Reversible Wavelet Transforms and Their Application to Embedded Image Compression

by

Michael David Adams
B.A.Sc., University of Waterloo, 1993

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

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

We accept this thesis as conforming
to the required standard

Dr. Andreas Antoniou, Supervisor (Dept. of Elec. and Comp. Eng.)

Dr. Wu-Sheng Lu, Departmental Member (Dept. of Elec. and Comp. Eng.)

Dr. Reinhard Illner, Outside Member (Dept. of Mathematics and Statistics)

Dr. Dale J. Shpak, External Examiner (Adjunct Professor, Dept. of Elec. and Comp. Eng.)

Supervisor: Dr. Andreas Antoniou

ABSTRACT

The design and implementation of reversible wavelet/subband transforms and their application to reversible embedded image compression are studied. Reversible embedded image coding provides a natural way for building unified lossy/lossless image compression systems, and reversible transforms are a key component in such systems.

A lifting-based method is examined as a means for constructing reversible versions of linear M-band subband transforms. The method is shown to produce reversible transforms that well approximate their parent linear transforms even when computed using fixed-point arithmetic. Also, a simple and practical software algorithm for reversible transform construction is described. By combining ideas from both lifting and the S+P transform, a more general framework for the design of reversible transforms is proposed. This new framework generates a larger class of reversible transforms than lifting alone, and includes all lifted transforms as a subset.

Several reversible transforms constructed using lifting are employed in a reversible embedded image compression system based on the so called Embedded Zerotree Wavelet (EZW) coding scheme. Both 2-band and M-band transforms are considered. Many practically useful observations as to which transforms are most effective for various classes of images and what types of artifacts are associated with the various transforms are made. The merits of full versus partial embedding and periodic versus symmetric extension are also investigated.

Based on some of the observations made, a multi-transform approach to image compression is proposed. With this scheme, the decorrelating transform employed by the image coder is selected on a per image basis using image-specific information. Although the transform selection algorithm is extremely simple and has only modest computational requirements, results show that this new approach yields better compression performance than is possible with the use of a single fixed transform such as the S+P transform.

Examiners:

Dr. Andreas Antoniou, Supervisor (Dept. of Elec. and Comp. Eng.)

Dr. Wu-Sheng Lu, Departmental Member (Dept. of Elec. and Comp. Eng.)

Dr. Reinhard Illner, Outside Member (Dept. of Mathematics and Statistics)

Dr. Dale J. Shpak, External Examiner (Adjunct Professor, Dept. of Elec. and Comp. Eng.)

Table of Contents

List of Tables

List of Figures

List of Abbreviations

bpp	Bits per Pixel
BR	Bit Rate
CR	Compression Ratio
CREW	Compression with Reversible Embedded Wavelets
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
EZW	Embedded Zerotree Wavelet (Coding)
FIR	Finite-Length Impulse Response
GIF	Graphics Interchange Format
HS	Half-Sample Symmetric
IIR	Infinite-Length Impulse Response
ISO	International Standards Ogranization
JPEG	Joint Photographic Experts Group
LSB	Least Significant Bit
MRA	Multiresolution Analysis
MSB	Most Significant Bit
MSE	Mean Squared Error
PE	Periodic Extension
PR	Perfect Reconstruction
PSNR	Peak-Signal-to-Noise Ratio
QMF	Quadrature Mirror-Image Filter (Bank)
SE	Symmetric Extension
SPIHT	Set Partitioning in Hierarchical Trees
USC	University of Southern California
WS	Whole-Sample Symmetric

Acknowledgements

This thesis would never have been written without the generous help and support that I received from numerous people along the way. I would now like to take this opportunity to express my sincerest thanks to these individuals.

First, I would like to thank my supervisor, Dr. Andreas Antoniou, for seeing potential in me, and giving me the opportunity to work with one of the best in the field. I was also fortunate to have been given complete flexibility in choosing my research topic, and because of this, I have enjoyed my studies all the more. Andreas has always been patient and supportive. He has provided we with guidance and an environment conducive to learning and quality research. I also thank Andreas for introducing me to the world of research publications. I know that my technical writing skills have benefitted substantially as a result of his many comments. Perhaps most of all, however, I would like to thank Andreas for always having confidence in me and my abilities, even when I didn't.

I would be remiss if I also didn't thank Dr. Warren Little. I spent my first year of graduate studies under Warren's supervision. During this time, Warren was very supportive of my research efforts. I also thank him for being very understanding when I realized that my true research interests lay elsewhere (namely in digital signal processing) and for allowing me to switch research areas.

Next, I would like to express my gratitude to Dr. Wu-Sheng Lu. I don't think I have ever met a professor who is more enthusiastic about teaching. He has been a constant source of inspiration, and I hope that, one day, I can be half as good a teacher as he. I thank Dr. Lu for his excellent course on wavelets and filter banks. I certainly learned a lot from it. And I also thank Dr. Lu for his infinite patience in answering my never-ending barrage of questions about filter banks and wavelet systems.

It is often said that mathematicians and engineers live in different worlds, and I believe that there is some truth to this statement. With this said, I am grateful to Dr. Reinhard Illner for giving me the opportunity to sit in on his wavelet analysis class. Being from an engineering background myself, it was both refreshing and enlightening to study the subject of wavelets from a mathematician's point of view. I have also come to appreciate how many finer parts of the math get swept under the carpet by engineers. I also thank Dr. Illner for taking the time to answer my many questions along the way.

I will be forever indebted to John Dorocicz for some critical advice he offered me regarding graduate studies. Without his counselling services, this thesis never would have been started, much less completed.

I would like to express my gratitude to my landlady, Edele Bonnaig. I often think of Edele as a second mom. She constantly reminds me of the importance of a balanced diet, and all of those other things that mothers often do. Edele has helped me with things both academic and otherwise. Most of all, however, I would like to thank Edele just for being a good friend and giving me that hug when I really needed it.

At this point, I would like to thank Dr. Yury Stepanenko, Dr. Dale Olesky, Catherine Chang, Vicky Smith, Lynne Barrett, Maureen Denning, Annie Dong, Sami Saab, Inderpreet Singh, and Elise Fear, all of whom helped me in some capacity along the way.

Lastly, I would like to thank my wife, Mariko Morimoto, for her love, patience, understanding, and constant support. I am also grateful to Mariko for donating her old personal computer to my research cause. Having a computer at home has helped me to work much more efficiently than before. Although many miles have separated Mariko and me for the duration of my Master's studies, she has always been close to my heart. We shall soon be together again.

To my wife, Mariko

Chapter 1

Introduction

> Those who educate children well are more to be honored than parents, for these only gave life, those the art of living well.
>
> —Aristotle

1.1 Historical Perspective

The first wavelet system was constructed by Haar [19] in 1910. The Haar wavelet system, as it is now known, uses piecewise constant functions to form an orthonormal basis for $L^2(\mathbb{R})$. Although wavelet theory has a relatively long history, it was not until the 1980s that the term "wavelet" came into use. Consequently, much of the early work on wavelets was done under the guise of other names such as Littlewood-Paley theory or Calderon-Zygmund operator theory. Although wavelet theory has intimate ties with concepts from many diverse branches of mathematics and engineering, many of these linkages were not discovered until the 1980s. Until that time, wavelet theory was really a disjoint set of ideas from many areas that lacked a clear unifying framework. In some sense, wavelet theory was only in its infancy until these linkages were first established.

In the mid-to-late 1980s, a revolution in wavelet theory occurred as a result of several important discoveries. This revolution served to draw together concepts from many different areas of mathematics and engineering resulting in a unified theory for the study of wavelet systems. In 1984, the term "wavelet" was introduced by Grossman and Morlet [18]. In 1988, a tremendous breakthrough in wavelet analysis was brought about by Daubechies. In her now classic paper [14], Daubechies introduced a family of compactly supported orthogonal wavelet systems with arbitrarily high, but fixed, regularity. The construction methods she employed were also closely related to filter banks. Daubechies' work stimulated a rapid development in wavelet theory. In 1989, Mallat [29] presented the theory of multiresolution analysis and also what later became known as the Mallat algorithm. This work provided a unifying framework for the study of wavelet systems tying together many previously disjoint ideas. In 1992, Cohen, Daubechies, and Feauveau [11] established the theory of biorthogonal wavelet systems. In the case of 2-band wavelet systems, biorthogonality has the advantage that it allows for symmetric finitely-supported basis functions. This property is not possible to have with orthogonal systems except in the trivial case of the Haar and other Haar-like transforms. The symmetry property can offer significant benefits in many applications, image compression being one example.

In 1995, Sweldens [51] proposed lifting, a technique for the design and implementation of wavelet systems. Subsequently, he has authored/coauthored many papers on the subject (e.g., [15], [49], [51]). Later, in 1996, Calderbank, Daubechies, Sweldens and Yeo [10] proposed a systematic lifting-based technique for constructing reversible versions of any 2-band wavelet transform. This method is of great significance in the context of lossless signal compression.

There are many important linkages between wavelet and filter bank theory. Digital filter bank methods have a somewhat shorter history than wavelet theory. Until the linkage between filter bank theory and wavelet theory was established, filter bank theory evolved independently from wavelet theory. Early contributions to (digital) filter bank theory and subband coding were made by Crochiere, Webber, and Flanagan [12], and Croisier, Esteban, and Galand [13], and others. Later, perfect reconstruction filter banks were studied in depth by many, with major contributions by Mintzer [31], Smith and Barnwell [43], Vetterli [55], and Vaidyanathan [53].

Wavelet transforms have proven to be extremely useful for image coding as many have shown (e.g., [6], [27], [9]). Consequently, many image compression schemes have adopted the use of such transforms. In 1993, Shapiro [41] first introduced the notion of embedded coding. His coding scheme, called Embedded Zerotree Wavelet (EZW) coding, was based on a wavelet decomposition. Due to the obvious advantages of the embedding property in many applications, embedded coding quickly grew in popularity—especially as a means to build unified lossy/lossless compression systems. In 1995, Zandi et al. [60] proposed Compression with Reversible Embedded Wavelets (CREW), a reversible embedded image compression system based on some of the ideas of Shapiro. Not long after, in 1996, Said and Pearlman [37] introduced a new coding scheme known as Set Partitioning in Hierarchical Trees (SPIHT) which is similar in spirit to EZW.

1.2 Overview and Contribution of the Thesis

This thesis studies reversible wavelet transforms and their application to reversible embedded image compression. Reversible embedded image coding provides a very useful framework for building unified lossy/lossless image compression systems. This thesis draws on ideas from all of the previously described developments, but the two of most direct importance are the lifting-based method for the design of reversible wavelet transforms proposed by Calderbank et al. [10] and the embedded coding scheme proposed by Shapiro [41]. Many of the results presented are closely related to the ideas in these two papers. Some of the topics addressed in detail by this thesis include:

- methods for constructing reversible versions of M-band subband transforms (of which wavelet transforms are a special case)
- practical issues associated with the implementation of reversible transforms such as computational efficiency
- effectiveness of various transforms for both lossless and lossy image compression
- techniques for transform selection based on image-specific properties

- modifications and enhancements to the EZW scheme to handle reversible coding, M-band transforms, arbitrary-sized images, etc.

The thesis is divided into five chapters. The first two chapters provide introductory information to put this work in context and background information necessary to understand the rest of the work. The remaining chapters present a mix of research results and other fundamental concepts required to understand these results.

In Chapter 2, multirate filter banks and wavelet systems are introduced. The chapter begins by presenting the fundamentals of multirate systems. Then quadrature mirror-image filter (QMF) banks are examined in some detail. Finally, the link between QMF banks and wavelet systems is established.

In Chapter 3, reversible transforms are discussed in depth. First, the reversibility property is defined and the reason why transforms with this property are desirable is explained. Next, lifting is examined as a means to realize M-band subband transforms. This implementation technique extends the ideas presented in [15] to the case of M-band transforms. Then lifting is examined as a means for constructing reversible versions of transforms. Methods for handling finite-length signals and multidimensional signals are examined. Next, reversible transforms derived using the lifting-based method are shown to be well suited to implementation using only fixed-point arithmetic. That is, the use of fixed-point arithmetic (instead of floating-point) does not significantly affect approximation behavior of the reversible transform. The chapter also examines the problem of finding good lifting factorizations using computer software. A simple method for producing a reversible version of any M-band subband transform is described. Some good reversible transforms such as the S+P transform do not quite fit into the lifting framework. This leads to the proposal of a new more general structure for reversible transforms through extension of the ideas from lifting. Several examples of reversible transforms are given in the chapter. Some of the work in this chapter also appears in [2].

In Chapter 4, both theoretical and practical aspects of embedded image compression are discussed. The chapter begins by introducing the basic ideas behind transform-based image compression systems. Measures for compression performance are discussed. Then embedded coding is defined and its benefits explained. Next, the EZW coding scheme is briefly introduced. The chapter also evaluates the performance of several reversible transforms for both lossy and lossless compression and identifies the classes of images for which the various transforms are most effective. Based on these observations, a multi-transform approach to image compression is proposed in which the decorrelating transform is selected on a per image basis using image-specific characteristics. Modifications and enhancements to the EZW coding scheme are also discussed. Some of the work described in this chapter also appears in [1] and [4].

Finally, Chapter 5 summarizes some of the more important results presented in this thesis along with the contributions it makes. The chapter concludes by suggesting directions for future research.

1.3 Notation

Before continuing further, a brief digression regarding the notation used in this thesis is appropriate. Some of the notation and conventions used are as follows:

- The symbols $\mathbb{Z}, \mathbb{R}, \mathbb{C}$ denote the sets of integers, real numbers, and complex numbers, respectively.
- The symbol j denotes the quantity $\sqrt{-1}$.
- Functions of a continuous variable are indicated with round brackets, for example, $f(t)$ where $t \in \mathbb{R}$.
- Functions of a discrete variable are indicated with square brackets, for example, $x[n]$ where $n \in \mathbb{Z}$.
- Boldface type is used to denote matrix and vector quantities.
- The symbol $\boldsymbol{I}_k$ denotes the $k \times k$ identity matrix.
- The quantity $\boldsymbol{A}^T$ denotes the transpose of $\boldsymbol{A}$.
- The quantity $\boldsymbol{A}^*$ denotes the conjugate of $\boldsymbol{A}$.
- The quantity $\boldsymbol{A}^\dagger$ denotes the conjugate transpose of $\boldsymbol{A}$.
- For a function $H(z)$, the notation $H_*(z)$ denotes conjugation of the coefficients without conjugating z. Note that $H^*(z) = H_*(z^*)$. The notation $\tilde{H}(z)$ stands for $H_*^T(z^{-1})$. If $z = e^{j\omega}$, then $\tilde{H}(z) = H^\dagger(z)$.
- The z-transform of a sequence $x[n]$ is denoted as $X(z)$ and is defined as

$$\mathcal{Z}\{x[n]\} = X(z) = \sum_n x[n] z^{-n}$$

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

$$\langle f(t), g(t) \rangle = \int_{-\infty}^{\infty} f(t) g(t) dt$$

  and in the discrete variable case as

$$\langle f[n], g[n] \rangle = \sum_n f[n] g[n]$$

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

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

- The notation $f * g$ stands for the convolution of f and g.
- The notation $\lfloor x \rfloor$ denotes the greatest integer not more than x (or equivalently, x rounded to the nearest integer in the direction of $-\infty$).
- The notation $n|m$ means that n is evenly divisble by m.

- Filter coefficients are always assumed to be real-valued unless explicitly stated otherwise.
- The sampling period for all discrete sequences is assumed to be one. In other words, the sampling period is always normalized to be one. Thus, if we are given a sequence $x[n]$ with z-transform $X(z)$, the frequency spectrum associated with the signal is always obtained by evaluating $X(z)$ at $z = e^{j\omega}$.
- A filter H has the transfer function $H(z)$. That is, roman text is used to name a particular filter while italics are used to indicate the transfer function associated with the filter.

... And that's why wavelets are so popular. Any idiot can use them.

—Anonymous professor

Chapter 2

Multirate Filter Banks and Wavelet Systems

There is only one thing you should do. Go into yourself. Find out the reason that commands you to write; see whether it has spread its roots into the very depths of your heart; confess to yourself whether you would have to die if you were forbidden to write. This most of all: ask yourself in the most silent hour of your night: must I write? Dig into yourself for a deep answer. And if this answer rings with a strong, simple "I must," then build your life in accordance with this necessity; your whole life, even into its humblest and most indifferent hour, must become a sign and witness to this impulse.

—Rainer Maria Rilke, "Letters to a young poet"

2.1 Introduction

Multirate systems and filter banks play an important role in the study of wavelet systems. In particular, the class of systems known as quadrature mirror-image filter (QMF) banks are especially significant in this regard. Not only can wavelet transforms be constructed through the design of QMF banks, but they can also be implemented very efficiently in terms of these structures.

In order to study QMF banks, we must first understand the fundamentals of multirate systems. This chapter begins by introducing some basic multirate system concepts, and then uses these concepts to establish a general framework for the study of QMF banks. Lastly, the link between QMF banks and wavelet systems is established.

In addition to the material presented in this thesis, the author recommends [54], [56], and [46] for more detailed coverage of multirate filter banks and wavelet systems.

2.2 Multirate Systems

Depending on the number of different sampling rates it employs, a discrete-time system can be classified as being either unirate or multirate. A system that processes signals at a single sampling rate is referred to as unirate. Most of us are all too familiar with unirate systems as they are traditionally studied in any introductory digital signal processing course. In contrast, a system that processes signals at more than one sampling rate is referred to as multirate.

Multirate systems are extremely useful for many signal processing applications. Often, a multirate system can be used to perform a task more easily or efficiently than is possible with a unirate system. Other times, a signal processing task inherently requires the use of multiple sampling rates. In such cases, a multirate system must be used.

The basic building blocks of unirate systems are familiar to us all, namely, the unit delay/advance, adder, and multiplier. Multirate systems have these same building blocks plus two additional ones: the downsampler and upsampler. These two new building blocks are examined next.

2.3 Downsampler

One of the basic building blocks of multirate systems is the downsampler, sometimes also referred to as a compressor[1]. The M-fold downsampler, shown in Figure 2.1, takes an input sequence $x[n]$ and produces the output sequence

$$y[n] = x[Mn] \tag{2.1}$$

where M is an integer. The constant M is referred to as the downsampling factor. In simple terms, the downsampling operation keeps every Mth sample and discards the others. From equation (2.1), it follows that downsampling is a linear time-varying operation. The relationship between the input and output of the downsampler in the z-domain is given by

$$Y(z) = \frac{1}{M} \sum_{k=0}^{M-1} X(z^{1/M} e^{-j2\pi k/M}) \tag{2.2}$$

Assuming that the sampling period before and after downsampling is normalized to one, this leads directly to the frequency domain relation

$$Y(e^{j\omega}) = \frac{1}{M} \sum_{k=0}^{M-1} X\left(e^{j(\omega - 2\pi k)/M}\right) \tag{2.3}$$

Downsampling has a very simple frequency domain interpretation. The spectrum of the downsampled signal 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 (2.3), 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

[1] The term "decimator" is also frequently employed as a synonym for "downsampler". Unfortunately, "decimator" can also be used to mean "a downsampler in cascade with an anti-aliasing filter", as in the case of sampling rate conversion applications. Thus, in order to avoid confusion, the author prefers to use the terms "downsampler" and "compressor" instead of "decimator".

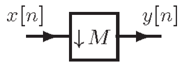


Figure 2.1. *M-fold downsampler.*

from its downsampled version. Aliasing can be avoided if $x[n]$ is a lowpass signal bandlimited to $|\omega| < \pi/M$. This is equivalent to saying that the Nyquist criterion must not be violated if aliasing is to be avoided.

To illustrate the frequency domain effects of downsampling, let us consider two examples. In the first example, the input signal is chosen to be sufficiently bandlimited that aliasing will not occur. In the second example, the input signal is selected to result in aliasing.

In the first case, suppose we take a signal with the spectrum shown in Figure 2.2(a) and apply it 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 2.2(b). Due to the renormalization of the sampling period, these spectra also appear stretched. The resulting spectrum is shown in Figure 2.2(c). Because the two spectra in Figure 2.2(b) do not overlap, there is no aliasing. The shape of the original spectrum is clearly discernable in the final output spectrum.

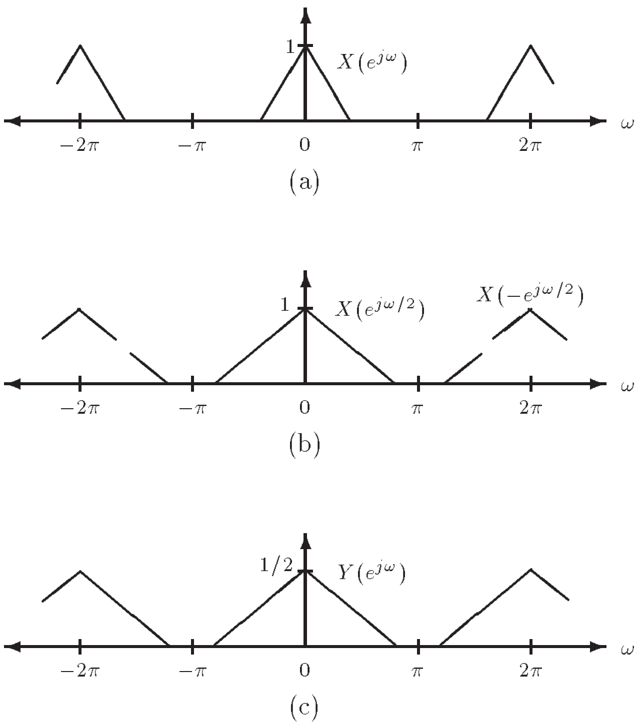


Figure 2.2. *Effects of two-fold downsampling in the frequency domain (no aliasing case). (a) Spectrum of the input signal. (b) The two stretched and shifted versions of the original input spectrum used to form the spectrum of the downsampled signal. (c) Spectrum of the downsampled signal.*

In the second case, suppose we take a signal with the spectrum shown in Figure 2.3(a) and apply it 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 2.3(b). Again, these spectra appear stretched due to the renormalization of the sampling period. The resulting spectrum is shown in Figure 2.3(c). Because in Figure 2.3(b) the two spectra overlap, aliasing occurs. Consequently, it is not possible to recover the original signal from its downsampled version. This is also evident from the spectrum of the downsampled signal shown in Figure 2.3(c). Due to aliasing, the spectrum has been distorted and no longer resembles that of the original input.

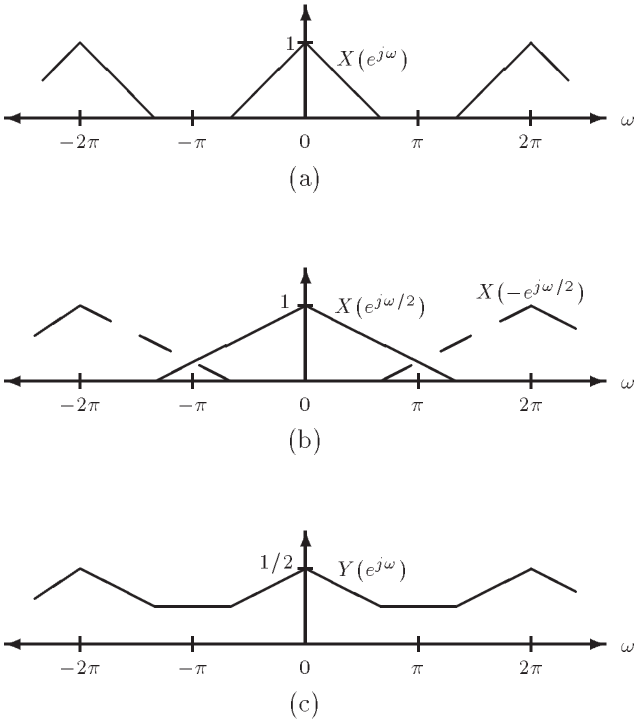


Figure 2.3. *Effects of two-fold downsampling in the frequency domain (aliasing case). (a) Spectrum of the input signal. (b) The two stretched and shifted versions of the original input spectrum used to form the spectrum of the downsampled signal. (c) Spectrum of the downsampled signal.*

2.4 Upsampler

Another basic building block of multirate systems is the upsampler which is also sometimes referred to as an expander[2]. The M-fold upsampler, depicted in Figure 2.4, takes an input sequence $x[n]$

[2]The term "interpolator" is also frequently employed as a synonym for "upsampler". Unfortunately, "interpolator" has many other different and conflicting meanings. For this reason, the author favors the use of the terms "upsampler" and "expander" instead of "interpolator".

and produces the output sequence

$$y[n] = \begin{cases} x[n/M] & \text{if } n \text{ is an integer multiple of } M \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

where M is an integer. The constant M is referred to as the upsampling factor. In simple terms, upsampling results in the insertion of $M-1$ zeros between the samples of the original signal. From equation (2.4), it follows that upsampling is a linear time-varying operation. The relationship between the input and output of the upsampler in the z-domain is given by

$$Y(z) = X(z^M) \tag{2.5}$$

Assuming that the sampling period before and after upsampling is normalized to one, this directly yields the frequency domain relation

$$Y(e^{j\omega}) = X(e^{j\omega M}) \tag{2.6}$$

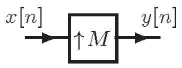


Figure 2.4. *M-fold upsampler.*

Upsampling has a very straightforward interpretation in the frequency domain. The upsampling process simply serves to move 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 is also 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 upsampling itself.

Since the shape of the spectrum is not altered by upsampling, there is no information loss and the original signal can always be recovered from its upsampled version. Upsampling, however, does result in the creation of multiple copies of the original baseband spectrum. These copies are referred to as images. And this phenomenon is called imaging.

To illustrate the frequency domain effects of upsampling, let us consider an example. Suppose we take a signal with the spectrum shown in Figure 2.5(a) and apply it to the input of a two-fold upsampler. The spectrum of the upsampled signal is simply that shown in Figure 2.5(b). Due to the sampling period renormalization, the spectrum appears compressed. Although multiple copies of the original baseband spectrum appear, each copy has the same shape as the original input spectrum. As a result, the original signal can always be recovered from the upsampled signal with the appropriate use of filtering.

2.5 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 system

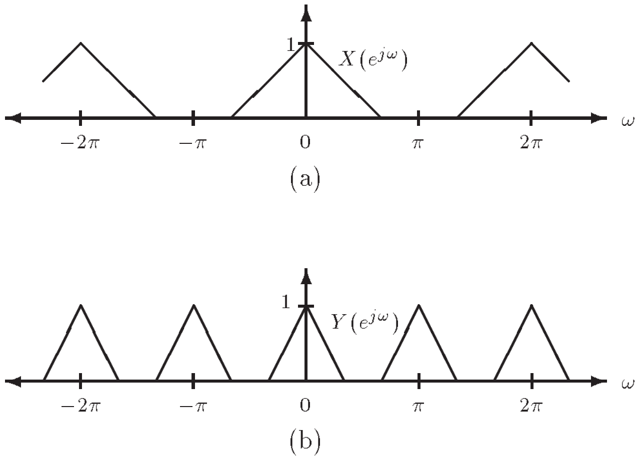


Figure 2.5. *Effects of two-fold upsampling in the frequency domain. (a) Spectrum of the input signal. (b) Spectrum of the upsampled signal.*

behavior, it is sometimes 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 2.6 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 2.7. It is important to emphasize that these identities hold only if the transfer function $G(z)$ is a rational polynomial expression.

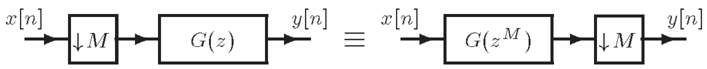


Figure 2.6. *First noble identity.*

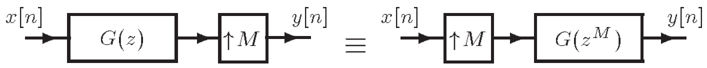


Figure 2.7. *Second noble identity.*

In addition to their theoretical utility, the noble identities are of great practical significance. For performance reasons, it is usually desirable to perform filtering operations on the side of an upsampler (or downsampler) with the lower sampling rate. Using the noble identities, we can move filtering operations across upsamplers (or downsamplers) and achieve improved computational efficiency.

2.6 Polyphase Form of a Filter

The concept of a polyphase representation or realization is one of fundamental importance in the study of multirate systems. The utility of polyphase methods is twofold. First, these methods provide a mathematically convenient representation for filtering operations in a multirate framework. This representation, known as the polyphase representation, is extremely useful as it greatly simplifies many theoretical results allowing easier analysis of multirate systems. Second, polyphase methods offer a desirable means for implementing filtering operations in a multirate framework. This implementation strategy, known as the polyphase realization, leads to computationally efficient structures for many multirate (and even some unirate) systems.

