Trans. Image Processing*, vol. 4, pp. 1053-1060, Aug. 1995.

[34] I. Balasingham, T. A. Ramstad, and J. M. Lervik, "Survey of odd and even length filters in tree structured filter banks for subband image compression," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. 4, pp. 3073-3076, 1997.

[35] J. Bradley, C. Brislawn, and T. Hopper, *The FBI Wavelet/scalar Quantization Standard for Gray-scale Fingerprint Image Compression*, Tech. Report LA-UR-93-1659, Los Alamos National Laboratories, Los Alamos, New Mexico, 1993.

[36] M. J. T. Smith and T. P. Barnwell, "Exact reconstruction for tree structured subband decoders," *IEEE Trans. Acoust. Speech Signal Processing*, vol. 34, pp. 434-441, June 1986.

[37] J. P. Andrew, P. O. Ogunbona, and F. J. Paoloni, "Comparison of wavelet filters and subband analysis structure for still image compression," *Proc. IEEE Intern. Conf. Acoustics, Speech, and Signal Processing*, vol. 5, pp. 589-592, 1994.

[38] S. Masud and J. V. Mc Canny, "Finding a suitable wavelet for image compression applications," *Proc. IEEE Intern. Conf. Acoustics, Speech, and Signal Processing*, vol. 5, pp. 2581-2584, 1998.

[39] G. Karlsson and M. Vetterli, "Extensions of finite length signals for subband coding," *Signal Processing*, vol. 17, pp. 161-168, June 1989.

[40] K. Nishikawa, H. Kiya, and M. Sagawa, "Property of circular convolution for subband image coding," *Proc. IEEE Intern. Conf. Acoustics, Speech, and Signal Processing*, vol. 4, pp. 281-284, 1992.

[41] S. A. Martucci and R. M. Mersereau, "The symmetric convolution approach to the nonexpansive implementation of FIR filter banks for images," *Proc. IEEE Intern. Conf. Acoustics, Speech, and Signal Processing*, vol. 5, Minneapolis, MN, pp. 65-68, April 1993.

[42] P. Lux, "A novel set of closed orthogonal functions for picture coding," *Archiv fur Elektronik und Uebertragungstechnik*, vol. 31, pp. 267-274, July 1977.

[43] W. Sweldens, "The lifting scheme: A custom-design construction of biorthogonal wavelets," *Applied and Computationa Harmonic Analysis*, vol. 3, no. 2, pp. 186-200, 1996.

[44] R. L. Claypoole, Jr. and R. G. Baraniuk, "Adaptive wavelet transforms via lifting,"

Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing, vol 3, pp. 1513-1516, 1998.

[45] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *Tech. Report*, Bell Laboratories, 1996.

[46] G. Strang and T. Nguyen, *Wavelets and Filter Banks*, Wellesley-Cambridge Press, 1996.

[47] A. R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, "Wavelet transforms that map integers to integers," *Tech. Report*, Department of Mathematics, Princeton University, August 1996. Available from http://cm.bell-labs.com/who/wim/papers/integer.ps.

[48] I. Balasingham, J. M. Lervik, and T. A. Ramstad, "Lossless image compression using integer coefficient filter banks and class-wise arithmetic coding," *Proc. of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, vol 3, pp 1349-1352, 1998.

[49] M. D. Adams and F. Kossentini, "Performance Evaluation of different reversible decorrelating transforms in the JPEG-2000 baseline system," *Proc. of 1998 IEEE Sym. Advances in Digital Filtering and Signal Processing*, Victoria, BC, Canada, pp. 20-24, June 1998.

[50] M. D. Adams, *Reversible wavelet transforms and their applications to embedded image compression*, M.A.Sc thesis, Department of Electrical and Computer Engineering, University of Victoria, Victoria, Canada, January 1998. Available from *http://www.ece.ubc.ca/ mdadams/papers/mascthesis.ps.Z.*

[51] R. L. Claypoole, Jr., G. Davis, W. Sweldens, and R. G. Baraniuk, "Nonlinear wavelet transforms for image coding," *Proc. of 31st Asilomar Conf.*, Pacific Grove, CA, 1997.