The polyphase representation of a filter is nothing more than the filter's transfer function expressed in a special form. Suppose we are given a filter with the transfer function

$$G(z) = \sum_{n=-\infty}^{\infty} g[n] z^{-n}$$

The polyphase representation of this transfer function has the general form

$$G(z) = \sum_{l=0}^{M-1} z^{a_l} B_l(z^M) \tag{2.7}$$

where

$$B_l(z) = \sum_{n=-\infty}^{\infty} g[Mn - a_l] z^{-n}$$

and the a_l are integers chosen such that

$$a_i \bmod M \neq a_j \bmod M \quad \text{for all } i, j \text{ with } i \neq j$$

Functions $B_l(z)$ are called the polyphase components of $G(z)$, and M denotes the number of phases in the polyphase decomposition.

Although many different variations on the polyphase representation are possible, there are four specific representations that are most frequently used in practice. Two of these representations are commonly designated as type 1 and type 2 polyphase representations (e.g., as in [54], [46]). The other two representations do not have standard names associated with them, and for convenience will be referred to as type 3 and type 4 polyphase representations. In the case of these four commonly used polyphase representations, the a_l in equation (2.7) are chosen as

$$a_l = \begin{cases} -l & \text{type 1} \\ -(M-1-l) & \text{type 2} \\ l & \text{type 3} \\ -l & \text{type 4} \end{cases} \tag{2.8}$$

Different choices of a_l serve only to time shift and permute the polyphase components. For example, the type 2 polyphase components of $G(z)$ are simply a permutation of the type 1 polyphase components of $G(z)$. More specifically, the order of the components are reversed with respect to one another.

When realized in its type 1, 2, 3, or 4 M-phase polyphase form, a filter is implemented using a delay/advance chain, adders, and M filters each having a transfer function that is a rational polynomial expression in z^M. The structures corresponding to the four types of polyphase representations are presented in Figures 2.8, 2.9, 2.10, and 2.11.

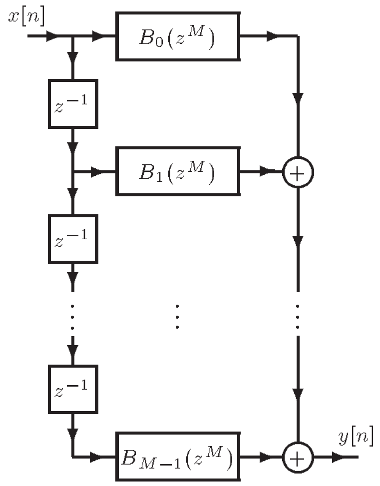


Figure 2.8. *Type 1 polyphase realization of a filter.*

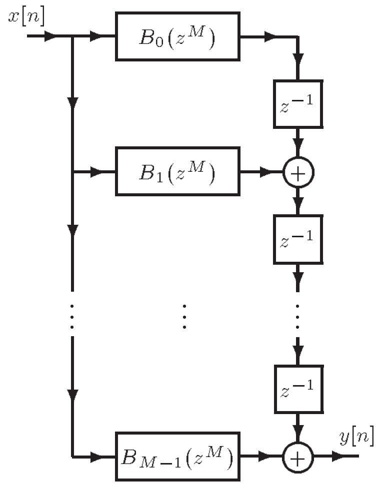


Figure 2.9. *Type 2 polyphase realization of a filter.*

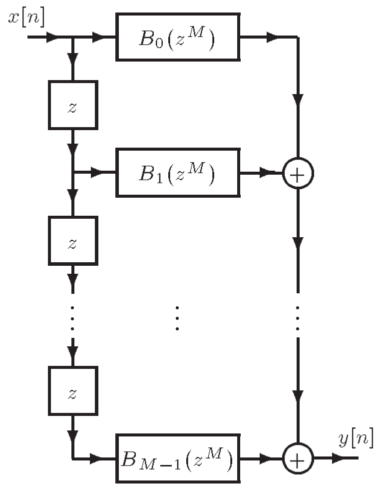


Figure 2.10. *Type 3 polyphase realization of a filter.*

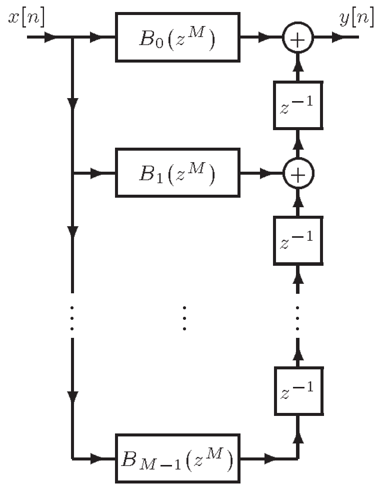


Figure 2.11. *Type 4 polyphase realization of a filter.*

Although the type 1 and type 4 polyphase representations appear mathematically identical, there is a subtle distinction between the two. Each representation is also associated with a particular implementation structure. The type 1 and type 4 polyphase representations look the same mathematically, but their implied realization structures are different.

In multirate systems, filters are often connected in cascade with upsamplers/downsamplers. For reasons of computational efficiency, it is usually preferable to perform any filtering on the side of the upsampler/downsampler with the lower sampling rate. We have seen how the noble identities can be used to move a filtering operation across upsamplers/downsamplers, but in order to move filtering to the side with the lower sampling rate, the transfer function of the filter must be an expression in z^M which is not generally the case. The polyphase representation, however, provides us with a means to express any filter as a set of filters with transfer functions in z^M so that filtering can be moved to the more desirable side of an upsampler/downsampler.

Suppose a filter realized in either its type 1 or type 3 polyphase form is immediately followed by an M-fold downsampler. In such a case, the first noble identity can be used to move the downsampling operation before the polyphase filtering operations. Likewise, suppose a filter realized in either its type 2 or type 4 polyphase form is immediately preceded by an M-fold upsampler. In such a case, the second noble identity can be used to move the upsampling operation after the polyphase filtering operations. In this way, we can use the polyphase implementation of filters to obtain efficient multirate structures.

2.7 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 bank. When they share a common output, they form a synthesis bank. These two types of filter banks are depicted in Figures 2.12 and 2.13. Each of the filter banks shown consists of M filters. The filters H_k belonging to the analysis bank are called analysis filters and the filters F_k comprising the synthesis bank are referred to as synthesis filters. The signals $u_k[n]$ and $v_k[n]$ are called subband signals. The frequency responses of the analysis/synthesis filters may be non-overlapping, marginally overlapping, or greatly overlapping depending on the application.

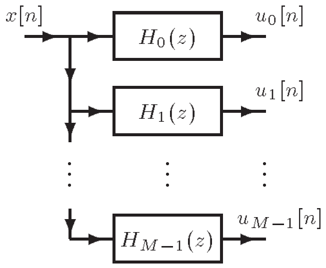


Figure 2.12. *Analysis bank.*

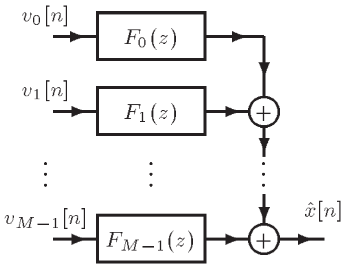


Figure 2.13. *Synthesis bank.*

2.8 M-Channel QMF Banks

Although many filter bank configurations exist, an extremely useful one is the so called M-channel quadrature mirror-image filter (QMF) bank. Such a system is more correctly referred to as a uniformly maximally decimated filter bank, but the term QMF bank is often used for historical reasons. The general structure of such a system is shown in Figure 2.14. The input signal $x[n]$ is split into M channels, processed by the analysis filters H_k, downsampled by M, upsampled by M, transformed by the synthesis filters F_k, and then summed to produce the output signal $\hat{x}[n]$. Since each of the M channels is downsampled and upsampled by M, the filter bank is referred to as uniformly maximally decimated.

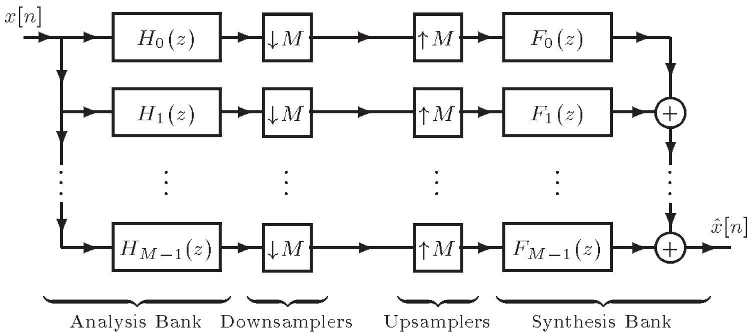


Figure 2.14. *M-channel QMF bank.*

2.9 Perfect Reconstruction QMF Banks

For the QMF bank of Figure 2.14 to be of practical use, the output $\hat{x}[n]$ is usually required to be an accurate reproduction of the input $x[n]$. If the system is such that $\hat{x}[n] = x[n - n_0]$ for all $x[n]$ and for some integer n_0, the system is said to have the perfect reconstruction (PR) property. In other words, a PR system can reproduce the input signal exactly except for a time shift. Generally, there are three reasons[3] that the reconstructed signal $\hat{x}[n]$ can differ from $x[n]$: aliasing distortion, amplitude distortion, and phase distortion. The analysis and synthesis filters can be designed in such a way so as to eliminate some or all of these distortions depending on what is required by the application.

2.10 Polyphase Form of a QMF Bank

Although the structure for the M-channel QMF bank shown in Figure 2.14 may be intuitively appealing, it is often not the most convenient structure with which to work. This leads us to

[3] Coding/quantization of the subband signals can also introduce distortion, but this distortion cannot be corrected and is therefore not considered.

examine the polyphase representation of a QMF bank. The polyphase representation has many advantages, but most importantly it simplifies many theoretical results and suggests an efficient means by which to implement filter banks.

The polyphase form of a QMF bank is based on the polyphase representation discussed earlier in the context of individual filters. Although many different variations on the polyphase form of a QMF bank exist, two particular variants are most commonly used. Since no standard terminology exists in the literature to distinguish between these two variants, the author introduces his own at this point. The first variant is referred to as the type 1/2 polyphase representation. It involves representing the filters of the analysis bank in their type 1 polyphase form and the filters of the synthesis bank in their type 2 polyphase form. The second variant is called the type 3/4 polyphase representation. It involves representing the filters of the analysis bank in their type 3 polyphase form and the filters of the synthesis bank in their type 4 polyphase form.

Suppose we have an M-channel QMF bank with analysis filters H_k and synthesis filters F_k. For notational convenience, we define the quantities

$$\boldsymbol{h}(z) = \begin{bmatrix} H_0(z) \\ H_1(z) \\ \vdots \\ H_{M-1}(z) \end{bmatrix}, \quad \boldsymbol{f}(z) = \begin{bmatrix} F_0(z) \\ F_1(z) \\ \vdots \\ F_{M-1}(z) \end{bmatrix}, \quad \boldsymbol{e}_a(z) = \begin{bmatrix} 1 \\ z \\ \vdots \\ z^{M-1} \end{bmatrix}, \quad \boldsymbol{e}_d(z) = \begin{bmatrix} 1 \\ z^{-1} \\ \vdots \\ z^{-(M-1)} \end{bmatrix}$$

First, let us consider the analysis filters of the QMF bank. We can express the transfer functions $H_k(z)$ in polyphase form as

$$H_k(z) = \sum_{l=0}^{M-1} z^{a_l} E_{k,l}(z^M)$$

This equation can be rewritten in matrix form as

$$\begin{bmatrix} H_0(z) \\ H_1(z) \\ \vdots \\ H_{M-1}(z) \end{bmatrix} = \begin{bmatrix} E_{0,0}(z^M) & E_{0,1}(z^M) & \cdots & E_{0,M-1}(z^M) \\ E_{1,0}(z^M) & E_{1,1}(z^M) & \cdots & E_{1,M-1}(z^M) \\ \vdots & \vdots & \ddots & \vdots \\ E_{M-1,0}(z^M) & E_{M-1,1}(z^M) & \cdots & E_{M-1,M-1}(z^M) \end{bmatrix} \begin{bmatrix} z^{a_0} \\ z^{a_1} \\ \vdots \\ z^{a_{M-1}} \end{bmatrix}$$

or more compactly as

$$\boldsymbol{h}(z) = \boldsymbol{E}(z^M)\boldsymbol{a}(z) \tag{2.9}$$

where

$$\boldsymbol{E}(z) = \begin{bmatrix} E_{0,0}(z) & E_{0,1}(z) & \cdots & E_{0,M-1}(z) \\ E_{1,0}(z) & E_{1,1}(z) & \cdots & E_{1,M-1}(z) \\ \vdots & \vdots & \ddots & \vdots \\ E_{M-1,0}(z) & E_{M-1,1}(z) & \cdots & E_{M-1,M-1}(z) \end{bmatrix}, \quad \boldsymbol{a}(z) = \begin{bmatrix} z^{a_0} \\ z^{a_1} \\ \vdots \\ z^{a_{M-1}} \end{bmatrix}$$

From equation (2.8), for type 1 and type 3 polyphase decompositions of the analysis bank, we have

$$\boldsymbol{a}(z) = \begin{cases} \boldsymbol{e}_d(z) & \text{type 1} \\ \boldsymbol{e}_a(z) & \text{type 3} \end{cases}$$

Equation (2.9) completely characterizes the analysis bank and is called the polyphase representation of the analysis bank. Matrix $\boldsymbol{E}(z)$ is referred to as the analysis polyphase matrix.

Now, let us consider the synthesis filters of the QMF bank. We can express the transfer functions $F_k(z)$ in polyphase form as

$$F_k(z) = \sum_{l=0}^{M-1} z^{b_l} R_{l,k}(z^M)$$

In matrix form, this becomes

$$\begin{bmatrix} F_0(z) & F_1(z) & \cdots & F_{M-1}(z) \end{bmatrix} =$$
$$\begin{bmatrix} z^{b_0} & z^{b_1} & \cdots & z^{b_{M-1}} \end{bmatrix} \begin{bmatrix} R_{0,0}(z^M) & R_{0,1}(z^M) & \cdots & R_{0,M-1}(z^M) \\ R_{1,0}(z^M) & R_{1,1}(z^M) & \cdots & R_{1,M-1}(z^M) \\ \vdots & \vdots & \ddots & \vdots \\ R_{M-1,0}(z^M) & R_{M-1,1}(z^M) & \cdots & R_{M-1,M-1}(z^M) \end{bmatrix}$$

which can be written more concisely as

$$\boldsymbol{f}^T(z) = \boldsymbol{b}^T(z)\boldsymbol{R}(z^M) \tag{2.10}$$

where

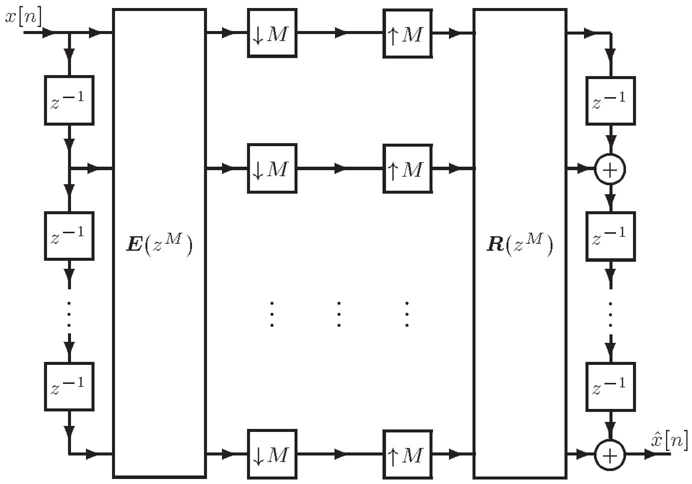
$$\boldsymbol{R}(z) = \begin{bmatrix} R_{0,0}(z) & R_{0,1}(z) & \cdots & R_{0,M-1}(z) \\ R_{1,0}(z) & R_{1,1}(z) & \cdots & R_{1,M-1}(z) \\ \vdots & \vdots & \ddots & \vdots \\ R_{M-1,0}(z) & R_{M-1,1}(z) & \cdots & R_{M-1,M-1}(z) \end{bmatrix}, \quad \boldsymbol{b}(z) = \begin{bmatrix} z^{b_0} \\ z^{b_1} \\ \vdots \\ z^{b_{M-1}} \end{bmatrix}$$

From equation (2.8), for type 2 and type 4 polyphase decompositions of the synthesis bank, we have

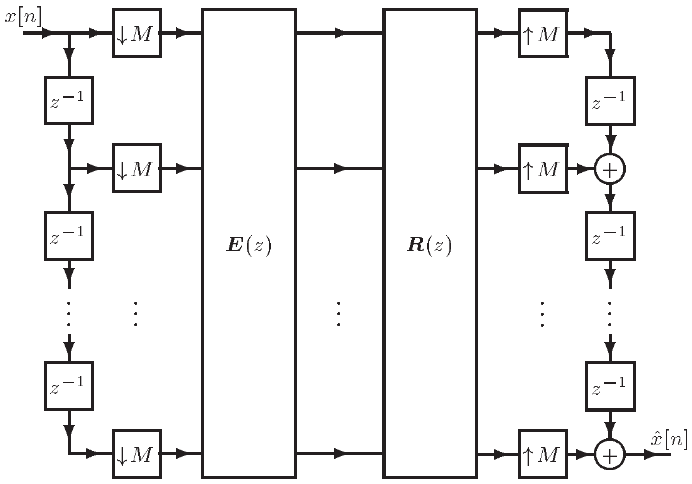
$$\boldsymbol{b}(z) = \begin{cases} z^{-(M-1)}\tilde{\boldsymbol{e}}_d(z) & \text{type 2} \\ \boldsymbol{e}_d^T(z) & \text{type 4} \end{cases}$$

Equation (2.10) completely characterizes the synthesis bank and is called the polyphase representation of the synthesis bank. Matrix $\boldsymbol{R}(z)$ is referred to as the synthesis polyphase matrix.

Notice that equations (2.9) and (2.10) provide an alternative way in which to express the analysis and synthesis banks of the QMF bank. Suppose now that we have a QMF bank where the analysis and synthesis banks have been decomposed using type 1 and type 2 polyphase representations, respectively. In this case, these equations give us the transformed, but mathematically equivalent, system shown in Figure 2.15(a). Using the noble identities, however, we can move the analysis polyphase filtering to the right of the downsamplers and the synthesis polyphase filtering to the left of the upsamplers. This gives us the polyphase form of the filter bank shown in Figure 2.15(b). Given a QMF bank where the analysis and synthesis banks have been decomposed using type 3 and type 4 polyphase representations, respectively, we can use the same process to obtain the structures in Figures 2.16(a) and 2.16(b).



(a)



(b)

Figure 2.15. *Type 1/2 polyphase form of an M-channel QMF bank. (a) Before simplification. (b) After rearrangement using the noble identities.*

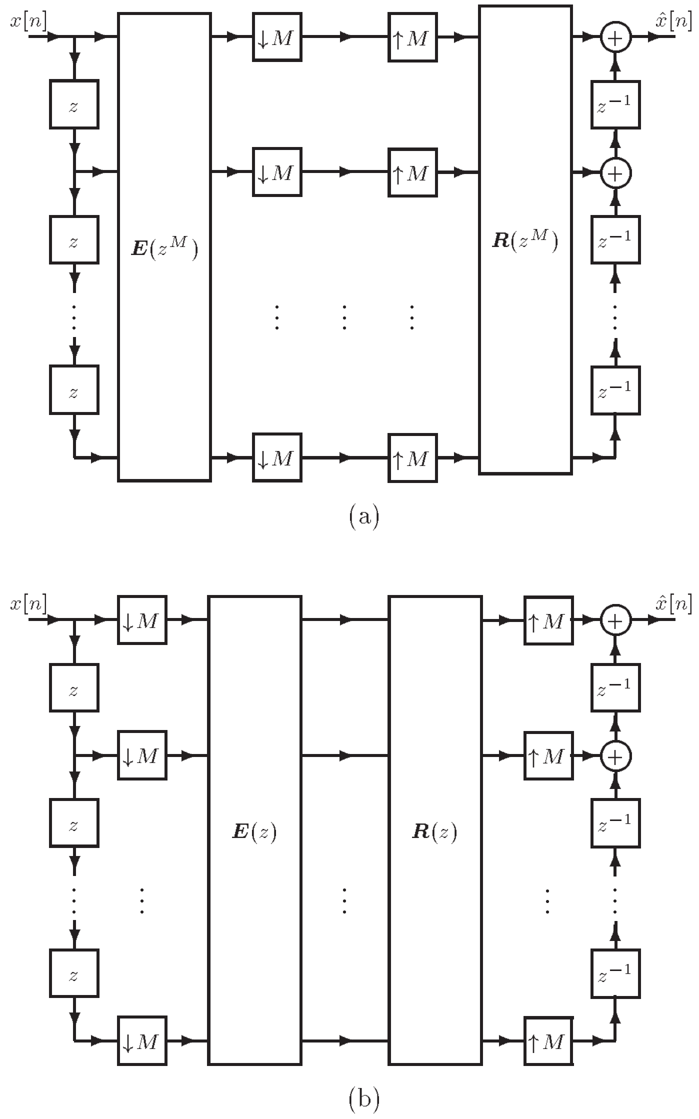


Figure 2.16. *Type 3/4 polyphase form of an M-channel QMF bank. (a) Before simplification. (b) After rearrangement using the noble identities.*

2.11 Effects of Filter Normalization

Sometimes it is desirable to transform the filters of a QMF bank by adding a constant gain and phase delay to them. Of course, such an transformation changes the underlying polyphase matrices. The effects of such a change are summarized by the theorem below. This theorem will prove extremely useful in later developments.

Theorem 2.1 *Suppose we are given an M-channel QMF bank having analysis and synthesis filters with transfer functions $H_k(z)$ and $F_k(z)$, respectively. Further, assume that the QMF bank is repre-*

sented in either type 1/2 or type 3/4 polyphase form with analysis and synthesis polyphase matrices $\boldsymbol{E}(z)$ *and* $\boldsymbol{R}(z)$*, respectively.*

Consider now a second set of analysis and synthesis filters with transfer functions $H'_k(z)$ *and* $F'_k(z)$*, respectively, related to the original filters as*

$$\begin{aligned} H'_k(z) &= \beta_0 z^{-L_0} H_k(z) \\ F'_k(z) &= \beta_1 z^{-L_1} F_k(z) \end{aligned}$$

where the β_i *are nonzero constants, and the* L_i *are integers. In other words, the new filters are obtained by adding a constant gain and constant phase delay to the original filters. This transformation results in the new analysis and synthesis polyphase matrices* $\boldsymbol{E}'(z)$ *and* $\boldsymbol{R}'(z)$*, respectively. Given the relationship between the original and transformed filters, the original and transformed polyphase matrices are related as follows:*

$$\boldsymbol{E}'(z) = \begin{cases} \beta_0\ \boldsymbol{E}(z) \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_0} & \text{type 1/2} \\ \beta_0\ \boldsymbol{E}(z) \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_0} & \text{type 3/4} \end{cases}$$

$$\boldsymbol{R}'(z) = \begin{cases} \beta_1 \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_1} \boldsymbol{R}(z) & \text{type 1/2} \\ \beta_1 \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_1} \boldsymbol{R}(z) & \text{type 3/4} \end{cases}$$

Proof. See Appendix A. ■

2.12 Conditions for PR System

Since it is often desirable to have a filter bank with the PR property, it is only natural to wonder what conditions the filter bank must satisfy in order to have PR. Fortunately, there is a very simple and practical answer to this question. The answer lies in the polyphase matrices and is given by the following theorem.

Theorem 2.2 *An* M*-channel QMF bank in either type 1/2 or type 3/4 polyphase form with analysis polyphase matrix* $\boldsymbol{E}(z)$ *and synthesis polyphase matrix* $\boldsymbol{R}(z)$ *has the PR property if and only if the product* $\boldsymbol{P}(z) \triangleq \boldsymbol{R}(z)\boldsymbol{E}(z)$ *has the form*

$$\boldsymbol{P}(z) = \begin{cases} \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{K} & \text{type 1/2} \\ \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{K} & \text{type 3/4} \end{cases}$$

for some integer K. If this condition is satisfied, the relationship between the input signal $x[n]$ and the reconstructed signal $\hat{x}[n]$ is given by

$$\hat{x}[n] = x[n - n_0]$$

where

$$n_0 = \begin{cases} K + M - 1 & \text{type 1/2} \\ K & \text{type 3/4} \end{cases}$$

This theorem holds regardless of whether the analysis/synthesis filters are of the FIR or IIR type.

Proof. See Appendix A. ■

The preceding theorem has some interesting implications—the most important of which is stated in the following corollary:

Corollary 2.3 *Any M-channel PR QMF bank in either type 1/2 or type 3/4 polyphase form with analysis polyphase matrix $\boldsymbol{E}(z)$ and synthesis polyphase matrix $\boldsymbol{R}(z)$ must be such that the product $\boldsymbol{P}(z) \triangleq \boldsymbol{R}(z)\boldsymbol{E}(z)$ has a determinant of the form*

$$\det \boldsymbol{P}(z) = (-1)^{K(M-1)} z^{-K}$$

where K is an integer. Moreover, if the analysis and synthesis filters of the QMF bank are of the FIR type, this condition implies that the polyphase matrices must have determinants of the form

$$\det \boldsymbol{E}(z) = \alpha_0 z^{-L_0}$$
$$\det \boldsymbol{R}(z) = \alpha_1 z^{-L_1}$$

where the α_i are nonzero constants and the L_i are integers. That is, the determinants of the polyphase matrices must be monomials.

Proof. See Appendix A. ■

One comment is in order concerning Theorem 2.2. If we let $K = 0$, $\boldsymbol{P}(z)$ becomes the $M \times M$ identity matrix. Since the identity matrix is sometimes an attractive matrix with which to work, one might wonder what degree of freedom is lost by constraining K to be zero. As it turns out, not much is sacrificed. This quantity only serves to introduce additional delay into the analysis/synthesis filters. For this reason, we often only consider the case of $\boldsymbol{P}(z) = \boldsymbol{I}$ when designing PR filter banks. Delay (or advance) can always be added to the analysis and synthesis filters after the fact if required. Finally, with $\boldsymbol{P}(z) = \boldsymbol{I}$, the problem of designing PR filter banks, in some sense, reduces to a problem of factorizing the identity matrix.

2.13 Octave-Band Filter Banks

The analysis side of an M-channel QMF bank decomposes the input signal $x[n]$ into M subband signals $v_k[n]$. The synthesis side then recombines these subband signals to obtain $\hat{x}[n]$, the reconstructed version of the original signal. There is nothing, however, to prevent the use of additional

QMF banks to further decompose some or all of the subband signals $v_k[n]$. Of course, some or all of the resulting subband signals can again be decomposed with even more QMF banks. In other words, this idea can be applied recursively, and the final result is a filter bank with a tree structure. If a tree structured QMF bank is such that

1. only the lowpass subband signal is decomposed at each level in the tree,
2. the same basic QMF bank building block is used for decomposition at all levels, and
3. this basic block has PR and satisfies certain regularity conditions

then the filter bank can be shown to compute a wavelet decomposition of some continuous-time signal associated with the input signal $x[n]$. Such a tree-structured filter bank is called an octave-band filter bank. The analysis side of the octave-band filter bank calculates the forward wavelet transform and the synthesis side calculates the inverse wavelet transform.

At this point, our motivation for studying M-channel PR QMF banks becomes apparent. Under the conditions stated above, an M-channel PR QMF bank can be directly linked to an M-band wavelet decomposition. Thus, QMF banks can be used to both design and implement M-band wavelet transforms.

2.14 QMF Bank Implementation

In principle, the design of a PR QMF bank amounts to decomposing the identity matrix into two factors with desired properties. These two factors are simply the polyphase matrices of the filter bank. Once we have the analysis and synthesis polyphase matrices $\boldsymbol{E}(z)$ and $\boldsymbol{R}(z)$, respectively, we are ready to proceed to the implementation of the filter bank. Of course, the filter bank could be realized by directly implementing the filtering operations in each of the polyphase matrices, but it is often beneficial to break the filtering process into a number of smaller and simpler cascaded stages.

Consider for a moment the analysis side of the filter bank. Instead of implementing $\boldsymbol{E}(z)$ directly, we further decompose $\boldsymbol{E}(z)$ as follows

$$\boldsymbol{E}(z) = \boldsymbol{E}_{n-1}(z) \cdots \boldsymbol{E}_1(z)\boldsymbol{E}_0(z)$$

Each of the $\boldsymbol{E}_i(z)$ can then be taken to represent a single filtering stage in the final implementation as depicted in Figure 2.17. Similarly, $\boldsymbol{R}(z)$ can be decomposed to produce

$$\boldsymbol{R}(z) = \boldsymbol{R}_{m-1}(z) \cdots \boldsymbol{R}_1(z)\boldsymbol{R}_0(z)$$

This corresponds to the cascade realization of the synthesis polyphase matrix shown in Figure 2.18. In the event that $\boldsymbol{R}(z)\boldsymbol{E}(z) = \boldsymbol{I}$, we have $\boldsymbol{R}(z) = \boldsymbol{E}^{-1}(z)$. Thus, we could choose $\boldsymbol{R}(z)$ as

$$\boldsymbol{R}(z) = \boldsymbol{E}_0^{-1}(z)\boldsymbol{E}_1^{-1}(z) \cdots \boldsymbol{E}_{n-1}^{-1}(z)$$

This factorization results in a certain symmetry between the analysis and synthesis sides of the filter bank which can often be advantageous.

Assuming that we want to realize the polyphase matrices with a number of cascaded blocks, what type of polyphase matrix factorization might we use? Also, given that we would like to construct reversible transforms, are there particular factorizations that help to achieve this goal? These questions will be addressed at length in the next chapter.

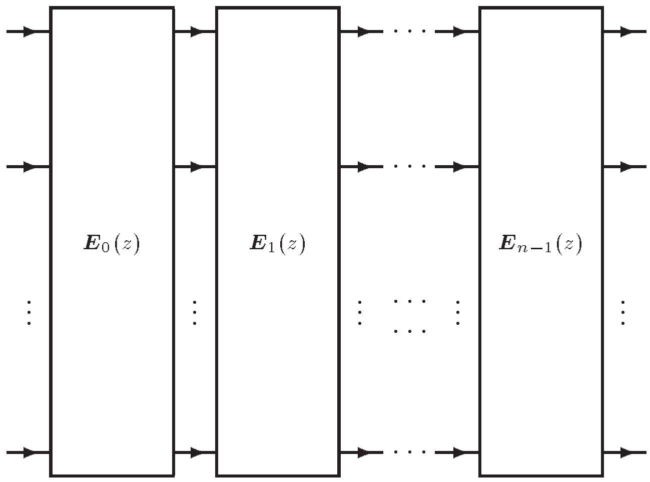


Figure 2.17. *Block cascade realization of analysis polyphase matrix.*

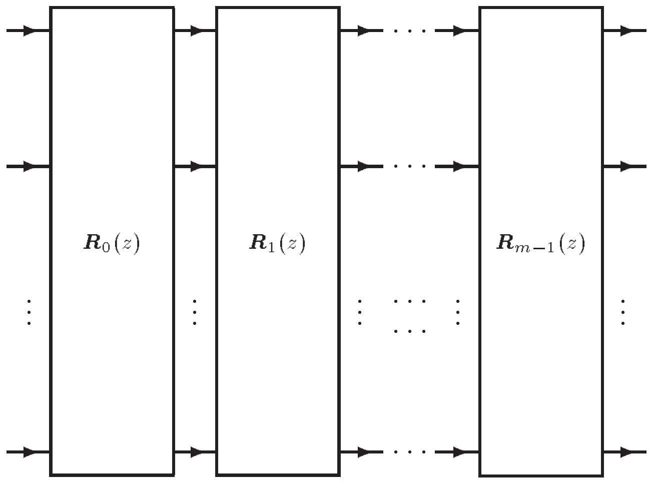


Figure 2.18. *Block cascade realization of synthesis polyphase matrix.*

2.15 QMF Banks and Series Expansions of Signals

Recall the structure of the M-channel QMF bank as shown in Figure 2.19. Assume now that the QMF bank has PR, and therefore, performs an invertible transformation. Mathematically, there are two different ways to interpret the action of such a QMF bank:

1. First, the QMF bank can be interpreted as calculating the coefficients for a series expansion of the discrete-time input signal $x[n]$. The subband signals $y_i[n]$ then represent the coefficients for this series expansion of $x[n]$.
2. Second, the QMF bank can be viewed as computing the coefficients for a series expansion of a continuous-time signal $g(t)$ from the coefficients of a related series expansion of the same signal. In this case, the input sequence $x[n]$ is interpreted as expansion coefficients for the original series, and the subband signals $y_i[n]$ represent the expansion coefficients for the new series expansion.

From a mathematical standpoint, both of these interpretations are equally valid. Which interpretation one chooses often depends on the application at hand. In many cases, however, valuable insights can be gained by considering both viewpoints. For the application of image compression discussed later in this thesis, the author believes that the first interpretation is probably more realistic. In this application, the input sequence $x[n]$ is typically the sampled values of some continuous-time signal (and not expansion coefficients). For this reason, the second interpretation (in which the input sequence is assumed to represent expansion coefficients) is somewhat artificial. Nevertheless, some useful insights can still be gained by considering the second interpretation.

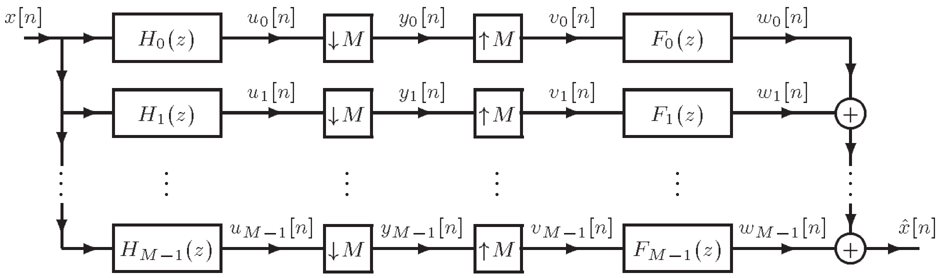


Figure 2.19. *M-channel QMF bank (revisited).*

Consider the M-channel QMF bank shown in Figure 2.19. For the purposes of this discussion, the reader is again reminded that the QMF bank is assumed to have PR. Denote the input to the QMF bank $x[n]$, the subband signals $y_i[n]$, and let $u_i[n]$, $v_i[n]$, and $w_i[n]$ denote the various intermediate signals as shown in the diagram. Further, denote the analysis and synthesis filter impulse responses as $h_i[n]$ and $f_i[n]$, respectively.

The QMF bank takes the input sequence $x[n]$, and transforms it into M new sequences $y_i[n]$ using the analysis filters H_k. Although the number of sequences increases, due to downsampling there is no net growth in the number of samples. We now claim that the subband signals $y_i[n]$ can be viewed as the coefficients of a series expansion of $x[n]$. To see why this is the case, we proceed as follows. First, we express the output of the analysis filters as

$$\begin{aligned} u_i[m] &= x[m] * h_i[m] \\ &= \sum_n x[n] h_i[m-n] \\ &= \langle x[n], h_i[m-n] \rangle \end{aligned}$$

for $i = 0, 1, \ldots, M-1$. From this, we can easily express the subband signals as

$$\begin{aligned} y_i[m] &= u_i[Mm] \\ &= \langle x[n], h_i[Mm-n] \rangle \end{aligned} \tag{2.11}$$

for $i = 0, 1, \ldots, M-1$. Thus, the analysis side of the QMF bank computes the inner products of the input signal $x[n]$ with the analysis basis functions

$$\tilde{\varphi}_k[n] = h_i[Mm-n] \quad \text{for } i = 0, 1, \ldots, M-1,\ m \in \mathbb{Z}$$

where $k = Mm + i$. These basis functions are nothing more than time-reversed shifted M-fold downsampled versions of the analysis filter impulse responses. Now, considering the synthesis side of the QMF bank, we can express the outputs of the upsamplers as

$$v_i[n] = \begin{cases} y_i[n/M] & \text{if } n \text{ is an integer multiple of } M \\ 0 & \text{otherwise} \end{cases}$$

for $i = 0, 1, \ldots, M-1$. From this, the outputs of the synthesis filters are given by

$$\begin{aligned} w_i[n] &= v_i[n] * f_i[n] \\ &= \sum_m v_i[m] f_i[n-m] \\ &= \sum_{\substack{m \\ m|M}} y_i[m/M] f_i[n-m] \\ &= \sum_m y_i[m] f_i[n-Mm] \\ &= \langle y_i[m], f_i[n-Mm] \rangle \end{aligned} \tag{2.12}$$

for $i = 0, 1, \ldots, M-1$. Using (2.12) and (2.11), we can write

$$\begin{aligned} x[n] &= \sum_{i=0}^{M-1} w_i[n] \\ &= \sum_{i=0}^{M-1} \sum_m y_i[m] f_i[n-Mm] \\ &= \sum_{i=0}^{M-1} \sum_m \langle x[m], h_i[Mm-n] \rangle f_i[n-Mm] \\ &= \sum_k \langle x[n], \tilde{\varphi}_k[n] \rangle \varphi_k[n] \end{aligned}$$

where $k = Mm+i$ and $\varphi_k[n] = f_i[n-Mm]$. Thus, we can see that the QMF bank computes a series expansion of $x[n]$ where the synthesis basis functions are $\varphi_k[n]$ and the analysis basis functions are $\tilde{\varphi}_k[n]$. The synthesis basis functions are simply M-shifts of the synthesis filter impulse responses. Since the QMF bank has the PR property, it follows that the analysis and synthesis bases are biorthogonal. In a strict mathematical sense, the bases are only biorthogonal if the reconstruction delay of the QMF bank is zero, although for the purposes of this discussion we need not worry about this. In the more general case, we simply have that shifted versions of the analysis basis functions are biorthogonal to the synthesis basis functions.

Although a QMF bank can be viewed as a structure that computes a series expansion of a discrete-time signal, this is not the only interpretation. In fact, another very useful interpretation exists.

Again, let us consider the M-channel QMF bank of Figure 2.19. Such a QMF bank can also be viewed as a structure that calculates a series expansion of a continuous-time signal. Moreover, this type of expansion corresponds to a single-level M-band wavelet decomposition (or wavelet transform) of a signal. To see why this is so, we proceed as follows. Suppose that solutions to the following two dilation equations exist:

$$\tilde{\varphi}_0(t) = \sum_n h_0[n]\tilde{\varphi}_0(Mt-n) \tag{2.13}$$

$$\varphi_0(t) = \sum_n f_0[n]\varphi_0(Mt-n) \tag{2.14}$$

As a matter of terminology, $\tilde{\varphi}_0(t)$ and $\varphi_0(t)$ are called the analysis and synthesis scaling functions, respectively. In what follows, we assume without loss of generality that both scaling functions have unit norms. That is, we assume

$$\int_{-\infty}^{\infty} \tilde{\varphi}_0(t)dt = 1$$

$$\int_{-\infty}^{\infty} \varphi_0(t)dt = 1$$

Further, let us define the functions

$$\tilde{\varphi}_i(t) = \sum_n h_i[n]\tilde{\varphi}_0(Mt-n) \quad \text{for } i = 1, 2, \ldots, M-1 \tag{2.15}$$

$$\varphi_i(t) = \sum_n f_i[n]\varphi_0(Mt-n) \quad \text{for } i = 1, 2, \ldots, M-1 \tag{2.16}$$

The quantities $\tilde{\varphi}_1(t), \tilde{\varphi}_2(t), \ldots, \tilde{\varphi}_{M-1}(t)$ are called analysis wavelet functions, and the quantities $\varphi_1(t), \varphi_2(t), \ldots, \varphi_{M-1}(t)$ are called synthesis wavelet functions. Since the QMF bank has the PR property, it computes an invertible transform and thus the analysis and synthesis bases are biorthogonal. In other words, we have

$$\langle \tilde{\varphi}_i(t-l), \varphi_k(t-m) \rangle = \delta[i-k]\delta[l-m] \quad \text{for all } i, k, l, m$$

Moreover, as the analysis and synthesis scaling functions are solutions to dilation equations this signal decomposition corresponds to a multiresolution analysis (MRA).

Consider now a continuous-time function $g(t)$ that can be expressed in the form

$$g(t) = \sum_n a_n \varphi_0(t-n) \tag{2.17}$$

Such a function can also be written as

$$g(t) = \sum_{i=0}^{M-1} \sum_n b_{i,n} M^{-1/2} \varphi_i(t/M - n) \tag{2.18}$$

Assume now that we wish to decompose the signal

$$g(t) = \sum_n x[n] \varphi_0(t-n)$$

into an expansion of the form given by (2.18). In other words, we want to represent the signal using rescaled versions of the original basis functions. How can this be accomplished? We begin with the original formula for $g(t)$ and use the fact that we have an MRA to write

$$g(t) = \sum_{i=0}^{M-1} \sum_n x[n] \sum_l \left\langle \varphi_0(t-n), M^{-1/2} \tilde{\varphi}_i(t/M - l) \right\rangle \varphi_i(t/M - l) \tag{2.19}$$

The inner product in the above formula can be computed by using the dilation equation for $\tilde{\varphi}_0(t)$ given by (2.13), the equation for the analysis wavelet functions given by (2.15), and biorthogonality as

$$\begin{aligned}\left\langle \varphi_0(t-n), M^{-1/2} \tilde{\varphi}_i(t/M - l) \right\rangle &= \int_{-\infty}^{\infty} \varphi_0(t-n) M^{-1/2} \sum_k h_i[k] \tilde{\varphi}_0(t - Ml - k) dt \\ &= M^{-1/2} \sum_k h_i[k] \int_{-\infty}^{\infty} \varphi_0(t-n) \tilde{\varphi}_0(t - Ml - k) dt \\ &= M^{-1/2} \sum_k h_i[k] \delta[k - n + Ml] \\ &= M^{-1/2} h_i[n - Ml]\end{aligned} \tag{2.20}$$

Substituting this result in (2.19) and rearranging, we have

$$\begin{aligned} g(t) &= \sum_{i=0}^{M-1} \sum_n x[n] \sum_l M^{-1/2} h_i[n - Ml] \varphi_i(t/M - l) \\ &= \sum_{i=0}^{M-1} \sum_l \left(\sum_n x[n] h_i[n - Ml] \right) \left(M^{-1/2} \varphi_i(t/M - l) \right) \\ &= \sum_{i=0}^{M-1} \sum_l y_i[n] \left(M^{-1/2} \varphi_i(t/M - l) \right) \end{aligned}$$

In other words, the new expansion coefficients that we seek in equation (2.18) are given by the subband signals $y_i[n]$. Thus, a QMF bank can also be seen as a structure for computing series expansions of continuous-time signals.

As a practical matter, we normally use sampled values of some continuous-time signal as direct input to the QMF bank instead of the true series expansion coefficients a_k given by (2.17). Provided the signal $g(t)$ does not change too rapidly with respect to the initial scale of the MRA, the expansion coefficients will be reasonably close to the sampled values of the function. It is important to emphasize, however, that in the case where the true expansion coefficients differ from the sampled values, the continuous-time function being decomposed no longer corresponds to the true original continuous-time signal.

Further to the sample value versus expansion coefficient issue, there is another important fact worth mentioning. Before proceeding, however, we need the following definition:

Definition 2.4 *A continuous-time function $\lambda(t)$ is said to be interpolating if*

$$\lambda(n) = \delta[n] \quad \textit{for all } n \in \mathbb{Z}$$

Suppose we have a continuous-time signal $g(t)$ that is scale-limited and can consequently be represented by the expansion

$$g(t) = \sum_k a_k \varphi_0(M^J t - k) \tag{2.21}$$

where $a_k = \langle g(t), \tilde{\varphi}_0(M^J t - k) \rangle$ and J is an integer. Further, assume that the synthesis scaling function $\varphi_0(t)$ is interpolating. Given this assumption, we now evaluate the function $g(\cdot)$ at M-adic points of the form $M^{-J}n$ to yield

$$\begin{aligned} g(M^{-J}n) &= \sum_k a_k \varphi_0(M^J[M^{-J}n] - k) \\ &= \sum_k a_k \varphi_0(n - k) \\ &= \sum_k a_n \delta[n - k] \\ &= a_n \end{aligned}$$

Thus, if the function $g(\cdot)$ is sampled at points $M^{-J}n$, the resulting sample values are exactly the coefficients a_k of the series expansion given in equation (2.21). Consequently, if the synthesis scaling function is interpolating, one does not introduce any error by using the sample values as expansion coefficients as they are one and the same. As we shall see later, some of the more performant wavelet transforms used for image compression, indeed, have interpolating synthesis scaling functions.

As we have seen in this section, PR QMF banks compute series expansions of signals using basis functions of a particular form. In other words, such a system simply performs a change of basis. This amounts to nothing more than a linear transform. Thus, the PR QMF bank provides a computationally efficient way to implement a particular class of linear transforms.

2.16 Octave-Band Filter Banks and Series Expansions of Signals

Having seen the relationship between PR QMF banks and signal expansions, we are now in a position to consider the relationship between octave-band filter banks and signal decompositions.

An octave-band filter bank can be viewed as a system that computes a series expansion of the input signal $x[n]$. This follows directly from the single QMF bank case studied in the previous section. The only difference with a tree structure is that the series expansion is performed recursively. This, of course, results in different basis functions from the single QMF bank case, but the idea is fundamentally the same.

It is particularly interesting to note what happens to the analysis and synthesis basis functions as the number of levels in the tree increases. As the tree depth increases, the basis functions associated with the last level in the tree more and more closely resemble sampled versions of the scaling and wavelet functions associated with the QMF bank. In fact, if the axes are rescaled appropriately, in the infinite limit the discrete-time basis functions associated with the last level of the tree become continuous-time functions and converge to the scaling and wavelet functions. This argument, of course, presupposes that the scaling and wavelet functions exist. Otherwise, this iterative process will not converge.

In many cases, the tree depth need not be very large in order to begin obtaining basis functions that closely resemble sampled versions of the scaling and wavelet functions. Typically, a depth of three or four is sufficient for many sets of filters used in practice. In this sense, the scaling and wavelet functions are very important as they provide a good indication of the shape of the basis functions used in the series expansion of the discrete-time input signal $x[n]$.

We can also view an octave-band filter bank as computing a series expansion of a continuous-time signal. In particular, such a filter bank computes a wavelet decomposition of a signal. Again, this follows directly from the single QMF bank case as described in the previous section. Each time the lowpass coefficients are successively decomposed, some of the basis functions in the series expansion are replaced with rescaled versions. As we descend deeper into the tree, the corresponding basis functions become increasingly stretched.

In an earlier section, it was stated without justification that an octave-band filter bank computes a wavelet transform. This claim has now been substantiated.

2.17 Appropriate Basis Selection

The series expansion interpretations of filter banks are very important. Typically, linear transforms are used to obtain alternative representations of signals that are more convenient for the application at hand. In signal compression applications, we seek more compact representations of signals. To obtain more compact representations, however, it is necessary to choose a set of basis functions that can efficiently represent a signal. That is, we seek a basis that allows us to accurately represent a signal with as few expansion coefficients as possible. As one might expect, better results are achieved when the synthesis basis functions are similar in shape to the function being represented. For example, if a signal is very smooth, it is advantageous to employ very smooth synthesis basis functions. Similarly, if a signal has many sharp transitions, the synthesis basis functions should also have a similar behavior.

When octave-band filter banks are employed as in the case of wavelet transforms, the shape of the scaling and wavelet functions is of interest. These functions provide a good indication of the type of signal that can be most efficiently represented with a particular transform. As we have seen, the choice of basis functions is determined by the QMF bank filter coefficients. For this reason, the filter coefficients must be chosen appropriately in order to obtain effective transforms.

2.18 Multidimensional Signals

The signals of interest in this thesis are images. Since images are two-dimensional signals, clearly two-dimensional transforms are needed. Fortunately, two-dimensional transforms can be easily constructed from one-dimensional ones. In the case of images, we can simply apply a one-dimensional transform first to the rows and then to the columns of an image (or vice versa). This amounts to nothing more than forming a two-dimensional transform using tensor products of the basis functions of the one-dimensional transform. To express this in mathematical terms, suppose we have a one-dimensional transform with analysis and synthesis basis functions $\tilde{\varphi}_i$ and φ_i, respectively. If we apply this one-dimensional transform to each dimension of a two-dimensional signal separately, this is equivalent to expressing the signal $g(x, y)$ as

$$g(x, y) = \sum_i \sum_k \langle g(x, y), \tilde{\varphi}_{i,k}(x, y) \rangle \, \varphi_{i,k}(x, y)$$

where the basis functions $\tilde{\varphi}_{i,k}(x, y)$ and $\varphi_{i,k}(x, y)$ are given by

$$\begin{aligned} \tilde{\varphi}_{i,k}(x, y) &= \tilde{\varphi}_i(x) \tilde{\varphi}_k(y) \\ \varphi_{i,k}(x, y) &= \varphi_i(x) \varphi_k(y) \end{aligned}$$

By construction, a transform of this form can always be expressed as two separate one-dimensional transforms. Such a transform is said to be separable. Although separable transforms have axial dependencies that can sometimes be undesirable, they are more efficient to compute than their nonseparable counterparts. For this reason, separable transforms are most commonly used today, and this thesis considers only such transforms. For more details on transforms for multidimensional signals, the reader is referred to [56] and [17].

2.19 Finite-Length Signals

Until now, we have ignored the fact that images are usually only defined on some finite region. That is, such signals are not infinite in extent. Unfortunately, such finite-length signals introduce problems when being processed by a filter bank. The difficulty is how to handle filtering at the boundaries of the signal. Since the outputs of the filters (in the filter bank) depend on past and/or future sample values, once we get close enough to a signal edge the filters need sample values that are not defined. There are two general approaches that can be used to address this problem. The first approach is to extend the signal so it is defined for all possible sample indices. Such an approach is referred to as an extension method. The second approach is to change the filters at the boundaries of the signal in order to avoid needing undefined sample values. This type of approach is referred to as boundary filtering. In what follows, we consider two commonly used extension methods (i.e., periodic and symmetric extension) and make a few comments concerning boundary filtering methods.

Periodic extension is the simplest approach to handling finite-length signals. With this method, the original finite-length signal is repeated indefinitely to yield a periodic signal. If the original signal is defined over N sample indices, periodic extension results in an infinite-length signal of period N. Assume now that we process the extended signal with an M-channel QMF bank. Provided N is evenly divisible by M, the subband signals will all have period N/M. Therefore, only N/M samples are required to completely characterize each subband signal. Therefore, only N samples in total are required to completely characterize all of the subband signals. Thus, we have a transform that maps N values to N new values. That is, there is no net growth in the number of nonredundant samples. This property is desirable in many applications especially signal compression. Unfortunately, periodic extension has the potential disadvantage that the extended signal may possess very large jumps at the splice points between periods. Moreover, if N is not evenly divisible by M, periodic extension cannot be employed as described above. Of course, we can always pad the signal with extra samples so that divisibility is achieved, but doing this is often not desirable.

Symmetric extension is another relatively simple approach to handling finite-length signals. With this approach, the original signal is extended so that it is both symmetric and periodic. A signal can either be symmetric about one of its samples or about a point midway between two samples. These two cases are referred to as whole-sample symmetry (WS) and half-sample symmetry (HS), respectively, and examples of both are given in Figure 2.20. The centers of symmetry of periodic signals always come in pairs. If the period of the signal is even, both symmetry centers will be WS. If the period is odd, one will be WS and the other HS.

There is more than one way in which to symmetrically extend a signal. The main difference is in the number of times the first and last sample of the original signal are repeated in the extended signal. Here, we will only consider (1,1)-symmetric extension. In this case, the first and last sample of the original signal appear once (i.e., they are not repeated). An example of this type of extension is depicted in Figure 2.21. Notice that in the extended signal, the first and last sample appear only once in each period. Given a signal of length N, $(1,1)$-symmetric extension always yields a WS signal of period $2N-2$. If a signal extended in this way is used as the input to a 2-channel QMF bank with certain properties, the subband signals will all be symmetric and $(N-1)$-periodic. Regardless of the parity of N (i.e., whether N is odd or even), it is always true that only N samples are required in total to completely characterize the subband signals. Thus, such a transform maps

N values to N new values. In other words, the transform is not expansive. There is no net growth in the number of samples required to represent the original signal. Also, note that this holds regardless of the parity of N. Consequently, in the case of 2-channel QMF banks, symmetric extension yields nonexpansive transforms for input signals of arbitrary length. As we have seen, however, periodic extension does not always yield nonexpansive transforms. Also, symmetric extension has the added advantage that it does not introduce any jumps in the extended signal.

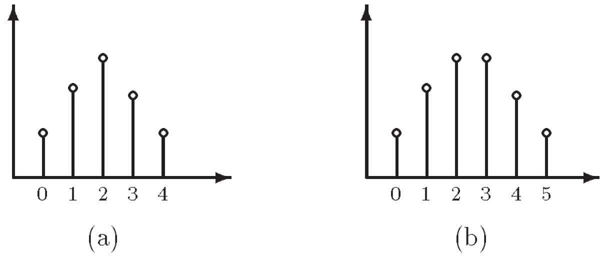


Figure 2.20. *Types of signal symmetries. (a) Whole-sample symmetry. (b) Half-sample symmetry.*

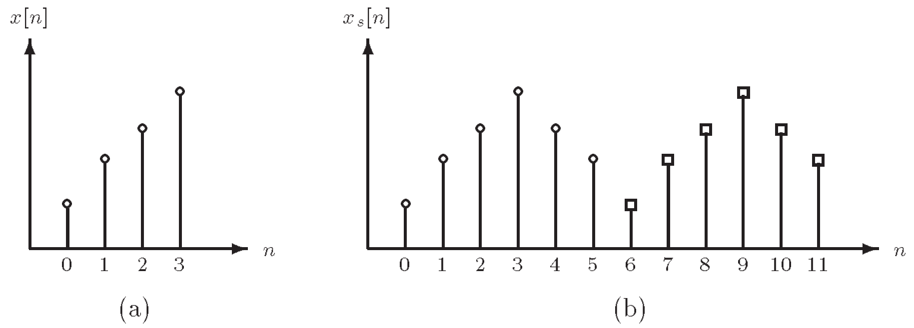


Figure 2.21. *(1,1)-symmetric extension example. (a) Original signal. (b) Extended signal.*

A much more sophisticated approach to handling finite-length signals is boundary filtering (i.e., time-varying filter banks). In this scheme, the filter bank actually processes a finite-length signal. To avoid needing undefined sample values, the filtering structure and/or filters are changed at the signal boundaries. Unfortunately, boundary filtering is inherently much more complicated than either periodic or symmetric extension. As a consequence, the software algorithms or circuits required to implement such schemes tend to be more complex and costly.

For more information on handling finite-length signals the reader is referred to [8], [7], [30], [33], [25], and [20].

2.20 Summary

In this chapter, we began by introducing the fundamentals of multirate systems. We studied the two key building blocks of such systems: the downsampler and upsampler. Then we proceeded to examine the time-domain, z-domain, and frequency-domain effects of the downsampling and upsampling operations. Next, the noble identities were introduced as a means for interchanging upsampling/downsampling and filtering operations. The polyphase form of a filter was also presented as a convenient theoretical tool in addition to offering a means for efficiently implementing filtering in a multirate framework.

After considering multirate systems in some depth, filter banks were introduced, and in particular, a type of multirate filter bank called a QMF bank was studied in detail. Next, the PR property was defined in the context of QMF banks, and necessary and sufficient conditions for PR were given. Also, the polyphase form of a QMF bank was discussed as a theoretical tool and a means for efficiently implmenting such filter banks. Next, a particular type of tree-structured filter bank called the octave-band filter bank was described.

Having studied QMF banks and octave-band filter banks, the relationship between such filter banks and linear series expansions of signals were discussed. In so doing, we showed that under certain conditions these types of filter banks are associated with wavelet decompositions. More specifically, a QMF bank and octave-band filter bank compute single-level and multi-level wavelet decompositions of a signal, respectively.

The chapter concluded by briefly examining techniques for handling multidimensional signals and finite-length signals. The former is handled by using separable transforms, and the latter by signal extension and boundary filtering methods.

2.21 Conclusions

All of the results presented in this chapter are well known. Perhaps, one distinguishing feature of this presentation is that it chooses to discuss both type 1/2 and type 3/4 polyphase forms of QMF banks together. Often, for reasons of convenience, authors prefer to discuss only one of the two commonly-used polyphase forms. Also, although for the most part, the theorems presented in this chapter are well-known results, the author has not seen some of them presented in quite the same fashion. For example, the author feels that his presentation of Theorem 2.2 is somewhat more convenient than the way it is often expressed (e.g., [54]). Notice the appearance of matrices of the form

$$\begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^K \quad \text{and} \quad \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^K$$

Matrices of this form appear in many places in this thesis, and lead to more compact notation than the alternatives that the author has seen in the available literature.

Any sufficiently advanced technology is indistinguishable from magic.
—Arthur C. Clarke

Chapter 3

Reversible Transforms

> A programmer is a person who passes as an exacting expert on the basis of being able to turn out, after innumerable punching, an infinite series of incomprehensive answers calculated with micrometric precisions from vague assumptions based on debatable figures taken from inconclusive documents and carried out on instruments of problematical accuracy by persons of dubious reliability and questionable mentality for the avowed purpose of annoying and confounding a hopelessly defenseless department that was unfortunate enough to ask for the information in the first place.
>
> —IEEE Grid newsmagazine

3.1 Introduction

Reversible transforms are useful for many lossless and hybrid lossy/lossless signal coding applications. In particular, such transforms are especially well suited for reversible embedded image compression, the application of interest in this thesis. As reversible transforms are a key component in reversible embedded image compression systems, this chapter studies such transforms in depth.