[52] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, Boston, MA, 1992.

[53] N. S. Jayant and P. Noll, *Digital Coding of Waveforms*, Prentice-Hall, Englewood-Cliffs, NJ, 1984.

[54] R. M. Gray, "Vector Quantization," *IEEE Acoustics, Speech, and Signal Processing Magazine*, vol. 1, pp. 4-29, April 1984.

[55] N. Chaddha, P. A. Chou, and T. H. Y. Meng, "Scalable compression based on tree structured vector quantization of perceptually weighted block, lapped, and wavelet transforms," *Proc. of IEEE Int. Conf. on Image Processing*, Washington DC, USA, pp. 89-92, Oct. 1995.

[56] H. Jafarkhani and N. Farvardin, "Scacalble wavelet image coding scheme using multistage pruned tree-structured vector quantization," *Proc. of the IEEE Int. Conf. on Image Processing*, Washington DC, USA, pp. 81-84, Oct. 1995.

[57] D. Mukherjee and S. K. Mitra, "Vector set partitioning with classified successive refinement VQ for embedded wavelet image and video coding," *Proc. of IEEE Int. Conf.*

on Acoustics, Speech, and Signal Processing, Seattle, WA, vol. 5, pp. 2809-2812, May 1998.

[58] I. Moccagatta and M. Kunt, "An image coding scheme based on perceptually classified VQ for high compression ratios," *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, vol. 5, pp. 2487-2491, 1995.

[59] I. Moccagatta and M. Kunt, "VQ and cross-band prediction for color image coding," *Proc. of Picture Coding Symp.-PCS '94*, Sacramento, CA, USA, Sept. 1994.

[60] T. C. Bell, J. G. Cleary, and J. H. Witten, *Text Compression*, Prentice-Hall, Englewood Cliffs, NJ, 1990.

[61] R. G. Gallangher, "Variations on a theme by Huffman," *IEEE Trans. Inform. Theory*, vol. 24, pp. 668-674, Nov. 1978.

[62] W. B. Pennebaker, J. L. Mitchell, G. G. Langdon, and R. B. Arps, "An overview of the basic principles of the Q-coder adaptive binary arithmetic coder," *IBM J. Research and Development*, vol. 32, pp. 717-726, Nov. 1988.

[63] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Information Theory*, vol. 23, no. 3, pp. 337-343, Mar. 1972.

[64] D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video," *IEEE Trans. Image Processing*, vol. 3, no. 5, pp. 572-588, May 1994.

[65] B.-J. Kim and W. A. Pearlman, "An embedded wavelet video coder using three dimensional set-partitioning in hierarchical trees (SPIHT)," *Proc. of IEEE Data Compression Conference*, Snowbird, Utah, pp. 251-260, 1997.

[66] V. N. Ramaswamy, N. Ranganathan, and K. R. Namuduri, "Performance analysis of wavelets in embedded zerotree-based lossless image coding schemes," *IEEE Trans. Signal Processing*, vol. 47, pp. 884-889, Mar. 1999.

[67] E. A. B. da Silva, D. G. Sampson and M. Ghanbari, "A successive approximation vector quantizer for wavelet transform image coding," *IEEE Trans. Image Processing*, vol. 5, pp. 299-310, Feb. 1996.

[68] R. Arps and T. Truong, "Comparison of international standards for lossless still image compression," *Proceedings of the IEEE*, vol. 82, pp. 889-899, June 1994.

[69] N. Ahmed and K. Rao, *Orthogonal transforms for digital signal processing*, Springer-Verlag, NY, 1975.

[70] E. Kuntzel, and T. Nguyen, "On structure of dyadic symmetric wavelets with integer coefficients," *IEEE Intern. Symp. Circuits and Systems*, vol. 1, pp. 181-184, Hongkong, May 1997.