This chapter begins by introducing the concept of reversibility and the motivation for the use of transforms with this property. Then, we discuss reversible transforms which are generally nonlinear, and are usually designed to approximate linear transforms with desirable properties. Techniques for handling multidimensional and finite-length signals are briefly revisited in the context of nonlinear reversible transforms.

Next, lifting is presented as a general framework for the design and implementation of invertible transforms. We then examine how lifting can be used in conjunction with M-band subband transforms. This leads to a discussion of the lifting realization of M-channel PR QMF banks. Next, we show how the lifting realization leads naturally to reversible transforms. Numerous practical issues relating to reversible transforms are also discussed in some detail.

This chapter introduces some new results relating to the lifting-based design of reversible transforms. In our discussion of the lifting realization of QMF banks, we consider the M-band case and also for completeness both type 1/2 and type 3/4 polyphase decompositions.

3.2 Reversible Transforms

Although any non-singular linear transform is invertible, this invertibility often depends on the fact that the transform is calculated using exact arithmetic. In practice, however, finite-precision arithmetic is usually employed, and such arithmetic is inherently inexact due to errors introduced by rounding. Unfortunately, transforms that are invertible in exact arithmetic are often not invertible in finite-precision arithmetic. Sometimes, however, it is possible to introduce finite-precision arithmetic without destroying invertibility. A transform that is invertible in finite-precision arithmetic is said to be reversible.

Reversible transforms are ideally suited to applications in which it is undesirable to employ transforms that can result in information loss, even when the transforms are computed using finite-precision arithmetic. For this reason, transform reversibility is often a requirement in applications such as lossless and hybrid lossy/lossless signal compression.

3.3 Reversible Transforms from Linear Transforms

Countless design techniques for linear wavelet/subband transforms have been developed over the years, and through the use of these methods many transforms with a variety of desirable properties have been produced. Unfortunately, most of these transforms lack reversibility, and in some applications this property is often a requirement. Although we could simply limit ourselves to using the small subset of linear transforms that are reversible, such an approach is extremely restrictive. Moreover, most linear transforms that are reversible are not practically useful. In order to exploit the large body of nonreversible linear transforms, a means for constructing reversible versions of these transforms is needed.

In principle, the idea behind creating a reversible version of a nonreversible linear 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 invertibility. Due to the use of quantization, the resulting reversible transform is generally nonlinear and only serves to approximate the linear transform from which it was derived. The degree to which the reversible transform is able to successfully approximate its parent transform is extremely important. If the approximation error is small, all is well. If, however, the reversible transform fails to mimic the behavior of its parent transform, the desirable properties of the parent transform will likely be lost and poor results will be obtained when the new transform is used. For this reason, the approximation characteristics of a reversible transform are a key consideration in the design process. Although the principles underlying reversible transform construction are easily stated, generating useful transforms is by no means a straightforward task.

In the past, reversible transforms have been developed largely by ad hoc methods that are difficult to generalize. Consequently, reversible transforms tended to be difficult to design, and few good transforms were known. In spite of these difficulties, however, some good transforms were developed. The S transform [28], RTS transform [60], and S+P transform [36] are three examples of such transforms. These are all nonlinear approximations to well-known linear transforms, and were devised using ad hoc methods. Fortunately, a systematic method for constructing reversible transforms has been recently proposed. This design approach is based on lifting and is discussed at length later in this chapter.

3.4 Multidimensional Signals

In this thesis, the signals of interest are images. Since such signals are two-dimensional, clearly two-dimensional transforms are needed. As is the case with linear transforms, we can construct two-dimensional reversible transforms from one-dimensional ones. In the case of images, we first apply a one-dimensional transform along the rows of the image and then along the columns (or vice versa). This, in effect, creates a two-dimensional transform. There is, however, one subtlety. With linear transforms, the order in which rows and columns are transformed does not matter. Due to linearity, the order can be switched and the transform remains the same. Reversible transforms, however, are generally nonlinear. As a result, with reversible transforms the order in which the rows and columns are transformed is important. If the forward transform acts first on rows and then on columns, the inverse transform must act first on columns and then on rows. That is, the inverse transform must operate on rows and columns in the reverse order from that used in the forward transform.

3.5 Finite-Length Signals

In the context of linear transforms, three methods were discussed for handling finite-length signals, namely, periodic extension, symmetric extension, and boundary filtering (see Section 2.19). All of these approaches are also applicable in the context of nonlinear reversible transforms. As one might expect, however, there are a number of caveats introduced by the nonlinearity of the systems involved.

Periodic extension is the most straightforward approach to handling finite-length signals. With some basic assumptions on the filtering structure used, if the filters are all of the FIR type, periodic extension can be employed. That is, despite the nonlinearities, the system will still have a periodic steady-state response to a periodic input. If IIR filters are employed, however, many problems arise. This is a consequence of the fact that the behavior of an IIR filter depends not only on its input, but also on its output (past and/or future). Due to the nonlinearities in the feedback paths, it is not clear that such a system will have a periodic steady-state response to a periodic input (with the same period in both cases). Moreover, even if such a steady-state response exists, finding it is very problematic. This involves solving for the steady-state conditions of the IIR filters. Unfortunately, in order to do this, one would have to solve a system of highly nonlinear equations that grows linearly with the length of the signal.

Symmetric extension also involves making the input signal periodic. As a consequence, one is faced with the same problems mentioned above when IIR filters are used. The issue of IIR filters aside, other complications exist. Although a nonlinear reversible transform may approximate a symmetry-preserving linear transform, the approximation error is inevitably nonzero. In most cases, it is highly likely that this error will lead to a reversible transform that does not preserve symmetry. To maintain the symmetry-preserving property of a transform when making it reversible requires very clever cancellation of quantization error. Not only must quantization error be cancelled in such a way that invertibility is not affected, but it must also cancel in such a way as to not affect the symmetry-preserving nature of the transform. Although this is generally difficult to do, it is certainly possible and one approach is discussed later.

As mentioned previously, the design of transforms based on boundary filtering (i.e., time-varying filter banks) is not considered in this thesis. However, one example of a transform employing this strategy (i.e., the S+P transform) is presented in a later section.

3.6 S Transform

The classic example of a reversible transform is the Sequential (S) transform as inspired by Lux [28]. Since its first introduction, the S transform has become quite popular for lossless signal compression (especially lossless image compression) and has been discussed by many (e.g., [60], [36], [38]).

The S transform is reversible and maps integers to integers. Several slightly different definitions of this transform exist in the literature. Here, we consider one of the more commonly used definitions. The forward transform splits the input signal $x[n]$ into the lowpass component $s[n]$ and highpass component $d[n]$ as follows:

$$s[n] = \left\lfloor \tfrac{1}{2}(x[2n] + x[2n+1]) \right\rfloor \tag{3.1a}$$

$$d[n] = x[2n] - x[2n+1] \tag{3.1b}$$

The inverse transform combines the lowpass and highpass components to yield the original signal as follows:

$$x[2n] = s[n] + \left\lfloor \tfrac{1}{2}(d[n] + 1) \right\rfloor \tag{3.2a}$$

$$x[2n+1] = x[2n] - d[n] \tag{3.2b}$$

The filtering structures associated with the forward and inverse transforms are shown in Figures 3.1(a) and 3.1(b), respectively. In these diagrams, the blocks labelled Q_T are quantizers with the transfer characteristic $Q_T(x) = \lfloor x \rfloor$. Although the transform, as defined, assumes the signal $x[n]$ to be infinite in length, finite-length signals can be easily handled by using periodic extension.

The development of the S transform is based on two key observations. First, any two numbers can be unambiguously determined from their sum and difference. And second, the sum and difference of any two integers always have the same parity (i.e., the sum and difference are either both odd or both even). The first observation leads us to formulate a transform that simply computes pairwise

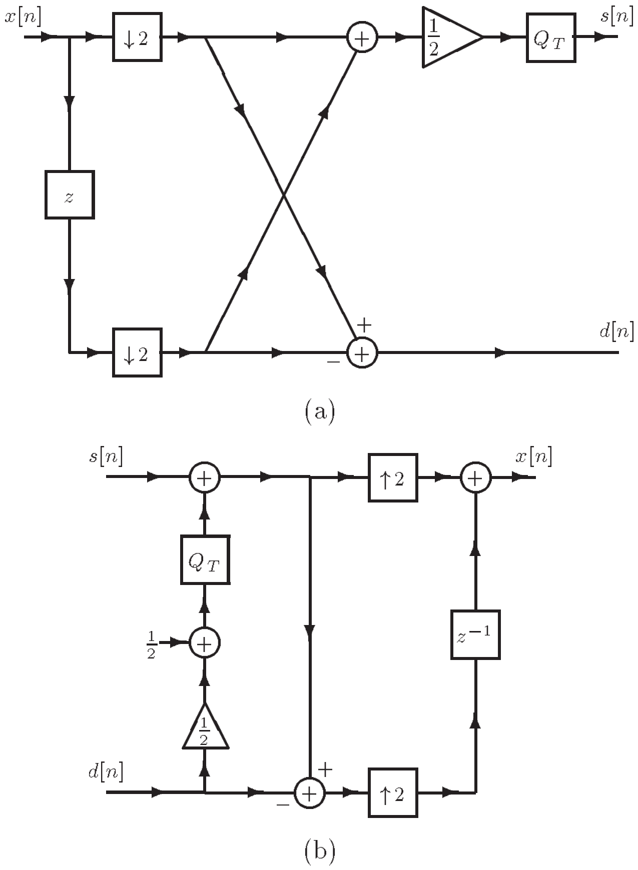


Figure 3.1. *Filtering structure for S transform. (a) Forward transform. (b) Inverse transform.*

sums and differences of samples. Although such a transform is reversible, we can use the second observation to further refine the transform. Because the sum and difference of two integers have the same parity, the least significant bit of either the sum or difference is redundant and may be discarded without losing information. Thus, we can choose to divide the sum by two and discard the fractional part of the result without information loss. This is exactly what the S transform does. Moreover, this explains why the S transform is invertible.

Although the above explanation is probably more enlightening than a formal proof of invertibility, such a proof is almost trivial. To assist in the proof, we rely on the following lemma:

Lemma 3.1 *For any two integers a and b, the following identities hold:*

$$\left\lfloor \tfrac{1}{2}(a+b) \right\rfloor + \left\lfloor \tfrac{1}{2}(a-b+1) \right\rfloor = a, \quad \left\lfloor \tfrac{1}{2}(b+a) \right\rfloor - \left\lfloor \tfrac{1}{2}(b-a) \right\rfloor = a$$

Proof. See Appendix A.

We are now in a position to show that the S transform is invertible. Let $\hat{x}[n]$ denote the reconstructed signal. We start with equation (3.2a), and then use equation (3.1) and the above lemma to yield

$$\begin{aligned}\hat{x}[2n] &= s[n] + \left\lfloor \tfrac{1}{2}(d[n]+1) \right\rfloor \\ &= \left\lfloor \tfrac{1}{2}(x[2n]+x[2n+1]) \right\rfloor + \left\lfloor \tfrac{1}{2}(x[2n]-x[2n+1]+1) \right\rfloor \\ &= x[2n]\end{aligned}$$

Thus, the even samples are reproduced exactly. Next, we look at equation (3.2b). Using equation (3.1) and the previous result, we have

$$\begin{aligned}\hat{x}[2n+1] &= \hat{x}[2n] - d[n] \\ &= x[2n] - d[n] \\ &= x[2n] - x[2n] + x[2n+1] \\ &= x[2n+1]\end{aligned}$$

Thus, the odd samples are also reproduced exactly. Hence, the S transform is invertible.

The S transform is a nonlinear approximation to a scaled version of the Haar transform. The Haar transform itself is one of the simplest 2-band subband transforms and corresponds to a 2-channel QMF bank having analysis and synthesis filters with transfer functions $H_k(z)$ and $F_k(z)$, respectively, where

$$H_0(z) = \tfrac{1}{2}(1+z), \quad H_1(z) = 1 - z, \quad F_0(z) = 1 + z, \quad F_1(z) = \tfrac{1}{2}(1-z)$$

This transform is a scaled version of an orthogonal transform. Therefore, the coefficients of the S transform must be weighted on a per subband basis in order to approximate an orthogonal transform.

It is worth noting that the lowpass signal $s[n]$ has the same dynamic range as the original signal $x[n]$. This property is particularly useful in pyramidal schemes in which the lowpass channel is successively decomposed. The S transform has very low computational complexity relative to other subband transforms. Unfortunately, it is also somewhat primitive, and has numerous shortcomings. It is known for producing undesirable blocking artifacts when used for lossy image compression, a behavior inherited from its parent linear transform. In addition, the S transform is also not particularly well suited to smoothly varying signals.

Unfortunately, the ideas on which the S transform is based do not generalize to transforms using more complicated relationships than simple pairwise sums and differences. Consequently, the S transform does not provide any further insight into how other classes of reversible transforms might be constructed.

3.7 Lifting

A new philosophy for the design and implementation of invertible transforms, called lifting, was first proposed by Sweldens in [51]. Lifting was initially conceived as an alternative method for constructing biorthogonal 2-band wavelet transforms. Since that time, numerous other papers have been written on lifting (e.g., [48], [50], [49], [15]), and it has proven to be useful in a number of other contexts as well. In fact, as will be seen later, lifting can be used for reversible transform design.

In its 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 into the original sequence. Suppose that we have a sequence which 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 modify the values in another sequence. This process may be repeated a number of times involving different pairs of sequences each time.
3. Scale. A final normalization is applied to each sequence.

The input to the inverse transform is now several sequences. Like the forward transform, the inverse transform is also computed in three stages:

1. Scale.
2. Predict.
3. Join.

The join stage corresponds to the sequences being combined to yield a single sequence. Each stage in the inverse transform calculation simply undoes the effects of the corresponding stage in the forward transform calculation. This type of computational structure conveniently leads to invertible transforms. As a matter of terminology, a transform calculated using this framework is called a lifted transform.

3.8 Lifted Subband Transforms

Subband transforms can be easily cast in a lifting framework. Here, we consider the large class of subband transforms that can be computed using an M-channel PR QMF bank with FIR filters. This class of transforms includes, as a subset, all wavelet transforms associated with finitely-supported scaling/wavelet functions. As it turns out, implementing subband transforms as lifted transforms requires nothing more than implementing the transform using a QMF bank in polyphase form with the polyphase matrices realized in a special way.

In [15], Daubechies and Sweldens consider the case of 2-band wavelet transforms computed by a QMF bank in type 3/4 polyphase form, but here we extend the concepts to general M-band transforms and also consider both type 1/2 and type 3/4 polyphase decompositions. Some of the ideas presented here are intimately related to those discussed by Kalker and Shah in [23] and [24]. In these works, ladder network realizations of polyphase matrices are studied in depth. Since the primary focus of these papers was not lifting and the construction of reversible transforms, there are still some new ideas in our treatment of the subject.

In the next section, the lifting realization of a QMF bank is described in detail. A lifted subband transform is simply a transform calculated using such a QMF bank implementation.

3.9 Lifting Realization of a QMF Bank

Suppose we have an M-channel PR QMF bank with FIR filters. Denote the transfer functions of the QMF bank's analysis and synthesis filters as $H_k(z)$ and $F_k(z)$, respectively. Further, assume that the system is represented in either type 1/2 or type 3/4 polyphase form with analysis and synthesis polyphase matrices $\boldsymbol{E}(z)$ and $\boldsymbol{R}(z)$, respectively. In a lifting realization, a QMF bank is implemented in its polyphase form. The distinguishing characteristic of this realization is the type of network used to realize the polyphase matrices.

Fundamental to the idea of the lifting realization is the lifting factorization which decomposes a matrix into two types of factors:

1. Scaling (Type S). This type of matrix is square ($M \times M$), diagonal, and corresponds to an elementary row/column operation that multiplies row/column i by the quantity k. Such a matrix has all diagonal elements equal to one except the (i, i) entry which is k and all off-diagonal elements equal to zero. This type of matrix is completely characterized by the pair (i, k) and is denoted $\mathcal{S}_M(i; k)$. In cases where the size of the matrix is clear from the context, the subscript M is omitted.
2. Adding (Type A). This type of matrix is square ($M \times M$) and corresponds to an elementary row/column operation that adds k times row j to row i or adds k times column i to column j. Such a matrix has all ones on the diagonal, k at position (i, j), and all other entries zero. This type of matrix is completely characterized by the triple (i, j, k) and is denoted $\mathcal{A}_M(i, j, ; k)$. Again, the subscript M may be omitted in cases where the size of the matrix is clear from the context.

More specifically, the lifting factorization decomposes a matrix into zero or more constant Type S factors that either premultiply or postmultiply zero or more Type A factors.

In a lifting realization, the polyphase matrices are implemented directly from their lifting factorizations. More specifically, the analysis polyphase matrix $\boldsymbol{E}(z)$ is decomposed using a lifting factorization as

$$\boldsymbol{E}(z) = \boldsymbol{S}_{\sigma-1} \cdots \boldsymbol{S}_1 \boldsymbol{S}_0 \boldsymbol{A}_{\lambda-1}(z) \cdots \boldsymbol{A}_1(z) \boldsymbol{A}_0(z) \tag{3.3}$$

where the $\boldsymbol{S}_i$ are Type S elementary matrices with all constant entries and the $\boldsymbol{A}_i(z)$ are Type A elementary matrices which can depend on z. In a lifting realization, the decomposition of the synthesis polyphase matrix $\boldsymbol{R}(z)$ is completely determined by the decomposition used for the analysis polyphase matrix $\boldsymbol{E}(z)$ and is given by

$$\boldsymbol{R}(z) = \boldsymbol{A}_0^{-1}(z) \boldsymbol{A}_1^{-1}(z) \cdots \boldsymbol{A}_{\lambda-1}^{-1}(z) \boldsymbol{S}_0^{-1} \boldsymbol{S}_1^{-1} \cdots \boldsymbol{S}_{\sigma-1}^{-1} \tag{3.4}$$

Furthermore, this choice of decomposition for $\boldsymbol{R}(z)$ implies that $\boldsymbol{R}(z) = \boldsymbol{E}^{-1}(z)$. Note that $\mathcal{A}^{-1}(i,j;k) = \mathcal{A}(i,j;-k)$ and $\mathcal{S}^{-1}(i;k) = \mathcal{S}(i;k^{-1})$. That is, the inverse of a Type A matrix is another Type A matrix, and the inverse of a Type S matrix is another Type S matrix. Thus, in equation (3.4) the $\boldsymbol{A}_i^{-1}(z)$ are Type A matrices and the $\boldsymbol{S}_i^{-1}$ are Type S matrices. Hence, the decomposition given for $\boldsymbol{R}(z)$ is, in fact, a lifting factorization of the matrix. Moreover, each pair of corresponding matrices $\boldsymbol{A}_i(z)$ and $\boldsymbol{A}_i^{-1}(z)$ are identical except for one element which differs only in sign. Similarly, each pair of corresponding matrices $\boldsymbol{S}_i$ and $\boldsymbol{S}_i^{-1}$ are identical except for one element which differ in a reciprocal relationship.

From the definition of the lifting realization, it follows that such a realization exists if and only if a lifting factorization of $\boldsymbol{E}(z)$ exists and $\boldsymbol{R}(z) = \boldsymbol{E}^{-1}(z)$. We shall revisit this issue later, but for the time being it suffices to say that any QMF bank can be made to satisfy these conditions through an appropriate normalization of its analysis and synthesis filters.

The decompositions in (3.3) and (3.4) correspond to block cascade realizations of the analysis and synthesis polyphase matrices, respectively. Since the QMF bank has M channels, each filtering block has M inputs and M outputs and is characterized by an $M \times M$ matrix. Each of the Type A factors corresponds to a block that adds a filtered version of a signal in one channel to a signal in another channel. This corresponds to a single ladder step in a ladder network as depicted in Figure 3.2. To simplify the diagram only the ith and jth inputs and outputs are shown. All other inputs pass directly through to their corresponding outputs unchanged. Each of the Type S factors corresponds to a block that scales the signal in a single channel. Such a scaling unit is shown in Figure 3.3. Only the ith input and output are shown as all other inputs pass directly through to their corresponding outputs without modification. Since the lifting factorization consists of only Type A and Type S factors, this decomposition yields a ladder structure with some additional scaling elements. The ladder structure is followed by a scaling on the analysis side and preceded by a scaling on the synthesis side of the filter bank. Due to the similarities between the factors in both of these decompositions, the structures used to perform the analysis and synthesis filtering possess a certain degree of symmetry. This symmetry has important consequences as will become evident later.

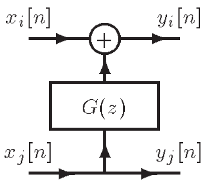


Figure 3.2. *Ladder step.*

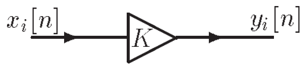
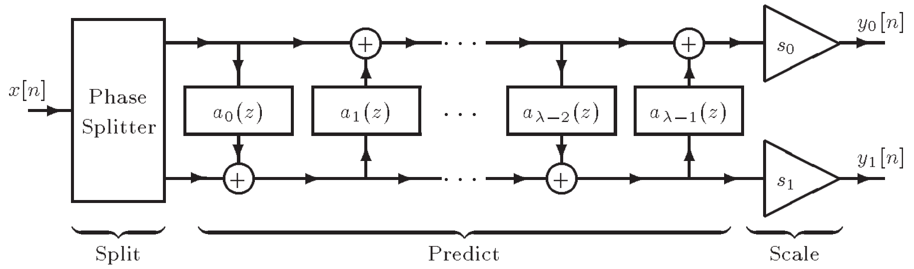


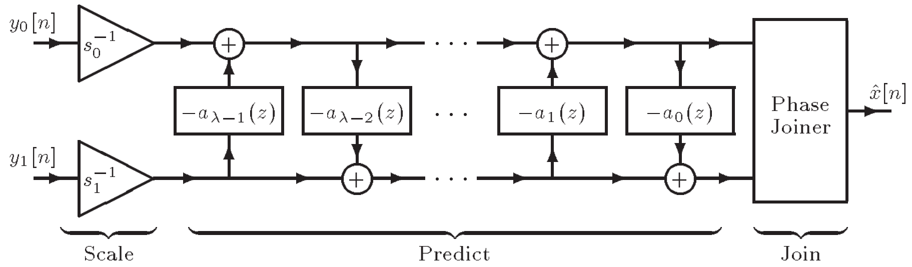
Figure 3.3. *Scaling unit.*

The resulting structure for the general 2-channel case is shown in Figure 3.4 (where some of the $a_i(z)$ may be zero). Notice the symmetry between the forward and inverse transform structures. By inspection, it is obvious that the filter bank has PR. The synthesis side simply undoes step-by-step each operation performed on the analysis side.

Now the correspondence between lifting and the lifting realization of a QMF bank becomes



(a)



(b)

Figure 3.4. *General two-band lifting structure. (a) Analysis side (forward transform). (b) Synthesis side (inverse transform).*

apparent. Indeed, we have a lifted transform. The analysis side of the QMF bank computes the forward transform where:

- The decomposition of the input signal into its polyphase components corresponds to the split stage in the calculation of a forward transform using lifting.
- The ladder network portion of the analysis polyphase matrix realization corresponds to the predict stage in the calculation of a forward transform using lifting.
- The scaling portion of the analysis polyphase matrix realization corresponds to the scale stage in the calculation of a forward transform using lifting.

The synthesis side of the QMF bank calculates the inverse transform where:

- The scaling portion of the synthesis polyphase matrix realization corresponds to the scale stage in the calculation of an inverse transform using lifting.
- The ladder network portion of the synthesis polyphase matrix realization corresponds to the predict stage in the calculation of an inverse transform using lifting.
- The recombination of the various polyphase components that yields the reconstructed signal corresponds to the join stage in the calculation of an inverse transform using lifting.

In the M-channel case, the lifting realization is completely analogous to the 2-band case, and has the general form shown in Figure 3.5. The $\boldsymbol{S}_i$ and $\boldsymbol{S}_i^{-1}$ blocks corresponds to scaling operations.

The $\boldsymbol{A}_i(z)$ and $\boldsymbol{A}_i^{-1}(z)$ blocks correspond to ladder steps. Moreover, the scaling and ladder steps on the analysis side are simply the inverses of the scaling and ladder steps on the synthesis side.

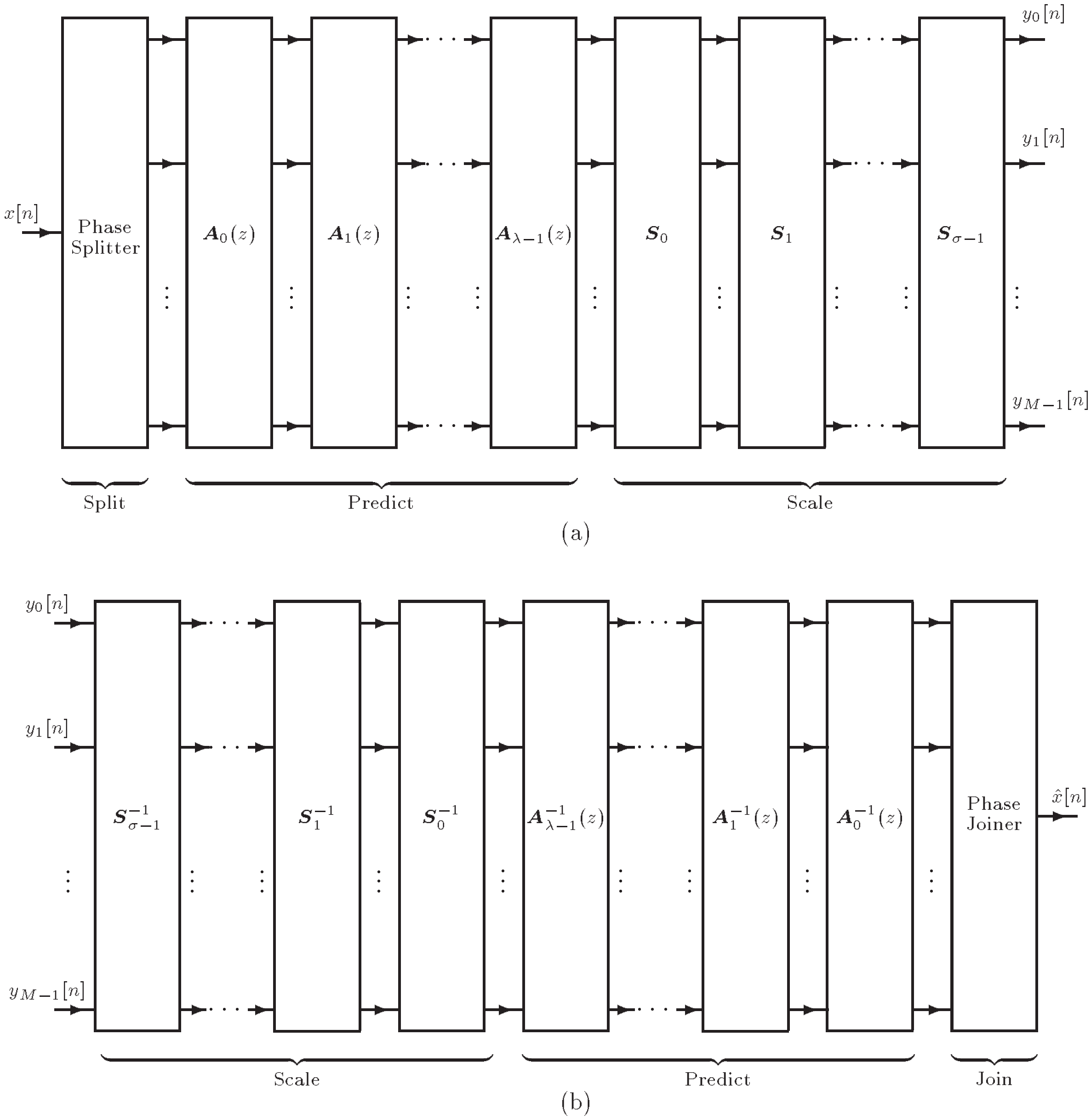


Figure 3.5. *General M-band lifting structure. (a) Analysis side (forward transform). (b) Synthesis side (inverse transform).*

Having decided to use a lifting realization, we must now concern ourselves with the existence of such a realization. That is, can any QMF bank be implemented using lifting? To answer this question, we must revisit the conditions for existence stated earlier. That is, a QMF bank can be realized using lifting if and only if the following conditions are satisfied:

$$\text{a lifting factorization of } \boldsymbol{E}(z) \text{ exists} \tag{3.5}$$

$$\boldsymbol{R}(z) = \boldsymbol{E}^{-1}(z) \tag{3.6}$$

Assuming that the realization exists, its uniqueness is also of concern.