[71] G. K. Wallace, "The JPEG still picture compression standard," *Comm. of the ACM*, vol. 34, pp. 30-45, April 1991.

[72] Y. W. Nijim and S. D. Stearns, "Differentiation applied to lossless compresssion of medical images," *IEEE Trans. Medical Imaging*, vol. 15, pp. 555-559, Aug. 1996.

[73] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Comm. Pure Appl. Math.*, vol. 4, pp. 909-996, Nov. 1988.

[74] Y. W. Nijim, S. D. Stearns, and W. B. Mikhael, "Lossless compression of images using a hybrid transform/differentiation technique," *Proc. of 1995 Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove, California, Oct. 29-Nov. 1, 1995.

[75] I. Daubechies, *Ten Lectures on Wavelets*, SIAM, 1992.

[76] I. Singh, P. Agathoklis, and A. Antoniou, "Lossless compression of bathymetric data using an improved nonlinear discrete wavelet transform," *Third Int. Conf. on Electronics, Circuits, and Systems*, Rodos, Greece, pp. 578-581, Oct. 1996.

[77] J. D. Foley, A. Dam, S. K. Feiner, and J. F. Huighes, *Computer Graphics: Principles and Practice*, Addison-Wesley, 1990.

[78] A. K. Jain, *Fundamentals of Digital Image Processing*, Englewood Cliffs, NJ: Prentice-Hall, 1989.

[79] C. E. Shannon, "A Mathematical theory of communications," *Bell Tech. Journal*, vol. 27, pp. 379-423, July 1948.

[80] R. M. Gray "Vector quantization," *IEEE Acoustics, Speech, and Signal Processing Magazine*, vol. 1, pp. 4-29, Apr. 1984.

[81] A. Gersho, "On the structure of vector quantizers," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 1568-1575, Oct. 1989.

[82] R. M. Gray, J. C. Kieffs, and Y. Linde, "Locally optimal block quantizer design," *Inform. Control*, vol. 45, pp. 178-198, May 1980.

[83] W. H. Equitz, "A new vector quantization clustering scheme," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 37, no. 10, Oct. 1993.

[84] I. Linares, R. M. Mersereau, and M. J. T. Smith, "Blocking reduction of landsat thematic mapper JPEG browse images using optimal PSNR estimated spectra adaptive postfiltering," *1994 Science Inform. Management and Data Compression Workshop*, vol. 3277, pp. 41, NASA Goddard Space Flight Center, Sept. 1994.

[85] S-W. Wu and A. Gersho, "Improved decoder for transform coding with applications to JPEG baseline system," *IEEE Trans. Commun.*, vol. 40, no. 2, pp. 251-254, Feb. 1992.

[86] S-W. Wu and A. Gersho, "Enhancement of transform coding by nonlinear interpolation," *Proc. SPIE Visual Commun. Image Processing*, vol. 1605, pp. 487-498, 1991.

[87] R. A Gopinath, M. Lang, H. Guo, and J. E. Odegard, "Wavelet-based post-processing of low bit-rate transform coded images," *Proc. of Int. Conf. on Image Processing*, vol. 2, pp. 913-917, Austin, TX, Nov. 1994.

[88] D. L. Donoho, *Denoising by Soft Thresholding*, Tech. Report 409, Department of Statistics, Stanford University, USA, Nov. 1992.

[89] I. M. Jonstone and B. W. Silverman, *Wavelet Threshold Estimates for Data with Correlated Noise*, Tech. Report, University of Bristol, US, Statistics Department, Sept. 1994.

[90] B. Vidakovik, *Nonlinear Wavelet Shrinkage with Bayes Rules and Bayes Factors*, Tech. Report, Duke University, ISDS, Aug. 1994.

[91] H. Lee, A. H. Rowberg, M. S. Frank, H. S. Choi and Y. Kim, "Subjective evaluation of compressed image quality," *SPIE Medical Imaging VI*, vol. 1653, pp. 241-251, Feb. 1992.

[92] H. Lee, Y. Kim, A. H. Rowberg, M. S. Frank, and W. Lee, "Lossy compression of medical images using prediction and classification," *SPIE Medical Imaging VII*, vol. 1897, pp. 282-290, Feb. 1993.

[93] A. Gentile, H. Cat, F. Kossentini, F. Sorbello, and D. S. Wills, "Real-time implementation of full-search vector quantization on a low memory SIMD architecture," *Data Compression Conference*, Snowbird, Utah, pp. 438, 1996.

[94] H. Abut, B.P. Tao, and J. Smith, "Vector quantization architectures for speech and image coding," *IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Dallas, TX, pp. 756-759, Apr. 1987.

[95] X. Sun and B. M. G Cheetham, "Speech excitation modelling for low bit speech coding," *1997 IEEE Workshop on Speech Coding for Telecomm. Proceedings*, pp. 9-10, Pennysylvania, Sept. 1997.

[96] W. B. Kleijn and J. Haagen, "A speech coder based on decomposition of characteristic waveforms," *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 508-511, Atlanta, 1995.

[97] M. R. Zad-Issa and P. Kabal, "A new LPC error criterion for improved pitch tracking," *1997 IEEE Workshop on Speech Coding for Telecomm. Proceedings*, pp. 1-2, Pennysylvania, Sept. 1997.

[98] A. Das, E. Paksoy, and A. Gersho, "Multimode and variable rate coding of speech," *Speech Coding and Synthesis*, W. B. Kleijn and K. K. Paliwal, Eds. Amsterdam, The Netherlands: Elsevier, pp. 257-288, 1995.

[99] M. Yong and A. Gersho, "Efficient encoding of the long-term predictor in Multimode vector excitation coders," *Advances in Speech Coding*, B. S. Atal, V. Cuperman, and A. Gersho, Eds. Boston, MA: Kluwer, pp. 329-338, 1991.

[100] D. Veeneman and B. Mazor, "Efficient multitap pitch prediction for stochastic coding," *Speech and Audio Coding for Wireless Networks*, B. S. Atal, V. Cuperman, and A. Gersho, Eds. Boston, MA: Kluwer, pp. 225-229, 1993.

[101] Coding of speech at 8Kbits/sec using conjugate-structure algebraic-code-excited linear predicton, *G.729 ITU-T Recommendation*, Mar. 1996.

[102] A silence compression scheme for G.729 optimized for terminals conforming to Recommendation V.70, *G.729B ITU-T Recommendation*, Nov. 1996.

[103] Reference floating-point implementation for G.729 CS-ACELP 8Kbits/sec speech coding, *G.729C ITU-T Recommendation*, Sept. 1998.

[104] 6.4Kbits/sec CS-ACELP speech coder, *G.729D ITU-T Recommendation*, Sept. 1998.

[105] 11.8Kbits/sec CS-ACELP speech coding algorithm, *G.729E ITU-T Recommendation*, Sept. 1998.

[106] B. Carnero and A. Drygajlo, "Perceptual coding of speech using a fast wavelet-packet transform algorithm," *Proc. EUSIPCO*, Trieste, Italy, pp. 1661-1664, Sept. 1996.

[107] J. Garofolo, T. Robinson and J. Fiscus, "The development of file formats for very large speech corpora: SPHERE and Shorten," *IEEE Conf. on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 113-116, 1994.

[108] I. Singh, P. Agathoklis, and A. Antoniou, "Lossless compression of color images using an improved integer-based nonlinear wavelet transform," *IEEE Int. Symp. on Circuits and Systems*, pp. 2609-2612, Hong Kong, June 1997.

[109] J-H. Chen, N. Jayant, and R. V. Cox, "Improving the performance of the 16 kbps LD-CELP speech coder," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 1, pp. 69-72, San Francisco, CA, Mar. 1992.