First, let us consider condition (3.5). That is, what conditions must hold in order for the analysis polyphase matrix $\boldsymbol{E}(z)$ to have a lifting factorization? To help us answer this question, we rely on the following theorem:

Theorem 3.2 *Any $M \times M$ Laurent polynomial matrix $\boldsymbol{U}(z)$ admits a factorization of the form*

$$\boldsymbol{U}(z) = \boldsymbol{S}_0(z)\boldsymbol{S}_1(z)\cdots\boldsymbol{S}_{Q-1}(z)\boldsymbol{A}_0(z)\boldsymbol{A}_1(z)\cdots\boldsymbol{A}_{P-1}(z)$$

where the $\boldsymbol{S}_i(z)$ are Type S elementary matrices and the $\boldsymbol{A}_i(z)$ are Type A elementary matrices and all factors are Laurent polynomial matrices. For any particular set of $\boldsymbol{S}_i(z)$, a set of $\boldsymbol{A}_i(z)$ can be found to complete the factorization if and only if

$$\prod_{i=0}^{Q-1} \det \boldsymbol{S}_i(z) = \det \boldsymbol{U}(z)$$

When the factorization can be completed, the choice of $\boldsymbol{A}_i(z)$ is not unique.

Proof. See Appendix A.

Recall, the form of the lifting factorization of $\boldsymbol{E}(z)$ is given by equation (3.3). Clearly, this factorization is a special case of the decomposition considered in Theorem 3.2. That is, the lifting factorization is a special case where the Type S matrices are constrained to be constant (i.e., independent of z). If in the theorem, we assume that the $\boldsymbol{S}_i$ are constant, this implies that $\det \boldsymbol{U}(z)$ must also be constant in order for the decomposition to exist. Relating this back to the original problem, a lifting factorization of the analysis polyphase matrix exists if and only if $\det \boldsymbol{E}(z)$ is a nonzero constant. Moreover, the theorem also tells us that the factorization is never unique when it exists. Since the lifting factorization of $\boldsymbol{E}(z)$ determines the structure of the lifting realization, the nonuniqueness of the factorization also implies the nonuniqueness of the lifting realization structure.

For a lifting realization to exist, we therefore require

$$\det \boldsymbol{E}(z) = \alpha \tag{3.7}$$

$$\boldsymbol{R}(z) = \boldsymbol{E}^{-1}(z) \tag{3.8}$$

where α is a nonzero constant. Since we are considering FIR PR QMF banks, the analysis and synthesis polyphase matrices must satisfy the conditions

$$\det \boldsymbol{E}(z) = \alpha z^{-K_0} \tag{3.9}$$

$$\boldsymbol{R}(z) = \begin{cases} \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{K_1} \boldsymbol{E}^{-1}(z) & \text{type } 1/2 \\ \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{K_1} \boldsymbol{E}^{-1}(z) & \text{type } 3/4 \end{cases} \tag{3.10}$$

where α is a nonzero constant and the K_i are integers. From (3.9) and (3.10), we can see that conditions (3.7) and (3.8) are not generally satisfied. In order for the conditions to be satisfied, we must have $K_0 = K_1 = 0$. Fortunately, it is always possible to normalize the analysis and synthesis filters so that these constraints are met. To show this, we rely on the following theorem:

Theorem 3.3 *Suppose we are given an M-channel PR QMF bank with FIR filters. Denote the transfer functions of the analysis and synthesis filters of the QMF bank as $H_k(z)$ and $F_k(z)$, respectively. Assume the system is represented in either type 1/2 or type 3/4 polyphase form and has analysis and synthesis polyphase matrices $\boldsymbol{E}(z)$ and $\boldsymbol{R}(z)$, respectively. Since the system is PR and has FIR filters, we have*

$$\det \boldsymbol{E}(z) = \alpha_0 z^{-K_0}$$

and

$$\boldsymbol{R}(z) = \begin{cases} \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{K_1} \boldsymbol{E}^{-1}(z) & \textit{type } 1/2 \\ \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{K_1} \boldsymbol{E}^{-1}(z) & \textit{type } 3/4 \end{cases}$$

where α_0 is a nonzero constant, and the K_i are integers.

Suppose we normalize the analysis and synthesis filters to obtain a new set of analysis and synthesis filters with transfer functions $H'_k(z)$ and $F'_k(z)$, respectively, where this normalization is defined as follows:

$$H'_k(z) = \beta_0 z^{-L_0} H_k(z)$$
$$F'_k(z) = \beta_1 z^{-L_1} F_k(z)$$

Denote the analysis and synthesis polyphase matrices associated with the new filters as $\boldsymbol{E}'(z)$ and $\boldsymbol{R}'(z)$, respectively. Given the above normalization, the following assertions are true:

$$\det \boldsymbol{E}'(z) = \alpha_0 \beta_0^M (-1)^{L_0(M-1)} z^{L_0+K_0}$$

$$\boldsymbol{R}'(z) = \begin{cases} \beta_0\beta_1 \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_0+L_1+K_1} (\boldsymbol{E}'(z))^{-1} & \textit{type 1/2} \\ \beta_0\beta_1 \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_0+L_1+K_1} (\boldsymbol{E}'(z))^{-1} & \textit{type 3/4} \end{cases}$$

This implies that with an appropriate normalization, we can force $\det \boldsymbol{E}'(z)$ *to be any arbitrary nonzero constant up to sign while simultaneously satisfying the constraint* $\boldsymbol{R}'(z) = [\boldsymbol{E}'(z)]^{-1}$.

Proof. See Appendix A.

Let us now normalize the filters of the system to obtain new analysis and synthesis filters with transfer functions $H'_k(z)$ and $F'_k(z)$, respectively, as specified by

$$H'_k(z) = \beta_0 z^{K_0} H_k(z) \tag{3.11}$$

$$F'_k(z) = \beta_1 z^{K_1-K_0} F_k(z) \tag{3.12}$$

where $\beta_0 = 1/\beta_1$. Denote the new analysis and synthesis polyphase matrices obtained from this normalization as $\boldsymbol{E}'(z)$ and $\boldsymbol{R}'(z)$, respectively. From the above theorem, this normalization will always yield a new system satisfying

$$\det \boldsymbol{E}'(z) = \alpha_0 \beta_0^M (-1)^{K_0(M-1)} \tag{3.13}$$

$$\boldsymbol{R}'(z) = [\boldsymbol{E}'(z)]^{-1} \tag{3.14}$$

Clearly, this new system satisfies conditions (3.7) and (3.8). Therefore, a lifting realization of the new system must always exist. Moreover, we can see from above that it is always possible to further force $\det \boldsymbol{E}'(z) \in \pm 1$ by choosing $\beta_0 = |\alpha|^{-1/M}$. If we can always force $\det \boldsymbol{E}'(z) \in \pm 1$, this implies we can always choose the scaling factors on the analysis side in the lifting realization to be ± 1. This turns out to be useful later when we use lifting to construct reversible transforms.

It is important to note that the normalization process does not affect the fundamental behavior of the QMF bank. The relative magnitude and phase responses of the analysis filters remain unchanged since all have an equal gain and equal phase shift added. The same statement also holds true for the synthesis filters. PR is not affected although the reconstruction delay will generally change as the normalization process forces a particular reconstruction delay. In terms of the linear transform calculated by the QMF bank, this normalization simply scales and translates the basis functions, but does not change their shape. Since the shape of the basis functions largely determines the transform's behavior, this normalization does not affect the fundamental behavior of the underlying transform. By performing the above normalization, we ensure that the filter bank under consideration can always be implemented using lifting.

The normalization suggested above is not the only one possible. Another possibility is suggested by Daubechies and Sweldens in [15]. In this case, they propose changing the phase delay and gain of a single analysis filter. The author felt that in some cases it might be more desirable to distribute the phase delay and gain change equally across all of the analysis filters—hence, the reason for proposing the above normalization scheme.

By using the lifting realization of a QMF bank, we obtain a transform with several desirable properties:

- The transform can be calculated in place without the need for auxiliary memory.
- Even in the presence of coefficient quantization error (for the ladder step filters), the PR property is retained.
- The inverse transform has exactly the same computational complexity as the forward transform.
- Asymptotically, for long filters, the lifting realization yields a more computationally efficient structure than the standard realization [15].
- The lifting realization can be used to easily construct reversible transforms (as will be seen later).

3.10 Lifting Factorization Algorithm

Lifting factorizations can be computed using a matrix Euclidean algorithm. To compute the lifting factorization of the analysis polyphase matrix $\boldsymbol{E}(z)$, we proceed as follows:

1. Assume that the filter bank has been normalized (as described previously) so that $\det \boldsymbol{E}(z)$ is a nonzero constant.
2. Choose any $\boldsymbol{S}_i$ for $i = 0, 1, \ldots, \sigma - 1$ from $\mathcal{S}(\cdot)$ that satisfy

$$\prod_{i=0}^{\sigma-1} \det \boldsymbol{S}_i = \det \boldsymbol{E}(z)$$

   For example, if $\det \boldsymbol{E}(z) = d$, we can simply select $\sigma = 1$, $\boldsymbol{S}_0 = \mathcal{S}(0; d)$.

3. Calculate the quantity $\boldsymbol{B}(z)$ as

$$\boldsymbol{B}(z) = \left[\prod_{i=0}^{\sigma-1} \boldsymbol{S}_i^{-1}\right] \boldsymbol{E}(z)$$

4. Perform operations from $\mathcal{A}(\cdot)$ (i.e., Type A row/column operations) on $\boldsymbol{B}(z)$ until it is reduced to an identity matrix. The idea here is similar to Gaussian elimination except that division of arbitrary Laurent polynomials is avoided and replaced by division with remainder (via the Euclidean algorithm). This process yields

$$\boldsymbol{D}_{\nu-1}(z) \cdots \boldsymbol{D}_1(z) \boldsymbol{D}_0(z) \boldsymbol{B}(z) \boldsymbol{C}_0(z) \boldsymbol{C}_1(z) \cdots \boldsymbol{C}_{\mu-1}(z) = \boldsymbol{I}$$

   where the $\boldsymbol{D}_i(z)$ and $\boldsymbol{C}_i(z)$ correspond to Type A elementary row and column operations on $\boldsymbol{B}(z)$, respectively. From this, we can write the factorization of $\boldsymbol{B}(z)$ as

$$\boldsymbol{B}(z) = \boldsymbol{D}_0^{-1}(z) \boldsymbol{D}_1^{-1}(z) \cdots \boldsymbol{D}_{\nu-1}^{-1}(z) \boldsymbol{C}_{\mu-1}^{-1}(z) \cdots \boldsymbol{C}_1^{-1}(z) \boldsymbol{C}_0^{-1}(z) \tag{3.15}$$

5. The lifting factorization of $\boldsymbol{E}(z)$ is then

$$\boldsymbol{E}(z) = \left[\prod_{i=0}^{\sigma-1} \boldsymbol{S}_i\right] \boldsymbol{B}(z)$$

   where $\boldsymbol{B}(z)$ is given by (3.15).

Since the choice of $\boldsymbol{S}_i(z), \boldsymbol{C}_i(z)$, and $\boldsymbol{D}_i(z)$ are not unique for a given $\boldsymbol{E}(z)$, the factorization is also not unique. In particular, the $\boldsymbol{C}_i(z)$ and $\boldsymbol{D}_i(z)$ are influenced by the terms cancelled during Euclidean division and the sequence of rows and columns operated on during the reduction of $\boldsymbol{B}(z)$.

It is important to emphasize that in a lifting decomposition all factors are Laurent polynomial matrices. Although the factorization algorithm is similar in spirit to Gaussian elimination, Gaussian elimination cannot be used since it requires closure on division. Laurent polynomials are not closed on division. That is, the quotient of two Laurent polynomials is generally a rational polynomial expression and not another Laurent polynomial. For more information on matrix Euclidean algorithms, the reader is referred to [16] (see Theorem 22.8).

3.11 Lifted Transform Example

To demonstrate the technique for finding lifting realizations of subband transforms, a simple but useful example is now provided. Consider the subband transform computed by the QMF bank having analysis and synthesis filters with transfer functions $H_k(z)$ and $F_k(z)$ respectively, where

$$H_0(z) = \tfrac{1}{2}(1+z), \quad H_1(z) = 1-z, \quad F_0(z) = 1+z, \quad F_1(z) = \tfrac{1}{2}(1-z)$$

This is nothing more than a scaled version of the Haar transform. Suppose we want to implement the QMF bank using the lifting realization in conjunction with the type 3/4 polyphase form. First, we calculate the analysis polyphase matrix $\boldsymbol{E}(z)$ and its determinant. This yields

$$\boldsymbol{E}(z) = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 1 & -1 \end{bmatrix}$$

$$\det \boldsymbol{E}(z) = -1$$

Since $\det \boldsymbol{E}(z)$ is a constant, a lifting realization exists and no re-normalization of the analysis filters is required. Using Euclid's algorithm, we find

$$\begin{aligned}
\boldsymbol{E}(z) &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ -1 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \\
&= \mathcal{S}(1;-1)\mathcal{A}(0,1;\tfrac{1}{2})\mathcal{A}(1,0;-1)
\end{aligned}$$

Thus, we immediately have the following decomposition for the synthesis polyphase matrix $\boldsymbol{R}(z)$:

$$\begin{aligned}\boldsymbol{R}(z) &= \begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}\begin{bmatrix}1 & -\frac{1}{2}\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\\ &= \mathcal{A}(1,0;1)\mathcal{A}(0,1;-\tfrac{1}{2})\mathcal{S}(1;-1)\end{aligned}$$

These lifting factorizations yield the realization shown in Figure 3.6. By inspection, it is obvious that the QMF bank is PR. The synthesis side undoes step-by-step each operation performed on the analysis side.

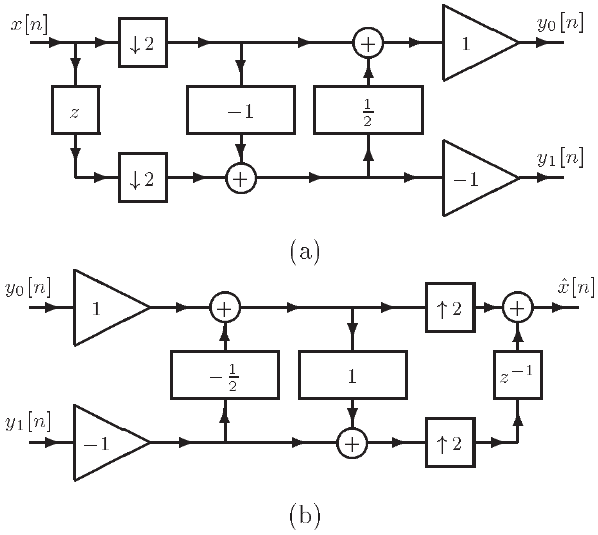


Figure 3.6. *Example lifting realization. (a) Analysis side (forward transform). (b) Synthesis side (inverse transform).*

3.12 Symmetry-Preserving Transforms

Although we can construct PR QMF banks with linear-phase filters that yield symmetric subband signals for a symmetric input, it would be convenient for a number of reasons if symmetry could also be preserved by each and every filtering step in the QMF bank. That is, it would be convenient if all intermediate signals were also symmetric. Fortunately, it is possible to achieve this goal under certain circumstances. The idea of such symmetry-preserving transforms is suggested by Daubechies and Sweldens in [15]. In what follows, we further elaborate on the idea in order to aid in understanding.

Here, we consider transforms computed by 2-channel PR QMF banks with FIR filters. Furthermore, the QMF banks are assumed to be implemented in type 3/4 polyphase form using lifting. In other words, we have a transform computed using the structure depicted in Figure 3.7. For convenience (in what follows), the ladder step filters have been labelled differently from previous sections, but the system is clearly of the same form presented before. Our goal here is to devise a scheme that preserves symmetry at all points in the circuit.

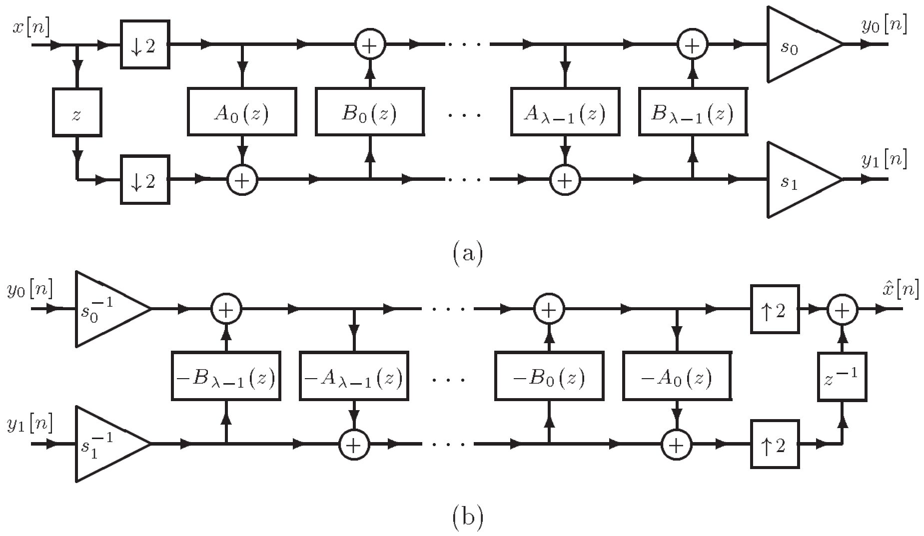


Figure 3.7. *Symmetry-preserving lifting realization. (a) Analysis side (forward transform). (b) Synthesis side (inverse transform).*

Obviously, it is only meaningful to speak of symmetry preservation in the case that the input signal $x[n]$ is, in fact, symmetric. Therefore, we begin by assuming that this is the case. In anticipation of things to come, we further assume that $x[n]$ is symmetric about $n = 0$. As it turns out, this additional assumption does not impose any serious practical limitations.

Denote the even and odd polyphase components of $x[n]$ as $s_0[n]$ and $d_0[n]$, respectively. Since $x[n]$ is symmetric about $n = 0$, it is not difficult to convince oneself that $s_0[n]$ and $d_0[n]$ are symmetric about $n = 0$ and $n = -\frac{1}{2}$, respectively. Therefore, the two inputs to the ladder network in Figure 3.7(a) are symmetric signals. Now we want to devise a scheme that preserves the symmetry in these two channels as they are processed by the ladder steps. In order to achieve this goal, we need to make the following observations:

1. If a linear-phase filter is presented with a symmetric input, the output of the filter will also be symmetric. Moreover, the center of symmetry of the output signal is equal to the center of symmetry of the input signal shifted by the group delay of the filter.
2. If two symmetric signals with the same center of symmetry are added, the result is another symmetric signal with the same center of symmetry.

From these two observations, we can see that a ladder step will preserve symmetry if the ladder step filter has linear phase and its group delay is such that the symmetry points of the two signals being added align. From above, the two polyphase components of $x[n]$ have centers of symmetry differing by $\frac{1}{2}$. This implies that ladder step filters must be chosen such that their group delays are $\pm\frac{1}{2}$ with the sign depending on which channel is being modified by the ladder step. To be more precise, the ladder steps can be forced to preserve signal symmetry if they are chosen such that

1. filters A_i have linear phase with group delay $-\frac{1}{2}$ (i.e., impulse responses $a_i[n]$ are symmetric about $n = -\frac{1}{2}$), and
2. filters B_i have linear phase with group delay $\frac{1}{2}$ (i.e., impulse responses $b_i[n]$ are symmetric about $n = \frac{1}{2}$).

Of course, the final scaling on the analysis side of the QMF bank does not affect symmetry. Thus, the above conditions are all that is required in order to preserve symmetry at all points on the analysis side of the filter bank. The subband signals $s[n]$ and $d[n]$ are obtained with centers of symmetry at $n = 0$ and $n = -\frac{1}{2}$, respectively. Due to the structural similarities between analysis and synthesis sides of the QMF bank, it is evident that symmetry is also preserved at all points on the synthesis side.

Given that it is possible to generate symmetry-preserving transforms in the above manner, the next natural question is what conditions are required in order for a lifting realization of the above form to exist. It can be shown that in order for such a realization to exist, the analysis filters H_0 and H_1 must satisfy the following conditions:

1. The filters H_0 and H_1 must be filters of odd length with impulse responses symmetric about $-\frac{1}{2}$ and $\frac{1}{2}$, respectively.
2. An appropriate scaling of the filter gains is required.

Clearly, it is not always possible to realize a QMF bank in this form, even if the filters all have linear phase. Fortunately, however, many practically useful sets of filters do satisfy the conditions given above. Consequently, this type of realization is of great practical value.

3.13 Symmetry-Preserving Transform Example

For illustrative purposes, we now generate a symmetry-preserving lifting realization of the CDF22 transform (described later in Table 4.1). This 2-band transform is associated with the QMF bank having analysis and synthesis filters with transfer functions $H_k(z)$ and $F_k(z)$, respectively, where

$$H_0(z) = \tfrac{1}{8}(-z^2 + 2z + 6 + 2z^{-1} - z^{-2}), \quad H_1(z) = \tfrac{1}{2}(-z^2 + 2z - 1),$$
$$F_0(z) = z^{-1}H_1(-z), \quad F_1(z) = -z^{-1}H_0(-z)$$

We begin by noting that the analysis filters satisfy the conditions stated in the previous section. Therefore, a realization of the desired form exists. Next, we compute the analysis polyphase matrix $\boldsymbol{E}(z)$ to be

$$\boldsymbol{E}(z) = \begin{bmatrix} -\frac{1}{8}z + \frac{6}{8} - \frac{1}{8}z^{-1} & \frac{1}{4} + \frac{1}{4}z^{-1} \\ -\frac{1}{2}z - \frac{1}{2} & 1 \end{bmatrix}$$

This matrix can be decomposed into a lifting factorization of the desired form given by

$$\begin{aligned} \boldsymbol{E}(z) &= \begin{bmatrix} 1 & \frac{1}{4}+\frac{1}{4}z^{-1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{2}-\frac{1}{2}z & 1 \end{bmatrix} \\ &= \mathcal{A}(0,1;\tfrac{1}{4}+\tfrac{1}{4}z^{-1})\mathcal{A}(1,0;-\tfrac{1}{2}-\tfrac{1}{2}z) \end{aligned}$$

This factorization yields the filtering structure shown in Figure 3.8. Clearly, the analysis side of the QMF bank will yield symmetric subband signals for a symmetric input with center of symmetry at $n = 0$. Such a system can be used in conjunction with symmetric extension.

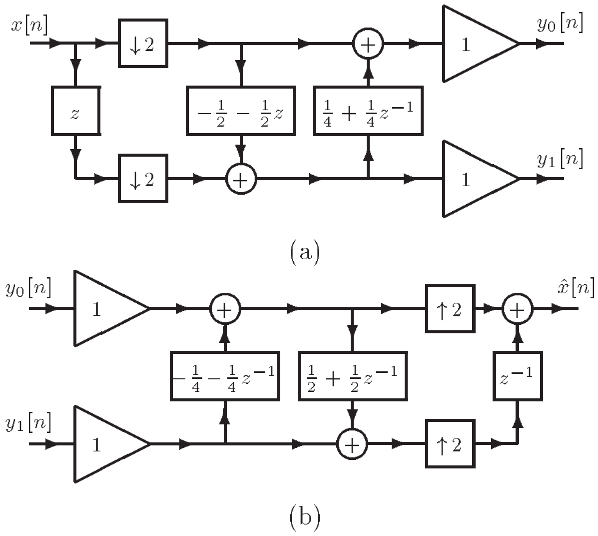


Figure 3.8. *Example symmetry-preserving transform. (a) Analysis side (forward transform). (b) Synthesis side (inverse transform).*

3.14 Reversible Transforms from Lifted Transforms

As suggested previously, lifting can be used to construct reversible versions of linear subband transforms. The basic idea was initially proposed by Calderbank et al. in [10] and is based on the lifting realization of a QMF bank. Although presented in the context of wavelet transforms, the method presented in [10] is directly applicable to any 2-band subband transform. Moreover, the approach is easily extended to the case of M-band transforms with arbitrary M by considering the lifting realization of M-channel QMF banks as was presented earlier. In what follows the more general case of arbitrary M is considered.

Suppose we have a linear subband transform that is computed by an M-channel PR QMF bank with FIR filters. Denote the transfer functions of the QMF bank's analysis and synthesis filters as $H_k(z)$ and $F_k(z)$, respectively. Further, assume that the system when represented in either type

1/2 or type 3/4 polyphase form has the analysis and synthesis polyphase matrices $\boldsymbol{E}(z)$ and $\boldsymbol{R}(z)$, respectively.

As discussed previously, we can always find a lifting realization of a QMF bank after applying an appropriate normalization to its analysis and synthesis filters. Such a realization has the general form shown in Figures 3.4 and 3.5 for the 2-band and M-band cases, respectively. Suppose now that we find a lifting realization such that all scaling factors on the analysis side of the QMF bank are integers. Previously, we showed that the analysis and synthesis filters could be normalized so that $\det \boldsymbol{E}(z)$ can be forced to be any nonzero constant up to sign. Consequently, we can always normalize the filters so that $\det \boldsymbol{E}(z)$ is an integer. Moreover, if $\det \boldsymbol{E}(z)$ is an integer, we can always choose the Type S factors in the lifting factorization of $\boldsymbol{E}(z)$ to be all integer matrices. For example, if $\det \boldsymbol{E}(z) = \alpha$, we can always choose one Type S factor of the form $\mathcal{S}(\cdot;\alpha)$. Thus, by normalizing the analysis and synthesis filters appropriately, a lifting realization of the desired form can always be obtained.

Suppose now that we modify the above lifting structure as follows. First, on the analysis side we add a quantizer at the output of each ladder step filter. In other words, we apply the transformation shown in Figure 3.9(a) to each of the ladder steps on the analysis side of the QMF bank. Second, on the synthesis side we modify each ladder step by negating the transfer function of the ladder step filter, adding a quantizer at the output of the resulting filter, and inverting the sign again at the input to the adder. That is, for each ladder step on the synthesis side, we apply the transformation depicted in Figure 3.9(b). After applying these transformations, we obtain the filtering structure shown in Figure 3.10 for the 2-band case. In the M-channel case, the structure is completely analogous, but it is not illustrated since it is difficult to represent pictorially. As an optimization, however, in the M-band case, one would probably want to group adjacent ladder steps with the same destination channel together where possible, summing their result and then performing quantization only once for the whole group. This will likely lead to reversible transforms that better approximate their parent linear transforms.

Examining the new filtering structure, it is easy to see that the PR property is maintained. In fact, the system remains PR for any (reasonable) choice of quantization operator $Q(\cdot)$. For example, rounding to the nearest integer where

$$Q(x) = \left\lfloor |x| + \tfrac{1}{2} \right\rfloor \operatorname{sgn} x$$

and truncation where

$$Q(x) = \lfloor x \rfloor$$

are two possible choices of quantization operators. As long as the same quantization scheme is employed on both the analysis and synthesis sides of the QMF bank, PR is not affected. Of course, the resulting system is nonlinear, and the new subband signals (i.e., the outputs of the analysis side) serve only to approximate the original subband signals. Provided the error introduced by the quantization operator $Q(\cdot)$ is not too large, the new nonlinear system will well approximate the input-output behavior of the original linear system.

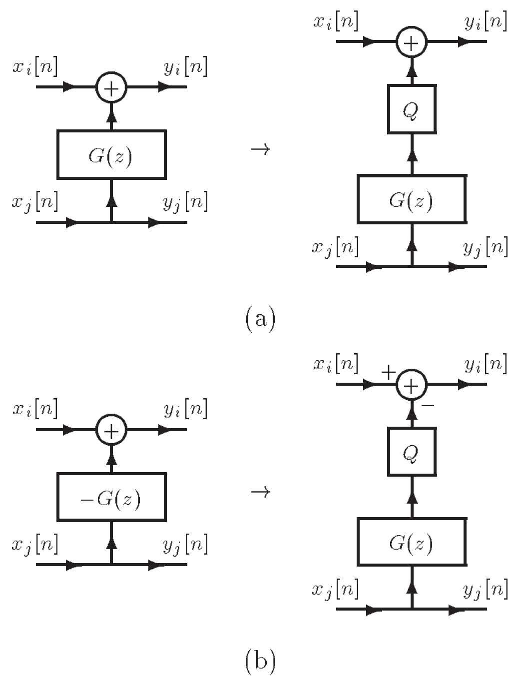


Figure 3.9. *Ladder step transformations. (a) For analysis side. (b) For synthesis side.*

Assume now that the input to the filter bank consists only of integers. This imposes no serious practical limitations since in most applications the input sequence can be easily made integer with an appropriate scaling. Now suppose we choose either rounding or truncation as the quantization strategy. Since the inputs to the analysis side of the filter bank are integers, and only integer adjustments are made by the ladder steps, the outputs of all ladder steps are integer. Furthermore, as the final scaling on the analysis side is by an integer (by assumption), the outputs of the analysis side consist only of integers. Thus, we have a transform that maps integers to integers. Also, it is not difficult to see that this whole process is invertible. The synthesis side of the filter bank will reproduce the original signal exactly despite the nonlinearities introduced by rounding. Moreover, by introducing rounding, we have limited the precision with which the transform must be calculated in order to remain invertible. Thus, we have obtained a reversible transform.

At this point, it becomes apparent why the final scaling on the analysis side is restricted to be by integers. Scaling by a non-integer would cause the precision required for the transform calculation to grow (assuming that invertibility must be preserved). If we try to combat this precision growth problem by rounding the outputs of the scaling units, the transform's invertibility would be compromised.

Although scaling by any integer on the analysis side is possible, it is usually most desirable to use scaling factors of ± 1. A lifting realization with such factors can always be obtained by normalizing the analysis filters such that $\det \boldsymbol{E}(z) \in \pm 1$. This choice is due to the fact that we are usually interested in using orthogonal or near-orthogonal transforms. If scaling factors other than ± 1 are

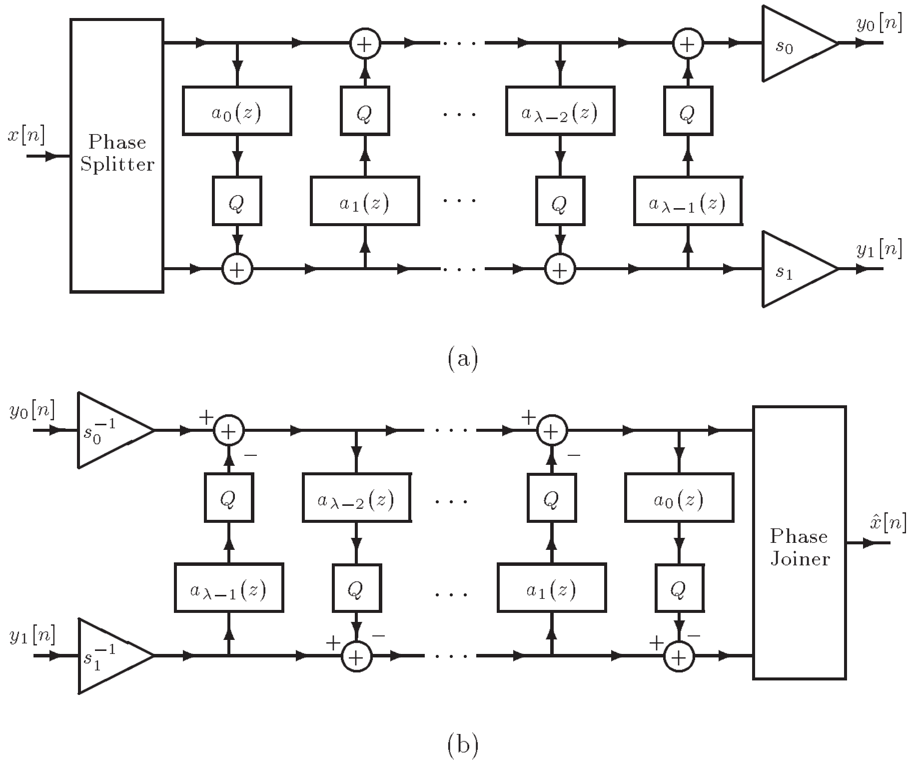


Figure 3.10. *General structure for reversible two-band transforms. (a) Analysis side (forward transform). (b) Synthesis side (inverse transform).*

used, the resulting transform will not be close to being orthogonal.

Using the lifting-based technique described in this section, it is possible to construct a reversible version of any linear transform from the large class of transforms under consideration. This method results in a reversible transform with the following desirable properties:

- The transform approximates the original linear transform from which it was derived.
- The transform maps integers to integers provided the quantization operator employed always yields integer values.
- The transform can be calculated in place.
- The transform is well suited to computation using only fixed-point arithmetic.
- The inverse transform has exactly the same computational complexity as the forward transform.
- The transform is invertible even in the presence of numerical overflow.

As a final note, it is important to emphasize that the introduction of quantization results in nonlinear transforms. Consequently, different lifting factorizations of the same filter bank lead to distinct reversible transforms with approximation characteristics that can vary considerably.

3.15 Reversible Transform Example

To demonstrate the method used to construct a reversible version of a subband transform, an example is now presented. Consider the 2-channel QMF bank with analysis and synthesis filters H_k and F_k, respectively, where

$$H_0(z) = \tfrac{1}{2}(1+z), \quad H_1(z) = 1-z, \quad F_0(z) = 1+z, \quad F_1(z) = \tfrac{1}{2}(1-z)$$

This corresponds to a scaled version of the Haar transform. Suppose now that we want to implement the QMF bank using a lifting realization in conjunction with the type 3/4 polyphase form. We begin by calculating the analysis polyphase matrix $\boldsymbol{E}(z)$ and its determinant as

$$\boldsymbol{E}(z) = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 1 & -1 \end{bmatrix}$$
$$\det \boldsymbol{E}(z) = -1$$

Since $\det \boldsymbol{E}(z) \in \pm 1$, a lifting factorization of the form desired for constructing reversible transforms exists. For the decomposition, we choose a single Type S factor of $\mathcal{S}(1;-1)$. Note that $\det \boldsymbol{E}(z) = \det \mathcal{S}(1;-1)$, so this choice is valid. Using Euclid's algorithm to complete the factorization, we obtain

$$\begin{aligned} \boldsymbol{E}(z) &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ -1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \\ &= \mathcal{S}(1;-1)\mathcal{A}(0,1;\tfrac{1}{2})\mathcal{A}(1,0;-1) \end{aligned}$$

Thus, we immediately have the decomposition

$$\begin{aligned} \boldsymbol{R}(z) &= \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & -\frac{1}{2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \\ &= \mathcal{A}(1,0;1)\mathcal{A}(0,1;-\tfrac{1}{2})\mathcal{S}(1;-1) \end{aligned}$$

for the synthesis polyphase matrix $\boldsymbol{R}(z)$. These polyphase matrix decompositions yield the reversible transform shown in Figure 3.11. Note that this transform approximates the same linear transform as the S transform. The reversible transform constructed in this example, however, has the added advantage that it can be calculated in place (unlike the S transform).

3.16 Symmetry-Preserving Reversible Transforms

In an earlier section, we alluded to the fact that it is possible to design symmetry-preserving reversible transforms. Now, this idea is explored in more detail.

Here, we consider transforms computed by 2-channel PR QMF banks with FIR filters. Suppose it is possible to realize the QMF bank of interest using a lifting realization of the form shown in

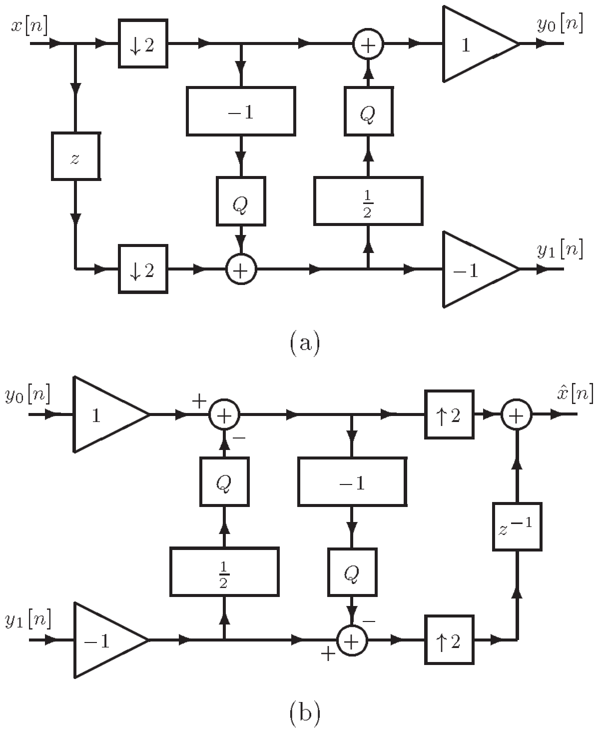


Figure 3.11. *Example reversible transform realization. (a) Analysis side (forward transform). (b) Synthesis side (inverse transform).*

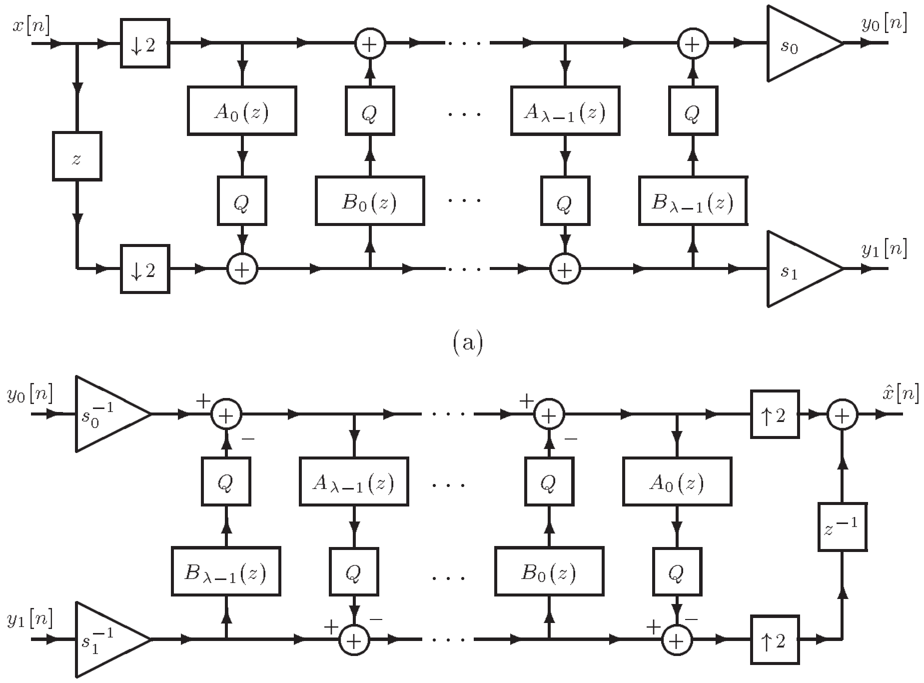


Figure 3.12. *Symmetry-preserving reversible transform realization. (a) Analysis side (forward transform). (b) Synthesis side (inverse transform).*

Figure 3.7 where the ladder step filters are constrained as described in Section 3.12. It has already been demonstrated that such a realization will preserve symmetry at all points in the filtering structure. Although we are dealing with a specific type of lifting realization, it is a lifting realization nevertheless. Consequently, we can make this transform reversible by adding quantizers as discussed earlier. This results in the new filtering structure shown in Figure 3.12. The only question now is whether or not the symmetry-preserving property remains after adding quantization. It is not difficult to see that in order for symmetry to be maintained, the quantization operator employed must have the property that quantizing a symmetric signal always results in a symmetric signal with the same center of symmetry. Clearly, any reasonable choice of quantization operator will satisfy this constraint. For example, one could choose the quantization operator to be rounding (to the nearest integer) or truncation. Thus, we can construct a symmetry-preserving reversible version of any transform that can be realized in the manner described in Section 3.12.

3.17 Symmetry-Preserving Reversible Transform Example

To illustrate the concepts behind the previous section, a symmetry-preserving reversible transform is now constructed. Again, we consider the CDF22 transform as presented in Section 3.13. In the

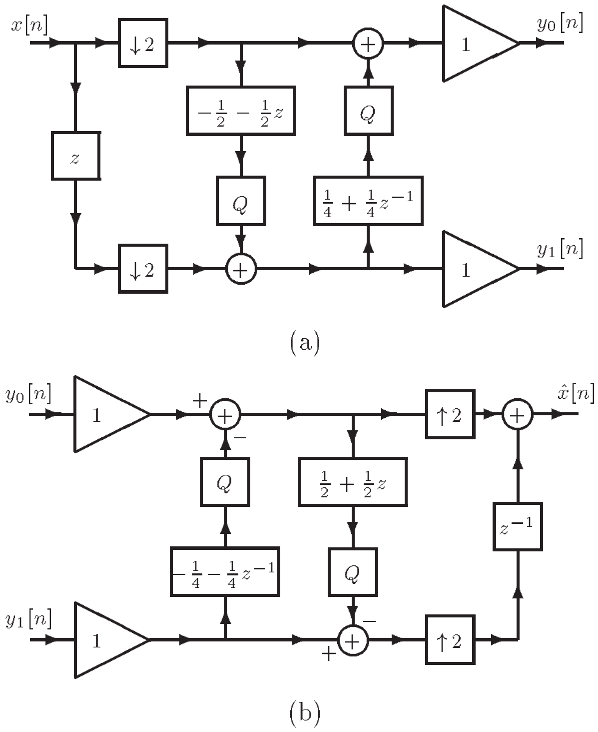


Figure 3.13. *Example symmetry-preserving reversible transform realization. (a) Analysis side (forward transform). (b) Synthesis side (inverse transform).*

same section, it was also shown that this transform can be computed using the lifting realization shown in Figure 3.8. In order to construct a reversible version of this transform, we simply add quantizers to yield the structure shown in Figure 3.13. For the reasons mentioned in the previous section, this transform is symmetry preserving.

3.18 Approximation Accuracy

Previously, it was claimed that the lifting-based reversible transform design method can yield transforms that well approximate their parent linear transforms. Moreover, it was also stated that transforms obtained from this method are well suited to implementation using only fixed-point arithmetic. Both of these claims will now be substantiated with numerical results.

In order to demonstrate the accuracy with which reversible transforms approximate their parent linear transforms, the approximation error was studied for numerous transforms. Here the results for two transforms are presented: reversible versions of the BCW3 and CDF97 transforms. These transforms and the particular scalings used were deliberately chosen to obtain arbitrary (i.e., non-dyadic rational) filter coefficients. To support the above claims, a six-level wavelet decomposition of two well-known 8 bit/pixel (bpp) grayscale test images (i.e., `lena` and `barb`) was computed using both the reversible transform and its parent linear transform. The transform coefficients obtained with the reversible transform were then compared to those obtained from the parent transform in each case. In particular, the mean absolute error, maximum absolute error, and percentage of coefficients that are in error by one or less were used for comparison purposes. Tables 3.1 and 3.2 show the results obtained for the BCW3 and CDF97 transforms, respectively. In both cases, we can see that quantization does not introduce appreciable error when floating-point arithmetic is employed for the reversible transform. Moreover, we also see that using fixed-point arithmetic in the reversible transform does not introduce significantly more error than floating-point provided at least 8 bits of fraction are used.

Table 3.1. *BCW3 fixed-point results (6-level wavelet decomposition)*

Image	Fraction Bits	Mean Absolute Error	Maximum Absolute Error	Percent Within One
`lena`	floating-point	0.51828	5.9091	87.901
	4	1.4866	1410.6	66.399
	5	1.0333	552.31	71.872
	6	0.64407	340.68	86.840
	7	0.66702	148.16	82.142
	8	0.51922	6.5028	87.879
	10	0.51922	6.5028	87.879
	12	0.51843	6.6594	87.956
	14	0.51835	6.6594	87.903
	16	0.51829	5.9091	87.900
`barb`	floating-point	0.51997	5.4138	87.830
	4	1.5277	1631.9	64.749
	5	1.0418	638.01	70.951
	6	0.64937	393.98	86.281
	7	0.66214	150.01	82.074
	8	0.51988	5.2079	87.818
	10	0.51988	5.2079	87.818
	12	0.52050	7.0792	87.819
	14	0.52032	6.0319	87.817
	16	0.51998	5.4138	87.830

In the case of some transforms, the filter coefficients may all be dyadic rational. That is, the coefficients may all be of the form $x/2^N$. Clearly, in such cases, using fixed-point arithmetic with N bits of fraction will introduce no further error beyond that caused by quantization.

Table 3.2. *CDF97 fixed-point results (6-level wavelet decomposition)*

Image	Fraction Bits	Mean Absolute Error	Maximum Absolute Error	Percent Within One
`lena`	floating-point	1.0899	8.7263	54.973
	4	4.4392	212.90	20.783
	5	3.8618	57.396	19.477
	6	1.9570	28.568	32.958
	7	1.5588	19.913	39.862
	8	1.0965	9.0052	54.731
	10	1.1038	9.9013	54.395
	12	1.0925	9.1511	54.918
	14	1.0902	8.1978	54.976
`barb`	floating-point	1.0939	9.7654	54.862
	4	4.9282	294.71	18.408
	5	3.9689	63.518	18.439
	6	2.0283	30.281	31.525
	7	1.5377	24.718	40.243
	8	1.1015	9.0794	54.519
	10	1.1126	9.4053	54.022
	12	1.0932	9.3577	54.838
	14	1.0951	9.7654	54.765

3.19 Practical Design Method

Given any linear transform computed by an FIR PR QMF bank, we can construct a reversible version of the transform using lifting. As we have seen, this process is equivalent to finding a lifting factorization of the analysis polyphase matrix of the QMF bank (with integer scaling factors). Since the lifting factorization is not unique, this raises the question of which factorization to select. In fact, we might want to compute many factorizations and then select one based on some particular criteria.

Although lifting factorizations can be computed by hand, this process can be very tedious and error prone (especially for the M-band case). This motivates us to calculate these factorizations using computer software. Unfortunately, the factorization problem can become numerically ill-conditioned. In order to address this difficulty, a simple iterative technique can be employed. Rather than using a complex factoring algorithm, a simple factoring algorithm can be employed iteratively until a good factorization is obtained.

In order to quickly determine if a factorization is a good one, a simple metric is employed. This goodness metric is defined in terms of the lifting factorization coefficients as

$$\gamma = \frac{\text{largest coefficient magnitude}}{\text{smallest coefficient magnitude}} \tag{3.16}$$

where a smaller value corresponds to a better factorization. When the value of the metric exceeds some prespecified threshold, a factorization is deemed to be poorly conditioned and is discarded. Typically, a value of about 100 to 1000 was found to work reasonably well. Once any bad factorizations have been discarded, some other criteria can be used to choose from the remaining factorizations. For example, one might choose the factorization with the fewest (Type A) factors.

This leads to the following algorithm for designing reversible transforms:

1. Input the number of channels M, the analysis filter transfer functions $H_k(z)$, the goodness threshold ρ, and the number of iterations N.
2. Normalize the transfer functions $H_k(z)$ to obtain the new analysis filter transfer functions $H'_k(z)$. This normalization should yield a new analysis polyphase matrix $\boldsymbol{E}'(z)$ such that $\det \boldsymbol{E}'(z)$ is an integer.
3. Using the method described in Section 3.10, compute a lifting factorization of $\boldsymbol{E}'(z)$. This step should be randomized so as to produce a different factorization on each iteration.
4. Calculate the goodness metric γ for the factorization obtained in step (3) as given by (3.16). If $\gamma < \rho$, then add the factorization to the list of feasible solutions.
5. Decrement N. If $N > 0$, then go to step (3).
6. Select a factorization from the feasible list based on some user-specified criteria, and output this factorization. This lifting factorization of $\boldsymbol{E}'(z)$ directly yields a reversible transform.

Typically, 100 iterations are sufficient to obtain a good design, although this value depends on the number of channels and length of the analysis filters involved.

To demonstrate the effectiveness of this method, a reversible version of the BCW3 transform (see Table 4.1) was constructed. The transform is based on a 2-channel QMF bank in type 1/2 polyphase form. The steps in the forward transform are given in Table 3.3. The inverse transform is obtained by performing the inverse of these steps in reverse order. The transform is intended to be calculated using fixed-point arithmetic with 15 bits of fraction.

Table 3.3. *Forward transform lifting factorization*

Step	Operation
0	$\mathcal{A}\left[1, 0; \frac{1}{32768}(2048z^{-2} - 18432z^{-1} - 4859 + 2048z)\right]$
1	$\mathcal{A}\left[0, 1; \frac{1}{32768}(32768)\right]$
2	$\mathcal{A}\left[1, 0; \frac{1}{32768}(-9598)\right]$
3	$\mathcal{A}\left[0, 1; \frac{1}{32768}(-2048z^{-1} - 27909 + 18432z - 2048z^{2})\right]$

Since the BCW3 transform is known to be effective for image compression, the new reversible transform was tested in this context. To evaluate the resulting transform, the two standard test images barb and lena were used. Both images are 512×512 and 8 bpp grayscale. The mean and maximum absolute difference between the reversible and nonreversible versions of the transform for these images are given in Table 3.4. As can be seen, the error is quite small; indeed, the reversible transform does an excellent job at approximating its parent transform. This demonstrates the effectiveness of the design method.

Table 3.4. *Transform approximation accuracy*

Image	Mean Absolute Error	Maximum Absolute Error
barb	0.5202	6.149
lena	0.5182	6.659

3.20 S+P Transform

Initially proposed by Said and Pearlman [36], the S+P transform is a reversible transform that maps integers to integers and is parameterized by two sets of filter coefficients. This transform is a further refinement of the S transform where the highpass coefficients are adjusted by an additional prediction step. The forward transform splits the input signal $x[n]$ into the lowpass and highpass signals $s[n]$ and $d[n]$, respectively, as given by

$$\begin{aligned} s[n] &= \left\lfloor \tfrac{1}{2}(x[2n] + x[2n+1]) \right\rfloor \\ d[n] &= d_0[n] - \left\lfloor \hat{d}[n] + 1/2 \right\rfloor \end{aligned}$$

where

$$\begin{aligned} d_0[n] &= x[2n] - x[2n+1] \\ \hat{d}[n] &= \sum_{i=L_0}^{L_1} \alpha_i (s[n-i-1] - s[n-i]) - \sum_{i=K}^{-1} \beta_i d_0[n-i] \end{aligned}$$

where L_0, L_1, and K are integers satisfying $L_0 \leq L_1$ and $K \leq -1$. The inverse transform uses the lowpass and highpass signals $s[n]$ and $d[n]$ to reconstruct the original input signal $x[n]$ as given by

$$\begin{aligned} x[2n] &= s[n] + \left\lfloor \tfrac{1}{2}(d_0[n] + 1) \right\rfloor \\ x[2n+1] &= x[2n] - d_0[n] \end{aligned}$$

where

$$d_0[n] = d[n] + \left\lfloor \hat{d}[n] + \tfrac{1}{2} \right\rfloor$$

and $\hat{d}[n]$ is as given above.

Table 3.5. *S+P transform predictor coefficients*

Predictor	α_{-1}	α_0	α_1	β_{-1}
A	$\frac{1}{4}$	$\frac{1}{4}$	0	0
B	$\frac{3}{8}$	$\frac{2}{8}$	0	$\frac{2}{8}$
C	$\frac{8}{16}$	$\frac{4}{16}$	$-\frac{1}{16}$	$\frac{6}{16}$

Three sets of predictor coefficients have been suggested [36] as listed in Table 3.5. Of these, predictor A has the smallest computational complexity and yields a reversible version of the well-known TS transform. In the degenerate case where all of the predictor coefficients are zero (i.e., $\alpha_i = \beta_i = 0$), the S transform is obtained.

The filtering structure for the S+P transform is shown in Figure 3.14 where $A(z)$, $B(z)$, $Q_T(x)$, and $Q(x)$ are defined as

$$A(z) = (1 + z^{-1}) \sum_{i=L_0}^{L_1} \alpha_i z^{-i}, \quad B(z) = \sum_{i=K}^{-1} \beta_i z^{-i}, \quad Q_T(x) = \lfloor x \rfloor, \quad Q(x) = \lfloor x + \tfrac{1}{2} \rfloor$$

Observe that the first part of the forward transform structure and last part of the inverse transform structure are nothing more than the S transform. As we saw earlier (in Section 3.15), it is possible to realize the S transform using lifting. Replacing the S transform with its lifted version, we obtain a transform that can be calculated in place. With this filter structure, it is important to note that the synthesis side has a feedback path if any of the β_i coefficients are nonzero. In such a case, the synthesis side of the filter bank is no longer associated with FIR filters.

If we disregard the effects of the truncation in the S+P transform, we have a linear subband transform that corresponds to a QMF bank having analysis filters with transfer functions $H_k(z)$ where

$$H_0(z) = \tfrac{1}{2}(1 + z), \quad H_1(z) = -\tfrac{1}{2}(1 + z)A(z^2) + (1 - z)[1 + B(z^2)]$$

Assuming that predictor A, B, or C is used, this linear transform is not orthogonal. Moreover, the transform must be scaled to be near-orthogonal. Thus, the S+P transform does not directly approximate an orthogonal or near-orthogonal transform. By weighting the transform coefficients associated with each subband by an appropriate constant, however, a near-orthogonal transform is obtained. As it turns out, in the two-dimensional case, these weighting factors are integer powers of two which is convenient for computational reasons.

So far, it has been assumed that we are dealing with signals of infinite length. The S+P transform, however, is easily adapted to handle finite-length signals if we assume the signal is defined for $n = 0, 1, \ldots, N-1$ where N is even. This assumption is required so that when the input signal is split into its two polyphase components, both signals will be defined for the same range of indices, namely $n = 0, 1, \ldots, N/2 - 1$. The only remaining problems are due to the filters A and B. Fortunately, the transform remains invertible independent of the choice of $A(z)$ and $B(z)$. As a consequence, we

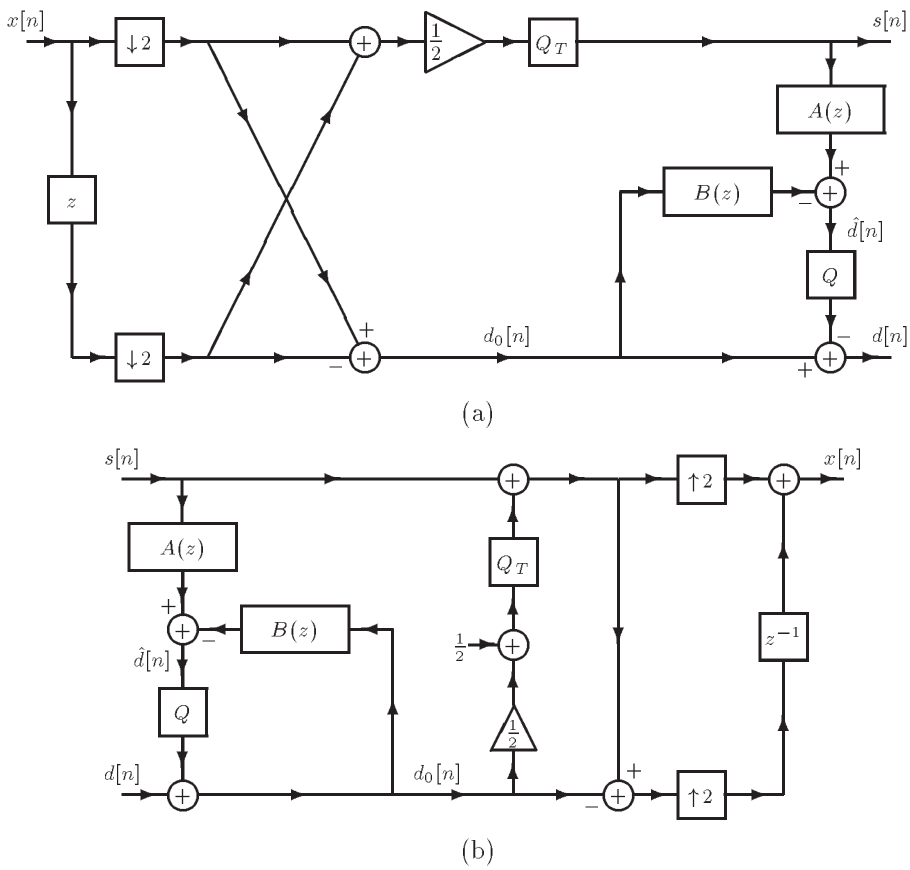


Figure 3.14. *Filtering structure for the S+P transform. (a) Forward transform. (b) Inverse transform.*

can use shorter filters at the signal boundaries to avoid needing undefined sample values. Note that changing the filters A and B is equivalent to altering the predictor $\hat{d}[n]$. For example, in [36] it is suggested that for predictor B, the predictor be changed at the boundaries as follows

$$\hat{d}[n] = \begin{cases} \frac{1}{4}(s[0] - s[1]) & \text{for } n = 0 \\ \frac{1}{4}(s[\frac{N}{2} - 2] - s[\frac{N}{2} - 1]) & \text{for } n = \frac{N}{2} - 1 \end{cases}$$

Although the above approach assumes that N is even, it is possible in principle to define further boundary filters to handle the case of odd N. Doing so, however, would significantly increase the complexity of the transform. An alternative solution is to simply pad a signal by one sample if it is not of even length. Then the S+P transform as defined above can be applied. Periodic extension is not used with the S+P transform due the potential for IIR filters in the synthesis bank.