[110] I. Singh, P. Agathoklis, and A. Antoniou, "Wavelet-based compression of speech signals on the TMS320C30 digital signal processor," *Proc. 1998 IEEE Symp. on Advances in Digital Filtering and Signal Processing*, Victoria, BC, Canada, pp. 178-182, June 1998.

[111] Texas Instruments Inc., *TMS320C3x User's Guide*, 1992.

[112] Spectrum Signal Processing Inc., *SDSP/C30 Sbus Board User's Guide*, Document no. 500-00049, Version 1.10, Mar. 1993.

[113] Spectrum Signal Processing Inc., *DSP/C30 Sbus Board Technical Reference Manual*, Document no. 500-00081, Version 1.02, Feb. 1994.

[114] Spectrum Signal Processing Inc., *Burr-Brown Analog Daughter Module User's Manual*, Document no. 500-00060, Version 1.03, Nov. 1993.

[115] Coding of speech at 64Kbits/sec, *G.711 ITU-T Recommendation*, 1988.

Appendix A

Approximation for analysis of finite discrete sequences

Let us assume a signal to be given by discrete (sampled) values at the finest scale possible, i.e., no finer scale information is avaliable. For simplicity let us assume the finest scale subspace to be V_0. In a multiresoultion analysis, each of the subspaces V_j is spanned by $\phi_{j,l} = 2^{-j/2}\phi(2^{-j}x - l)$. Since $f \in V_0$,

$$f(x) = \sum_{n \in Z} a_n^0 \phi_{0,n} = \sum_{n \in Z} a_n^0 \phi(x - n) \tag{A.1}$$

where $a_n^0 = < f, \phi_{0,k} >$. Hence

$$f(n) = \sum_k < f, \phi_{0,k} > \phi(n - k) \tag{A.2}$$

Consequently,

$$\sum_n f(n)e^{-jn\omega} = \left(\sum_k < f, \phi_{0,k} > e^{-jk\omega}\right)\left(\sum_m \phi(m)e^{-jm\omega}\right) \tag{A.3}$$

i.e., the inner product $< f, \phi_{0,k} >$ gives the Fourier coefficients of $(\sum_n f(n)e^{-jn\omega})\,(\sum_m \phi(m)e^{-jm\omega})^{-1}$. It follows that

$$< f, \phi_{0,k} > = \sum_n a_{k-n} f(n) \tag{A.4}$$

where

$$a_m = (2\pi)^{-1} \int_0^{2\pi} e^{jm\omega} \left(\sum_l \phi(l)e^{-jl\omega}\right) d\omega \tag{A.5}$$

So for any member of the Daubechies family of wavelets, the coefficient a_m can be found by explicit integration. For the Daubechies-4 wavelet, for example, $\phi(1) = \frac{1+\sqrt{3}}{2}$ and $\phi(1) = \frac{1-\sqrt{3}}{2}$. It can shown that for the Daubechies-4 wavelet basis, $a_m = (3\sqrt{3} - 5)(2 - \sqrt{3})^m$ where $m \in \{-1,\ 0,\ 1,\ \ldots\}$. Thus, (A.2) is used to obtain the inner products $< f, \phi_{0,k} >$ (of the input sequence $f(n)$ whereas (A.4) is used to obtain the discrete sequence $f(n)$ from the inner products. It can be shown that for the Daubechies-4 wavelet, the coefficients a_m in (A.5) decay rapidly with m. For instance, $a_{254} = 1.0421 \times 10^{-146}$, so most of the a_m are zero for all practical purposes. It has been shown that for most practical purposes, the assumption that the discrete input signal $f(n)$ is in subspace V_0 holds good. In other words,

the advantage of not calculating the complex integral in (A.5) far exceeds the improvements in performance if the inner products are used for wavelet analysis rather than the discrete sequence $f(n)$ itself. Hence, this assumption is made throughout the thesis.