The S+P transform is typically applied in a pyramidal fashion such that the lowpass signal is successively decomposed. In this case, we have a reversible wavelet transform. The S+P transform does have one peculiarity. Most wavelet systems employed today are associated with finitely-supported scaling and wavelet functions. The S+P transform, however, approximates a system associated with infinitely-supported scaling/wavelet functions. The S+P transform trades off some time localization to achieve better frequency localization.

Since its introduction, the S+P transform has become very popular for lossless image compression. In image compression applications, predictor B is best suited for natural images and predictor C is best suited for very smooth medical images.

3.21 More General Framework for Reversible Transforms

As stated previously, the S+P transform is nothing more than the S transform with an additional prediction step. If, in the S+P transform, we replace the traditional implementation of the S transform with a lifted implementation, we obtain a new structure that is almost in lifting form. The only difference is that one of the building blocks of the ladder network has been modified. This new block uses the signals in both channels to modify the highpass channel whereas a true lifting realization only ever uses the signal in one channel to modify the other. This idea can be generalized, leading to a more flexible structure for reversible transform construction. (A similar idea is suggested by an example in Calderbank et al. [10], but is not quite as general as the idea proposed here.) Instead of using a ladder step as a basic building block on the analysis side of the QMF bank, one could use the modified building block shown in Figure 3.15(a). The operation performed by such a building block is inverted by the structure in Figure 3.15(b). We can use these new building blocks to generalize the idea of lifting.

The general structure used to compute a 2-band lifted transform is shown in Figure 3.4. Suppose now that we replace each ladder step on the analysis side with a block of the form shown in Figure 3.15(a), and replace each ladder step on the synthesis side with a block of the form shown in Figure 3.15(b). In the case where all $B_k(z)$ and $C_k(z)$ are zero, this structure reduces to a lifting realization. Obviously, we can choose these filters to have nonzero transfer functions, and in this case a more general structure capable of realizing an even larger class of subband transforms is obtained. Before continuing further, it is important to understand that there are some constraints that must be placed on $B_k(z)$ and $C_k(z)$. These transfer functions must both be polynomials in strictly z or z^{-1} with a constant term of zero. The zero constant term is required in order to avoid delay-free loops which are physically unrealizable. The filters must be strictly causal or anti-causal in order for inversion to be possible. As one might anticipate, there are no constraints on the $A_k(z)$. Also, with the above constraints on the filters, in-place calculation is still generally possible.

Assuming that some of the $B_k(z)$ and $C_k(z)$ are nonzero, the resulting filtering structure contains feedback paths. Due to this recursion, it is not practical to use periodic extension in order to handle finite-length signals. Boundary filtering must be used instead. Stability can also be a concern due to the presence of feedback. Moreover, stability is difficult to analyze due to the nonlinear nature of the system. Fortunately, we are often only concerned with finite-length signals, and in such cases stability is less of a concern. One still needs to be somewhat careful, however. Even in the case of finite-length signals, instability can cause large amplitude signals for relatively small amplitude

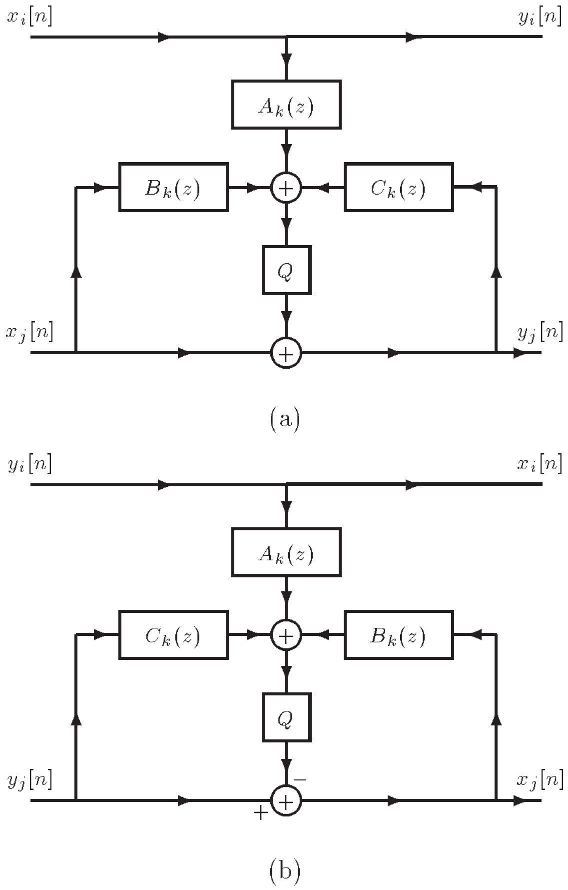


Figure 3.15. *Building blocks for generalized lifting realization. (a) Building block for analysis side. (b) Building block for synthesis side.*

input signals which can lead to numerical overflow.

The obvious advantage of this more general filtering structure is that it can be used to construct reversible versions of transforms based on QMF banks with IIR analysis/synthesis filters. This additional degree of freedom can be beneficial as demonstrated by the effectiveness of the S+P transform which employs IIR filters.

It is not difficult to see that this approach can be easily extended to the M-band case. In such a case, the filtering blocks shown in Figures 3.15(a) and 3.15(b) would have M inputs and M outputs. Since there are more channels, the top adder in both figures would have $M-2$ more inputs corresponding to the filtered versions of the other $M-2$ channels.

As more blocks are used, more boundary filters are required, and therefore complexity increases. Also, error introduced by quantization will tend to be worse for recursive structures as error can accumulate indefinitely. For this reason, one would probably not want to use an excessive number of blocks with $B_k(z)$ and $C_k(z)$ nonzero, but there may be advantages to using a small number of such blocks. This idea has potential for further development in future research.

3.22 Summary

In this chapter, we began by introducing the concept of reversibility, and explaining why transforms with this property are desirable. Then, we pointed out that reversible transforms are generally nonlinear, but are usually designed to approximate linear transforms with desirable properties. Next, we revisited multidimensional and finite-length signals in the context of nonlinear reversible transforms. For the most part, the methods previously discussed in the context of linear transforms are still applicable, but with some caveats due to the nonlinear nature of the reversible transforms.

Lifting was presented as a general framework for the design and implementation of invertible transforms. Then, it was shown how the lifting philosophy can be applied to M-channel PR QMF banks, yielding the lifting realization of a QMF bank. Next, we demonstrated how quantization can be introduced into the lifting realization of a QMF bank in order to obtain a reversible transform. This yields a systematic method for constructing reversible versions of any subband transform computed using an M-channel PR QMF bank with FIR filters. Reversible transforms constructed in this way have many desirable properties which make their use attractive. Moreover, this lifting-based design method allows us to also construct symmetry-preserving reversible transforms. The symmetry-preserving property is advantageous as it allows such transforms to be used in conjunction with symmetric extension.

3.23 Conclusions

In this chapter, several new results were presented relating to reversible transforms. We now summarize briefly these contributions.

By considering the generalization of the lifting realization to M-channel QMF banks, we were able to extend the reversible transform construction method of Calderbank et al. [10] to the case of M-band transforms. Also, for completeness, we formally considered lifting realizations based on both the type 1/2 and type 3/4 polyphase forms of a QMF bank. In our treatment of the subject, we suggested another possible way to normalize the analysis and synthesis filters in order to obtain a QMF bank that can be realized using lifting. This normalization is different from the one suggested by Daubechies and Sweldens in [15] (p. 7).

Through numerical results, we demonstrated that the reversible transforms obtained through the lifting-based design method well approximate their parent linear transforms. Although a relatively small approximation error is key to an effective reversible transform, this error has not been studied in the literature to date. From our experimental results, we were able to demonstrate that the lifting-based reversible transform filtering structure is well suited to implementation using only fixed-point arithmetic. For the case of arbitrary filter coefficients, typically, fixed-point arithmetic with 8 to 12 bits of fraction is sufficient to obtain results comparable to those obtained with floating-point. This observation is important as it implies that we can use lifting to build reversible integer-to-integer transforms that require only integer arithmetic for filters with arbitrary coefficients. Also, we suggest another practically useful property of these reversible transforms, namely, that they are invertible even in the presence of numerical overflow.

The lifting-based reversible transform design method is really nothing more than a matrix factorization problem. In this chapter, we proposed a simple lifting factorization algorithm that addresses numerical ill-conditioning problems. Although the algorithm is not the most elegant, it does yield reasonably good results for practically useful filter sets.

The S+P transform can be viewed as an example of a more general form of lifting. Based on this observation, we suggested a more general framework for constructing reversible transforms that combines ideas from both lifting and the S+P transform. This new framework has the advantage that it generates an even larger class of reversible transforms than lifting alone.

Beware of bugs in the above code; I have only proved it correct, not tried it.

—Donald Knuth

Chapter 4

Reversible Embedded Image Compression

To the engineer, all matter in the universe can be placed into one of two categories: (1) things that need to be fixed, and (2) things that will need to be fixed after you've had a few minutes to play with them. Engineers like to solve problems. If there are no problems handily available, they will create their own problems. Normal people don't understand this concept; they believe that if it ain't broke, don't fix it. Engineers believe that if it ain't broke, it doesn't have enough features yet.

—Scott Adams, "The Dilbert Principle"

4.1 Introduction

The particular application of reversible transforms considered in this thesis is reversible embedded image compression. In this chapter, topics relevant to this application are discussed at length. Then, research results obtained in this context are discussed.

The chapter begins with a brief introduction to image coding and image compression. Then, transform-based image compression is discussed along with the motivation for its use. Performance measures for both lossy and lossless compression are also described. Next, we formally specify what is meant by "reversible" and "embedded" in the context of image compression. This is followed by a brief description of the EZW coding scheme. Then, we proceed to describe the modified EZW coder used to obtain most of the results presented in this thesis. Wavelet transforms are discussed in the context of image compression, and it is explained why they are particularly effective for this purpose.

Several linear wavelet transforms are introduced, and then reversible versions of these transforms are constructed using the lifting-based method presented in the previous chapter. These new reversible transforms are then used in our EZW-based image coder and their effectiveness for compression evaluated. Some other issues such as the relative merits of periodic versus symmetric extension and full versus partial embedding are also examined. Based on some of the observations made in this chapter, a new multi-transform approach to image compression is proposed.

4.2 Image Coding and Compression

Image coding is essentially the process of finding an alternative representation of an image with some specific purpose in mind. The goal may be to find a representation with less redundancy so that fewer bits are required to encode the image, or to add redundancy in order to facilitate error detection/correction for information transmitted over noisy channels. The former type of coding is referred to as image compression. Since image compression has become such a popular research area, many use the term "image coding" to be synonymous with "image compression". Strictly speaking, however, this is an abuse of terminology.

There are two general approaches to image compression. In the first approach, the sample values of the original image are used directly to generate the compressed bitstream. Such an approach is often referred to as a spatial-domain approach. In the second method, a transformation is applied to the samples of the original image and then the transform data are used to produce the compressed bitstream. A compression system employing this philosophy is said to be transform-based. For lossy compression, transform-based coders are almost exclusively used as they tend to have much better performance for fixed complexity. On the other hand, in the case of lossless compression, spatial-domain coders are employed.

In this chapter, we consider the application of reversible embedded image compression which deals with both lossy and lossless compression. As it turns out, transform-based coders are superior for this particular application. Ultimately, the choice of a transform-based coder is driven by the fact that lossy compression needs to be handled. Although lossless compression can be handled reasonably well with either spatial-domain coders or transform-based coders, it is exceedingly difficult to handle lossy compression with a spatial-domain coder. Due to the importance of transform-based coders for the application at hand, such systems will be discussed in more detail in the next section.

For a more detailed background on image compression and signal coding techniques, the reader is referred to [34] and [22].

4.3 Transform-Based Image Compression Systems

The general structure of a transform-based image compression system is shown in Figure 4.1. In this diagram, $x[i, j]$ represents the original image, $y[i]$ denotes the compressed image, and $\hat{x}[i, j]$ represents the reconstructed image obtained from decompression. The goal, of course, is to design a system so that the coded signal $y[i]$ can be represented with fewer bits than the original signal $x[i, j]$. Although images are two dimensional signals, the coded signal $y[\cdot]$ is shown as being one-dimensional since it often makes more sense to view the coded signal in this manner. The reasoning behind this is simply that the notion of horizontal and vertical dimensions is usually completely lost after coding is performed. As indicated by the diagram, the compressed bitstream may be stored or transmitted. In the case of storage, compression has the benefit of reducing disk or memory requirements, and in transmission scenarios, compression reduces the bandwidth (or time) required to send the data.

Rather than attempt to code the sample values of the original image directly, a transform-based

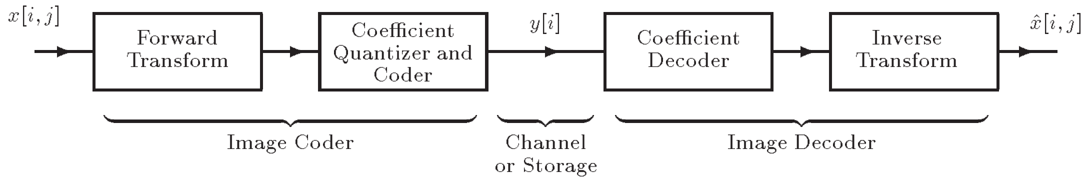


Figure 4.1. *General structure of transform-based image compression system.*

coder first applies a transform to the image and then codes the resulting transform coefficients instead. The transform is used in an attempt to obtain coefficients that are easier to code.

Many signals of practical interest are characterized by a spectrum that decreases rapidly with increasing frequency. Images are a good example of a class of signals with this property. By employing transforms that decompose a signal into its various frequency components, one obtains many small or zero-valued coefficients which correspond to the high-frequency components of the original signal. Due to the large number of small coefficients, the transformed signal is often easier to code than the original signal itself. As a practical matter, in order for a transform to be useful, it must be effective for a reasonably large class of signals and efficient to compute.

Consider now the compression process. First, the transform of the image is calculated. The transform serves to decorrelate the samples of the original signal. That is, the transform packs the energy from the original signal into a small number of coefficients that can be more efficiently coded. The next step in the compression process is to quantize and code the transform coefficients. Quantization is used to discard transform coefficient information that is deemed to be insignificant. In the case of lossless compression, no quantization is performed since all transform coefficient bits are equally important. Finally, the quantized coefficients are coded to produce the compressed bitstream. The coding process typically exploits a statistical model in order to code symbols with a higher probability of occurrence using fewer bits. In so doing, the size of the compressed bitstream is reduced. Assuming that the transform employed is truly invertible, the only potential for information loss is due to coefficient quantization, as the quantized coefficients are coded in a lossless manner.

The decompression process simply mirrors the process used for compression. First, the compressed bitstream is decoded to obtain the quantized transform coefficients. Then, the inverse of the transform used during compression is employed to obtain the reconstructed image $\hat{x}[i,j]$.

4.4 Compression Performance Measures

Obviously, in order to evaluate the performance of image compression systems, we need a way to measure compression. For this purpose, the compression ratio (CR) metric is often employed and is defined as

$$\text{CR} = \frac{\text{original image size in bits}}{\text{compressed image size in bits}}$$

Sometimes compression is instead quantified by stating the bit rate (BR) achieved by compression in bpp (bits per pixel). The bit rate after compression and compression ratio are simply related as

$$\text{BR} = (\text{bits/pixel for original image})/\text{CR}$$

In most cases, the author prefers to use the compression ratio measure since in order for bit rate measures to be meaningful one must also state the initial bit rate. The compression ratio measure, however, is meaningful if stated in isolation.

In the case of lossy compression, the reconstructed image is only an approximation to the original. The difference between the original and reconstructed signal is referred to as approximation error or distortion. Although many metrics exist for quantifying distortion, it is most commonly expressed in terms of mean-squared error (MSE) or peak-signal-to-noise ratio (PSNR). These quantities are defined as

$$\text{MSE} = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(x_{i,j} - \hat{x}_{i,j})^2$$

$$\text{PSNR} = 10\log_{10}\frac{(2^P - 1)^2}{\text{MSE}}$$

where $x[\cdot]$ is the original image with dimensions $M \times N$ and having P bits/pixel, and $\hat{x}[\cdot]$ is the reconstructed image. Evidently, smaller MSE and larger PSNR values correspond to lower levels of distortion. Although these metrics are frequently employed, it is important to point out that the MSE and PSNR metrics do not always correlate well with image quality as perceived by the human eye. This is particularly true at high compression ratios (i.e., low bit rates). For this reason, one should ideally supplement any objective lossy performance measurements with subjective tests to ensure that the objective results are not misleading.

Generally speaking, distortion varies with the amount of compression. In other words, distortion is implicitly a function of compression ratio (or bit rate). For this reason, plots (or tables) of distortion versus compression ratio are often used to analyze lossy compression performance. Obviously, for any given compression ratio, the lowest possible distortion is desired.

In the case of lossless compression, the reconstructed image is an exact replica of the original image. In other words, the distortion is always zero. Since the distortion is zero, we only need to consider the amount of compression achieved when analyzing lossless compression performance. In this context, compression is usually quantified by the compression ratio measure. Obviously, the larger the compression ratio, the better is the compression performance.

4.5 Reversible Embedded Image Compression

Although we have referred to reversible embedded image compression several times in this thesis, we have yet to formally define this term. In this section, we now explain more precisely, what it means for a compression system to be "embedded" and "reversible".

The idea of embedded coding was first popularized by Shapiro in his now well-known paper on EZW coding [41]. A coder is said to be embedded if it is such that for any given signal and any given bit rate, all encodings of the signal at lower bit rates are found at the beginning of the bitstream for the signal coded at the given bit rate. In other words, the information in the coded bitstream is generated in order of importance. As the bit rate increases, the bitstream grows in length, but all of the previously coded bits (generated at lower bit rates) remain unchanged. As a consequence of embedding, the decoder can stop decoding at any point (e.g., when a target bit rate has been met, or a distortion metric satisfied) and the same image is obtained as would have been at the bit rate corresponding to the truncated bitstream. If an embedded coder has the additional property that it exactly reproduces the original input signal at some sufficiently high bit rate, then the coder is said to be reversible.

Since EZW was first proposed, numerous other embedded coding schemes have been developed (e.g., CREW [60], SPIHT [37]). Over time, interest in embedded image compression has continued to grow, due mainly to the large number of applications that can benefit from the embedding property. In particular, embedded compression is ideally suited for image browsing over low bit rate channels. Moreover, reversible embedded image compression systems provide a convenient framework for building hybrid lossy/lossless compression systems. With the numerous benefits of embedded coding, it is likely that this approach will continue to gain favor in the future.

As a last note, the embedding property does impose serious constraints on the coded bitstream. For this reason, embedded coding generally leads to sub-optimal compression results. In practice, however, the compression performance is not usually degraded too significantly. And ultimately, the advantages offered by embedding more than compensate for the small performance penalties incurred by its use.

4.6 EZW Coding Scheme

The Embedded Zerotree Wavelet (EZW) coding scheme was proposed by Shapiro and is described in [39], [41], [40], and [42]. As the name suggests, this coding method has the embedding property. Essentially, there are three key elements to the EZW scheme:

1. The wavelet transform is used to form a hierarchical subband decomposition of an image.
2. The absence of significant information across scales is predicted by exploiting the self-similarity inherent in images.
3. Successive approximation quantization is combined with arithmetic coding (see [59], [26], [35], [32]) to produce the compressed bit stream.

The second point above is arguably the most important. The EZW scheme encodes the wavelet transform coefficients in bit significance order. Normally, this would also require that the coefficient positions be explicitly coded. By predicting the absence of significant information across scales, however, the EZW scheme is able to avoid encoding individual coefficient positions explicitly. Not having to explicitly encode the position of each coefficient saves bits and leads to excellent compression results.

Although the EZW scheme was designed for use in lossy image coders, the method can also be applied to hybrid lossless/lossy image coders by replacing the transform used with a reversible wavelet transform.

4.7 Reversible Embedded Image Compression System

A reversible embedded image compression system based on the EZW coding scheme was used to obtain various results presented in the remainder of this thesis. This codec is a reasonably faithful reproduction of the system originally proposed by Shapiro with the following modifications/extensions:

1. The codec was extended to allow the use of M-band wavelet transforms (for arbitrary M). As originally proposed, the EZW scheme employs a 2-band wavelet transform for decorrelation. Extension to the M-band case is straightforward.
2. The codec was generalized to handle arbitrary-sized images (i.e., non-square images and images with non-M-adic dimensions).
3. Reversible wavelet transforms are employed for decorrelation purposes instead of a nonreversible linear wavelet transform. This change was made to facilitate reversible (i.e., lossless) compression.
4. The EZW coding scheme was only used to code the most significant bits (MSBs) of the transform coefficients. The three least significant bits (LSBs) of the transform coefficients were coded directly using a simple context model and entropy coding. In many cases, this leads to improved compression performance in lossless (or near-lossless) operation. This change typically has no affect on the bitstream generated for compression ratios greater than 8. Thus, in lossy operation, the bitstream remains completely embedded. This approach is similar to that used in the SPIHT codec of Said and Pearlman [36] and is described in more detail in [3].

For more details regarding the codec software, the reader is referred to Appendix C.

4.8 Wavelet Transforms for Image Compression

Wavelet transforms have proven extremely effective for transform-based image compression. Since many of the wavelet transform coefficients for a typical image tend to be very small or zero, these coefficients can be easily coded. Thus, wavelet transforms are a useful tool for image compression.

The main advantage of wavelet transforms over other more traditional decomposition methods (like the DFT and DCT) is that the basis functions associated with a wavelet decomposition typically have both long and short support. The basis functions with long support are effective for representing slow variations in an image while the basis functions with short support can efficiently represent sharp transitions (i.e., edges). This makes wavelets ideal for representing signals having mostly low-frequency components mixed with a relatively small number of sharp transitions. With more traditional transforms techniques like the DFT and DCT, the basis functions have support over the entire image, making it difficult to represent both slow variations and edges efficiently.

The wavelet transform decomposes a signal into frequency bands that are equally spaced on a logarithmic scale. The low-frequency bands have small bandwidths, while the high-frequency bands have large bandwidths. This logarithmic behavior of wavelet transforms can also be advantageous. Since human visual perception behaves logarithmically in many respects, the use of wavelet decompositions can sometimes make it easier to exploit characteristics of the human vision system in order to obtain improved subjective lossy compression results.

Although wavelet transforms with many different characteristics are possible, orthogonal transforms with symmetric finitely-supported basis functions are ideally most desirable for image compression. Orthogonality is beneficial as it ensures that transform coefficients do not become unreasonably large and also because it easily facilitates the selection of the most important transform coefficients in the sense of minimizing mean squared error. Symmetric basis functions are desirable in order to avoid undesirable phase distortion as a result of compression. If phase is not preserved, edges and lines can become severely distorted, resulting in poor subjective image quality. Moreover, the symmetric extension method for handling finite-length signals can only be applied to transforms with symmetric basis functions. This is yet another incentive for using transforms with symmetric basis functions.

Unfortunately, in the case of 2-band wavelet transforms, orthogonality, symmetry, and finite support can only be achieved in the trivial case of the Haar and other Haar-like transforms. For this reason, we usually choose to sacrifice orthogonality, and use near-orthogonal transforms instead. In practice, this concession does not pose any serious problems.

To date, most research has focused exclusively on 2-band wavelet systems. There are good reasons, however, for believing that M-band systems (where $M > 2$) may be able to yield improved results for image compression. First, in the M-band case there is the obvious advantage that it is possible to construct orthogonal transforms with symmetric finitely-supported basis functions. As mentioned above, this is not possible (except for trivial counterexamples) in the 2-band case. Also, Zou and Tewfik [61] have shown that for any finite energy signal, the M-band wavelet transform coefficients decrease exponentially fast with scale at a rate roughly proportional to M^{-kP} where k is the scale and the $M-1$ wavelets have P vanishing moments. Thus, M-band wavelets (with $M > 2$) yield better energy compaction than 2-band wavelets (for a fixed number of scales). Better energy compaction, however, may lead to transform coefficients that can be more efficiently coded. This suggests that M-band wavelets may be able to offer superior performance for image compression purposes.

4.9 Examples of Wavelet Transforms

In the last decade, many 2-band wavelet transforms have been developed that work quite well for image compression. In this thesis, several of the more performant wavelet transforms are considered, namely the Haar, TS, CDF22, CDF24, CDF97, V610, MIT97, and BCW3 transforms. In addition to these 2-band transforms, two 4-band transforms are also studied: the V4 and A4 transforms. A brief synopsis of all ten of these transforms is given in Table 4.1. In the case of the 2-band transforms, the notation x/y indicates that the lowpass and highpass analysis filters have lengths x and y, respectively. Similarly, the notation (x, y) indicates that the analysis and synthesis wavelet functions have x and y vanishing moments, respectively. The analysis filter coefficients for each of the 2-band transforms are given in Table 4.2. The reader is referred to the references cited in Table 4.1 for the filter coefficients for the V4 and A4 transforms. The synthesis scaling and wavelet functions for each of the ten transforms are shown in Figures 4.2 to 4.11. These plots will prove useful later as the types of artifacts obtained in lossy compression are determined largely by the shape of these functions.

Table 4.1. *Brief synopsis of wavelet transforms*

Name	Description	References
Haar	Haar Transform ↔ 2-band, orthogonal, symmetric, interpolating, 2/2, (1,1)	[19], [36], [60], [38]
TS	Two-Six Transform ↔ 2-band, biorthogonal, symmetric, 2/6, (3,1)	[60], [36], Transform 5 in [57]
CDF22	Cohen-Daubechies-Feauveau (2,2) Transform ↔ 2-band, biorthogonal, symmetric, interpolating, 5/3, (2,2)	Transform 4 in [57]
CDF24	Cohen-Daubechies-Feauveau (2,4) Transform ↔ 2-band, biorthogonal, symmetric, interpolating, 9/3, (2,4)	Table I in [6], Transform 6 in [57]
CDF97	Cohen-Daubechies-Feauveau 9/7 Transform ↔ 2-band, biorthogonal, symmetric, 9/7, (4,4)	Table 2 in [9]
V610	Villasenor 6/10 Transform ↔ 2-band, biorthogonal, symmetric, 6/10	Transform 3 in [57]
MIT97	MIT 9/7 Transform ↔ 2-band, biorthogonal, symmetric, interpolating, 9/7, (4,2)	Section 4 (Item 3) in [45]
BCW3	Biorthogonal Coifman Transform (Order 3) ↔ 2-band, biorthogonal, symmetric, interpolating, 13/7, (4,4)	Table 3.1 in [52], [58]
V4	Vaidyanathan 4-Band Transform ↔ 4-band, orthogonal, symmetric	Table II in [44]
A4	Alkin 4-Band Transform ↔ 4-band, orthogonal, symmetric	Table II in [5]

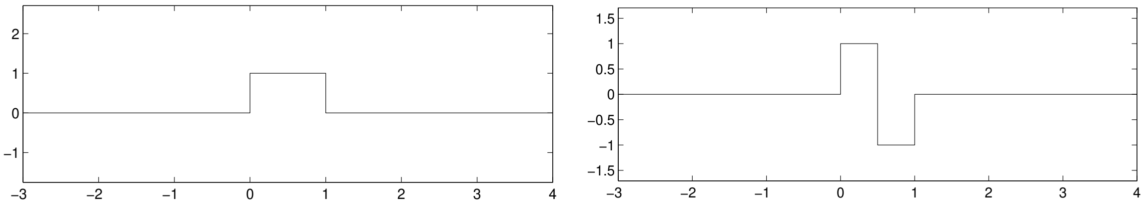
At this point, we note that the V4 transform has highly discontinuous synthesis scaling and

Table 4.2. *Analysis filters for 2-band transforms*

Haar	$\begin{cases} H_0(z) = \frac{1}{\sqrt{2}}(1+z^{-1}) \\ H_1(z) = \frac{-1}{\sqrt{2}}(1-z^{-1}) \end{cases}$
TS	$\begin{cases} H_0(z) = \frac{1}{\sqrt{2}}(1+z^{-1}) \\ H_1(z) = -\frac{1}{8\sqrt{2}}(1-z^{-5}) - \frac{1}{8\sqrt{2}}(z^{-1}-z^{-4}) + \frac{8}{8\sqrt{2}}(z^{-2}-z^{-3}) \end{cases}$
CDF22	$\begin{cases} H_0(z) = -\frac{1}{4\sqrt{2}}(1+z^{-4}) + \frac{2}{4\sqrt{2}}(z^{-1}+z^{-3}) + \frac{6}{4\sqrt{2}}z^{-2} \\ H_1(z) = \frac{1}{2\sqrt{2}}(1+z^{-2}) - \frac{2}{2\sqrt{2}}z^{-1} \end{cases}$
CDF24	$\begin{cases} H_0(z) = \frac{3\sqrt{2}}{128}(1+z^{-8}) - \frac{6\sqrt{2}}{128}(z^{-1}+z^{-7}) - \frac{16\sqrt{2}}{128}(z^{-2}+z^{-6}) + \frac{38\sqrt{2}}{128}(z^{-3}+z^{-5}) \\ \qquad + \frac{90\sqrt{2}}{128}z^{-4} \\ H_1(z) = \frac{\sqrt{2}}{4}(1+z^{-2}) - \frac{2\sqrt{2}}{4}z^{-1} \end{cases}$
CDF97	$\begin{cases} H_0(z) = 0.03783(1+z^{-8}) - 0.02385(z^{-1}+z^{-7}) - 0.1106(z^{-2}+z^{-6}) \\ \qquad + 0.3774(z^{-3}+z^{-5}) + 0.8527z^{-4} \\ H_1(z) = 0.06454(1+z^{-6}) - 0.04069(z^{-1}+z^{-5}) - 0.4181(z^{-2}+z^{-4}) \\ \qquad + 0.7885z^{-3} \end{cases}$
V610	$\begin{cases} H_0(z) = -0.1291(1+z^{-5}) + 0.04770(z^{-1}+z^{-4}) + 0.7885(z^{-2}+z^{-3}) \\ H_1(z) = -0.01891(1-z^{-9}) + 0.006989(z^{-1}-z^{-8}) + 0.06724(z^{-2}-z^{-7}) \\ \qquad + 0.1334(z^{-3}-z^{-6}) - 0.6151(z^{-4}-z^{-5}) \end{cases}$
MIT97	$\begin{cases} H_0(z) = \frac{\sqrt{2}}{64}(1+z^{-8}) - \frac{8\sqrt{2}}{64}(z^{-2}+z^{-6}) + \frac{16\sqrt{2}}{64}(z^{-3}+z^{-5}) + \frac{46\sqrt{2}}{64}z^{-4} \\ H_1(z) = -\frac{\sqrt{2}}{32}(1+z^{-6}) + \frac{9\sqrt{2}}{32}(z^{-2}+z^{-4}) - \frac{16\sqrt{2}}{32}z^{-3} \end{cases}$
BCW3	$\begin{cases} H_0(z) = -\frac{1}{256\sqrt{2}}(1+z^{-12}) + \frac{18}{256\sqrt{2}}(z^{-2}+z^{-10}) - \frac{16}{256\sqrt{2}}(z^{-3}+z^{-9}) \\ \qquad - \frac{63}{256\sqrt{2}}(z^{-4}+z^{-8}) + \frac{144}{256\sqrt{2}}(z^{-5}+z^{-7}) + \frac{348}{256\sqrt{2}}z^{-6} \\ H_1(z) = -\frac{1}{16\sqrt{2}}(1+z^{-6}) + \frac{9}{16\sqrt{2}}(z^{-2}+z^{-4}) - \frac{16}{16\sqrt{2}}z^{-3} \end{cases}$

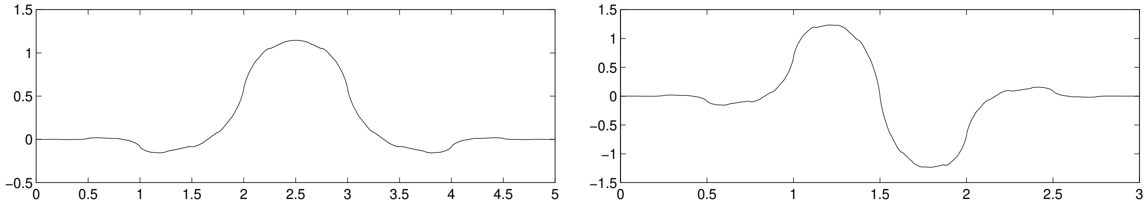
wavelet functions. The A4 transform is somewhat better behaved in this regard, but still the functions are somewhat rough. In the case of the 2-band transforms, the CDF97, V610, MIT97, and BCW3 transforms have very smooth synthesis scaling and wavelet functions, and the functions for the CDF22 and CDF24 transforms are continuous (although their first-order derivatives are discontinuous). The Haar transform has very sharp discontinuities in its scaling and wavelet functions, and the functions for the TS transform are rather rough.

It is important to note that the two 4-band transforms considered in this thesis serve primarily to provide a preliminary indication of the type of results one might hope to obtain with M-band transforms. There is little doubt that better M-band transforms (e.g., with smoother scaling functions) could be designed, but such work is beyond the intended scope of this research. Also, as a practical matter, some M-band transforms were required in order to demonstrate that the lifting-based



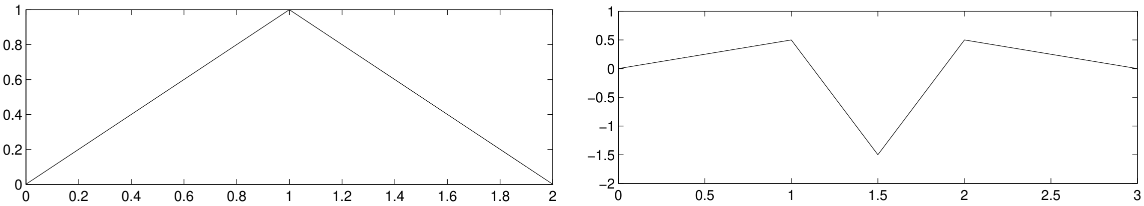
(a) (b)

Figure 4.2. *Synthesis scaling and wavelet functions for Haar transform. (a) Scaling function. (b) Wavelet function.*



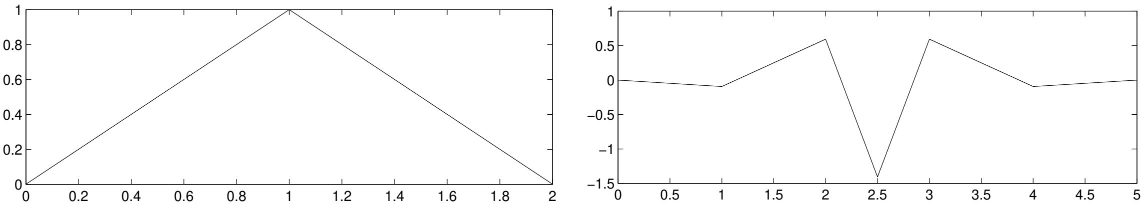
(a) (b)

Figure 4.3. *Synthesis scaling and wavelet functions for TS transform. (a) Scaling function. (b) Wavelet function.*



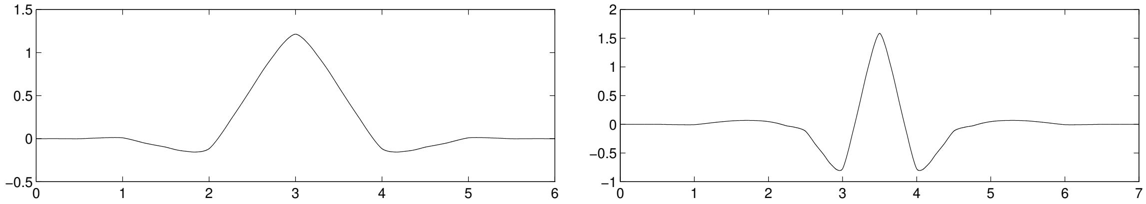
(a) (b)

Figure 4.4. *Synthesis scaling and wavelet functions for CDF22 transform. (a) Scaling function. (b) Wavelet function.*



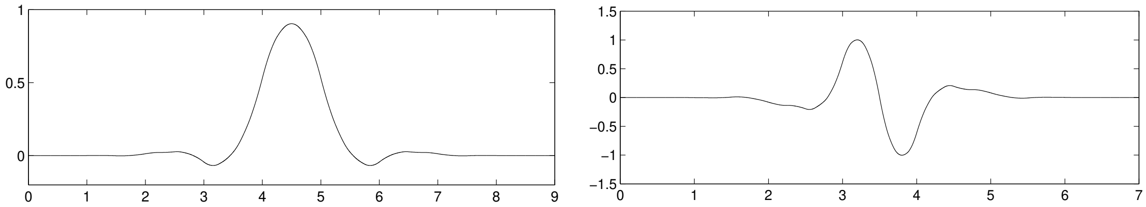
(a) (b)

Figure 4.5. *Synthesis scaling and wavelet functions for CDF24 transform. (a) Scaling function. (b) Wavelet function.*



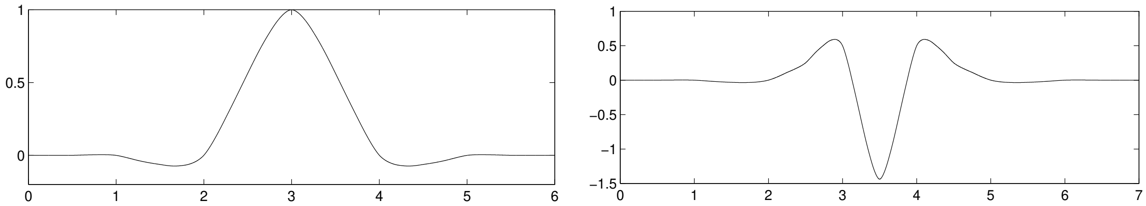
(a) (b)

Figure 4.6. *Synthesis scaling and wavelet functions for CDF97 transform. (a) Scaling function. (b) Wavelet function.*



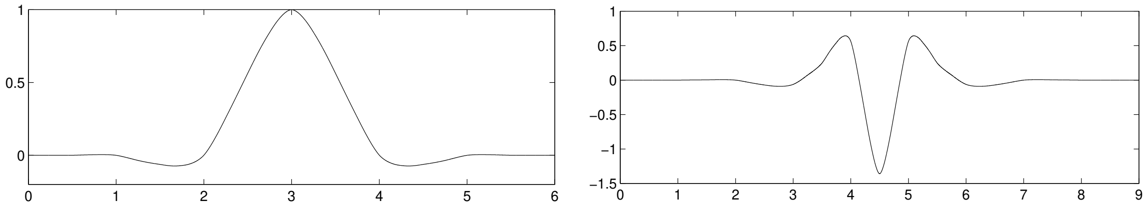
(a) (b)

Figure 4.7. *Synthesis scaling and wavelet functions for V610 transform. (a) Scaling function. (b) Wavelet function.*



(a) (b)

Figure 4.8. *Synthesis scaling and wavelet functions for MIT97 transform. (a) Scaling function. (b) Wavelet function.*



(a) (b)

Figure 4.9. *Synthesis scaling and wavelet functions for BCW3 transform. (a) Scaling function. (b) Wavelet function.*

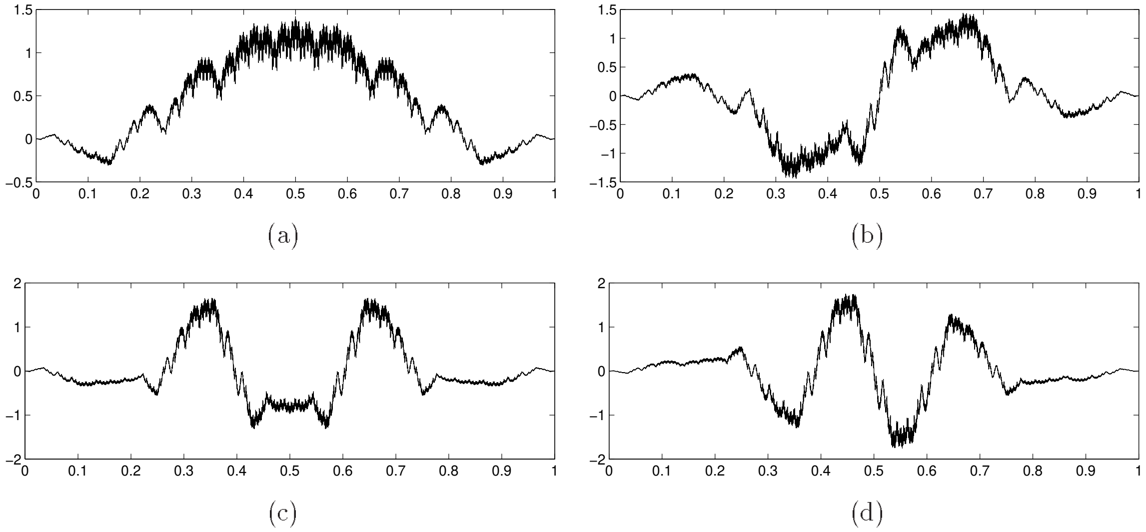


Figure 4.10. *Synthesis scaling and wavelet functions for V4 transform. (a) Scaling function. (b) First wavelet function. (c) Second wavelet function. (d) Third wavelet function.*

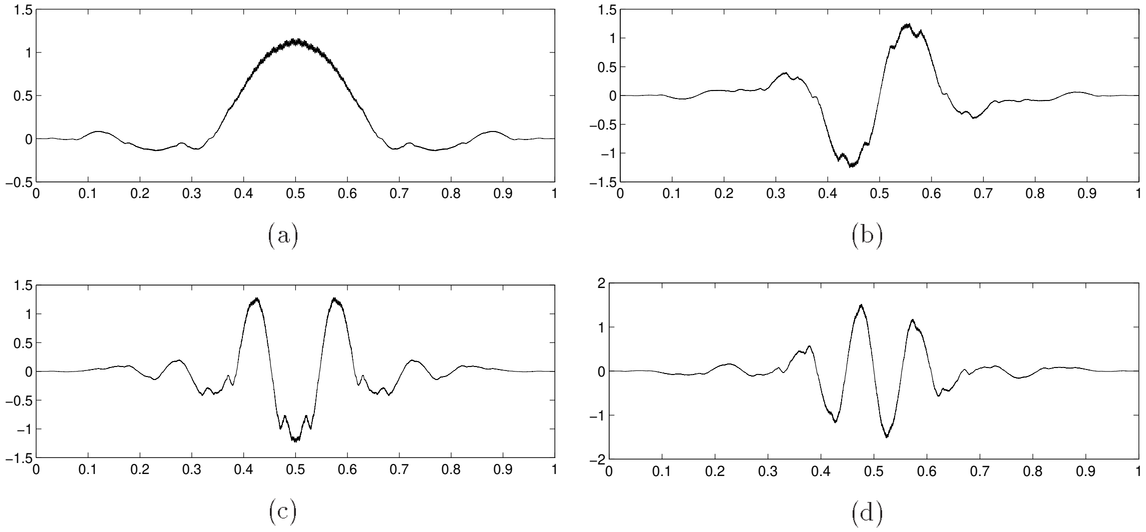


Figure 4.11. *Synthesis scaling and wavelet functions for A4 transform. (a) Scaling function. (b) First wavelet function. (c) Second wavelet function. (d) Third wavelet function.*

reversible transform design method is applicable to M-band transforms (where $M > 2$).

Although none of the transforms in Table 4.1 are reversible, the lifting-based technique described earlier can be used to construct reversible versions of these transforms as is considered in the next section.

4.10 Examples of Reversible Wavelet Transforms

At this point, we now construct reversible versions of each of the ten transforms discussed in the previous section. As described earlier, this is accomplished by computing a lifting factorization of the analysis polyphase matrix associated with each transform. The lifting factorizations corresponding to the various transforms are given in Tables 4.3 and 4.4. In the case of the 2-band transforms, the factorizations have been chosen to yield symmetry-preserving reversible transforms whenever possible. The reversible versions of the transforms follow directly from the lifting factorizations as described previously. Although some of these lifting factorizations have non-integer scaling factors, this is not a problem. We can simply omit the scaling operation during the transform calculation, and then assume an implicit per subband weighting of the transform coefficients. In instances where the weighting factors are $\sqrt{2}$ and $1/\sqrt{2}$, for the two-dimensional separable transform case, the per subband weights will be integer powers of two which is convenient from a computational perspective.

Table 4.3. *Lifting factorizations for 2-band transforms*

Haar†	$\boldsymbol{E}(z) = \mathcal{A}(1, 0; 1 - \sqrt{2}) \cdot \mathcal{A}(0, 1; \frac{1}{\sqrt{2}}) \cdot \mathcal{A}(1, 0; 1 - \sqrt{2})$
TS†	$\boldsymbol{E}(z) = \mathcal{S}(0; \sqrt{2}) \cdot \mathcal{S}(1; \frac{1}{\sqrt{2}}) \cdot \mathcal{A}(1, 0; -\frac{1}{4}[1 - z^{-2}]) \cdot \mathcal{A}(0, 1; \frac{1}{2}z) \cdot \mathcal{A}(1, 0; -z^{-1})$
CDF22‡	$\boldsymbol{E}(z) = \mathcal{S}(0; \sqrt{2}) \cdot \mathcal{S}(1; \frac{1}{\sqrt{2}}) \cdot \mathcal{A}(0, 1; \frac{1}{4}[1 + z^{-1}]) \cdot \mathcal{A}(1, 0; -\frac{1}{2}[z + 1])$
CDF24‡	$\boldsymbol{E}(z) = \mathcal{S}(0; \sqrt{2}) \cdot \mathcal{S}(1; \frac{1}{\sqrt{2}}) \cdot \mathcal{A}(0, 1; -\frac{3}{64}[z + z^{-2}] + \frac{19}{64}[1 + z^{-1}]) \cdot \mathcal{A}(1, 0; -\frac{1}{2}[1 + z])$
CDF97‡	$\boldsymbol{E}(z) = \mathcal{S}(0; 1.1496044) \cdot \mathcal{S}(1; 0.86986445)$ $\cdot \mathcal{A}(0, 1; 0.44350685[1 + z^{-1}]) \cdot \mathcal{A}(1, 0; 0.88291108[z + 1])$ $\cdot \mathcal{A}(0, 1; -0.052980119[1 + z^{-1}]) \cdot \mathcal{A}(1, 0; -1.5861343[z + 1])$
V610†	$\boldsymbol{E}(z) = \mathcal{S}(0; 1.4142136) \cdot \mathcal{S}(1; -0.70710678) \cdot \mathcal{A}(1, 0; -0.29306679[z - z^{-1}] - 0.39453543)$ $\cdot \mathcal{A}(0, 1; 0.87491869) \cdot \mathcal{A}(1, 0; -0.090074735z - 0.27022385)$ $\cdot \mathcal{A}(0, 1; -0.42797992 - 0.11953214z^{-1}) \cdot \mathcal{A}(1, 0; -0.36953625)$
MIT97‡	$\boldsymbol{E}(z) = \mathcal{S}(0; \sqrt{2}) \cdot \mathcal{S}(1; \frac{1}{\sqrt{2}}) \cdot \mathcal{A}(0, 1; \frac{1}{4}[1 + z^{-1}]) \cdot \mathcal{A}(1, 0; \frac{1}{16}[z^2 + z^{-1}] - \frac{9}{16}[z + 1])$
BCW3‡	$\boldsymbol{E}(z) = \mathcal{S}(0; \sqrt{2}) \cdot \mathcal{S}(1; \frac{1}{\sqrt{2}}) \cdot \mathcal{A}(0, 1; -\frac{1}{32}[z + z^{-2}] + \frac{9}{32}[1 + z^{-1}])$ $\cdot \mathcal{A}(1, 0; \frac{1}{16}[z^2 + z^{-1}] - \frac{9}{16}[z + 1])$

† type 1/2 polyphase system

‡ type 3/4 polyphase system

At this point, we again point out that these reversible transforms are nonlinear. Consequently, we cannot speak of basis functions in a strict mathematical sense. In spite of this, however, the scaling and wavelet functions still continue to be quite useful. Because these transforms well approximate

Table 4.4. *Lifting factorizations for 4-band transforms*

V4†	$$\begin{cases} \boldsymbol{E}(z) = \mathcal{A}(1,3;0.23529533z)\cdot\mathcal{A}(3,1;1.3819615-0.22295185z^{-1})\cdot\mathcal{A}(0,2;-0.23529533z) \\ \quad\cdot\mathcal{A}(2,3;-0.68564896)\cdot\mathcal{A}(2,0;0.64968013+0.22295185z^{-1}) \\ \quad\cdot\mathcal{A}(2,3;-1.6691273)\cdot\mathcal{A}(3,2;0.59911547)\cdot\mathcal{A}(0,1;-1.4584722) \\ \quad\cdot\mathcal{A}(0,2;-0.72011243)\cdot\mathcal{A}(1,2;-0.29432479)\cdot\mathcal{A}(2,1;1)\cdot\mathcal{A}(1,2;1.4343758) \\ \quad\cdot\mathcal{A}(2,1;-0.41078290)\cdot\mathcal{A}(2,0;0.83456365)\cdot\mathcal{A}(0,2;-1.1982309)\cdot\mathcal{A}(1,2;1) \\ \quad\cdot\mathcal{A}(2,3;-0.36818635)\cdot\mathcal{A}(1,3;-1.5664173)\cdot\mathcal{A}(0,3;1.8215658) \end{cases}$$
A4†	$$\begin{cases} \boldsymbol{E}(z) = \mathcal{A}(1,3;-1.7651533)\cdot\mathcal{A}(3,1;-0.41590824z^{-1}+0.10354399z^{-2}) \\ \quad\cdot\mathcal{A}(0,2;-0.56652300z)\cdot\mathcal{A}(1,3;1.1837342+0.94396953z) \\ \quad\cdot\mathcal{A}(3,1;0.13754027-0.79463855z)\cdot\mathcal{A}(2,0;-0.32261889z^{-1}+0.42887598) \\ \quad\cdot\mathcal{A}(0,2;0.37991750+0.47641511z)\cdot\mathcal{A}(2,0;-0.52451193z^{-1}-0.27252247z^{-2}) \\ \quad\cdot\mathcal{A}(3,2;0.71041770)\cdot\mathcal{A}(3,1;1.0750472)\cdot\mathcal{A}(1,3;-0.93019170) \\ \quad\cdot\mathcal{A}(0,3;0.41745501)\cdot\mathcal{A}(0,2;0.93213776)\cdot\mathcal{A}(2,3;2.6856408) \\ \quad\cdot\mathcal{A}(3,2;-0.65408237)\cdot\mathcal{A}(2,3;3.5494754)\cdot\mathcal{A}(3,0;1)\cdot\mathcal{A}(0,3;-0.24336963) \\ \quad\cdot\mathcal{A}(3,0;-0.32164931)\cdot\mathcal{A}(3,1;-1.1916059)\cdot\mathcal{A}(2,1;-1)\cdot\mathcal{A}(0,1;1.1916059) \end{cases}$$

† type 1/2 polyphase system

their parent linear transforms, they still behave very much like linear transforms with the same set of basis functions.

4.11 Transform Performance Evaluation Methodology

In order to assess which of the ten new reversible transforms presented in Section 4.10 are most effective for image compression, they were employed in a real image coding system. The system used for evaluation was a reversible embedded image codec based on the EZW coding scheme (as described in Section 4.7). Using each of the new transforms in the coder, approximately 75 grayscale images of different types and dimensions were compressed in both a lossy and lossless manner. The results obtained with each transform were then examined. In the case of lossy compression, performance was evaluated using the PSNR metric and further supplemented by subjective image quality tests. As usual, lossless compression performance was assessed based on the compression ratio measure. To gain further insight into the effectiveness of the new transforms, the results obtained with these transforms were also compared to those obtained using the S+P transform, a transform considered to be state-of-the-art by many.

In the next section, we present a representative subset of the compression results obtained with the new reversible transforms. Various compression results for about two dozen test images are considered. For the most part, the images employed were taken from standard test sets. More information about the test images is available in Appendix B. This appendix provides details for each image such as: the origin of the image when known, its dimensions and depth, a textual description of the image, and in some cases a depiction of the image itself.

4.12 Transform Performance Evaluation

First, we present the results obtained with the new 2-band reversible transforms. Table 4.5 shows the results obtained in the lossless compression of 25 test images for each of the transforms under consideration. For comparison purposes, results for the S+P transform are also provided. As a visual aid, the best result for each image has been highlighted. Evidently, the best results tend to be obtained from the Haar, CDF22, MIT97, BCW3, and S+P transforms. Which transform is best for a particular image, however, depends on the characteristics of the image in question. In particular, these results indicate that uniformity and smoothness of intensity variations in an image largely determine which transform is most effective.

For images characterized by smooth intensity variations and a lack of very large regions of uniform intensity, e.g., photographs, x-rays, the BCW3, S+P, and MIT97 transforms tend to perform quite well, followed by the CDF97 and V610 transforms. Of the test images employed, `ct`, `finger`, `lena`, and `xray1` are good examples of images of this type. Although the BCW3, S+P, and MIT97 transforms are relatively close in terms of performance, the BCW3 and S+P transforms appear to be marginally better. Since the BCW3 transform is also nonexpansive, it is more desirable than the S+P transform when arbitrary-sized images are involved. It is not surprising that the BCW3 and MIT97 transforms are effective for very smooth images. Their synthesis scaling and wavelet functions shown in Figures 4.9 and 4.8 are also very smooth.

For images with moderate amounts of nonsmooth variation, e.g., some computer generated images, and smoothly varying images with very large regions of uniform intensity, e.g., a photograph with a large proportion of sky, the CDF22 transform performs very well, and the CDF24 and TS transforms yield reasonable but less impressive results. Of the test images used, `chart_s` is a good example of the first type of image, and `airplane` and `molecule` are good examples of the second type. The CDF22 and CDF24 transforms have continuous, piecewise linear synthesis scaling and wavelet functions as shown in Figures 4.4 and 4.5. Thus, one would expect them to work well for moderately smooth images. Due to the discontinuities in the first-order derivatives of these functions, however, the CDF22 and CDF24 transforms are better able to represent images with sharp transitions than transforms with smoother basis functions. The TS transform is associated with synthesis scaling and wavelet functions that have no large jumps but are somewhat rough. Thus, it also performs reasonably well for images in this class.

For images characterized by much nonsmooth variation and large regions of uniform intensity, e.g., some computer generated images, images consisting mostly of text, and some compound images, the Haar transform is quite effective. Of the test images employed, `cmpnd1`, `cmpnd2`, `eafbcmpnd`, `express1`, `france`, and `library` are good examples of such images. Again, this behavior is con-

Table 4.5. *Lossless compression results for 2-band transforms*

Image	CR								
	Haar	TS	CDF22	CDF24	CDF97	V610	MIT97	BCW3	S+P
air1	1.403	1.447	1.468	1.461	1.468	1.464	1.476	1.476	1.480
air2	1.726	1.810	1.869	1.860	1.819	1.796	1.891	1.889	1.881
airplane	2.317	2.326	2.409	2.401	2.264	2.318	2.385	2.389	2.389
barb	1.556	1.657	1.686	1.692	1.710	1.708	1.724	1.733	1.712
bike3	1.651	1.688	1.728	1.719	1.706	1.695	1.725	1.725	1.715
chart_s	2.320	2.329	2.433	2.412	2.272	2.283	2.388	2.385	2.402
cmpnd1	4.490	3.559	3.714	3.407	2.946	2.988	3.103	3.041	3.504
cmpnd2	4.272	3.457	3.713	3.395	2.949	2.853	3.117	3.057	3.431
cr	1.864	1.909	1.930	1.928	1.919	1.924	1.924	1.928	1.911
ct	2.495	2.906	2.975	2.941	3.037	2.991	3.191	3.179	3.120
eafbcmpnd	2.445	1.701	1.851	1.706	1.475	1.480	1.524	1.501	1.697
express1	3.133	2.239	2.515	2.388	2.003	1.953	2.187	2.161	2.263
finger	1.272	1.431	1.439	1.432	1.451	1.432	1.477	1.475	1.445
france	5.328	3.245	3.759	3.452	2.527	2.561	2.945	2.900	3.228
gold	1.588	1.633	1.675	1.670	1.655	1.658	1.675	1.676	1.662
hotel	1.623	1.639	1.687	1.684	1.667	1.665	1.686	1.687	1.672
lax	1.358	1.358	1.371	1.368	1.366	1.368	1.366	1.368	1.360
lena	1.740	1.835	1.875	1.870	1.875	1.859	1.891	1.893	1.881
library	1.460	1.371	1.401	1.382	1.337	1.337	1.352	1.347	1.359
mandrill	1.283	1.315	1.329	1.327	1.330	1.331	1.333	1.335	1.332
medical_93	2.588	2.768	2.941	2.920	2.706	2.775	3.004	2.998	2.953
molecule	3.903	3.808	4.030	4.004	3.610	3.445	3.927	3.930	3.960
mri	1.722	1.809	1.845	1.838	1.850	1.860	1.879	1.879	1.885
us	2.567	2.424	2.539	2.484	2.202	2.253	2.429	2.415	2.507
xray1	2.696	2.857	3.103	3.088	2.812	2.751	3.156	3.155	3.092

sistent with the synthesis scaling and wavelet functions associated with the Haar transform. These functions have sharp discontinuities allowing the Haar transform to represent images with many sharp transitions.

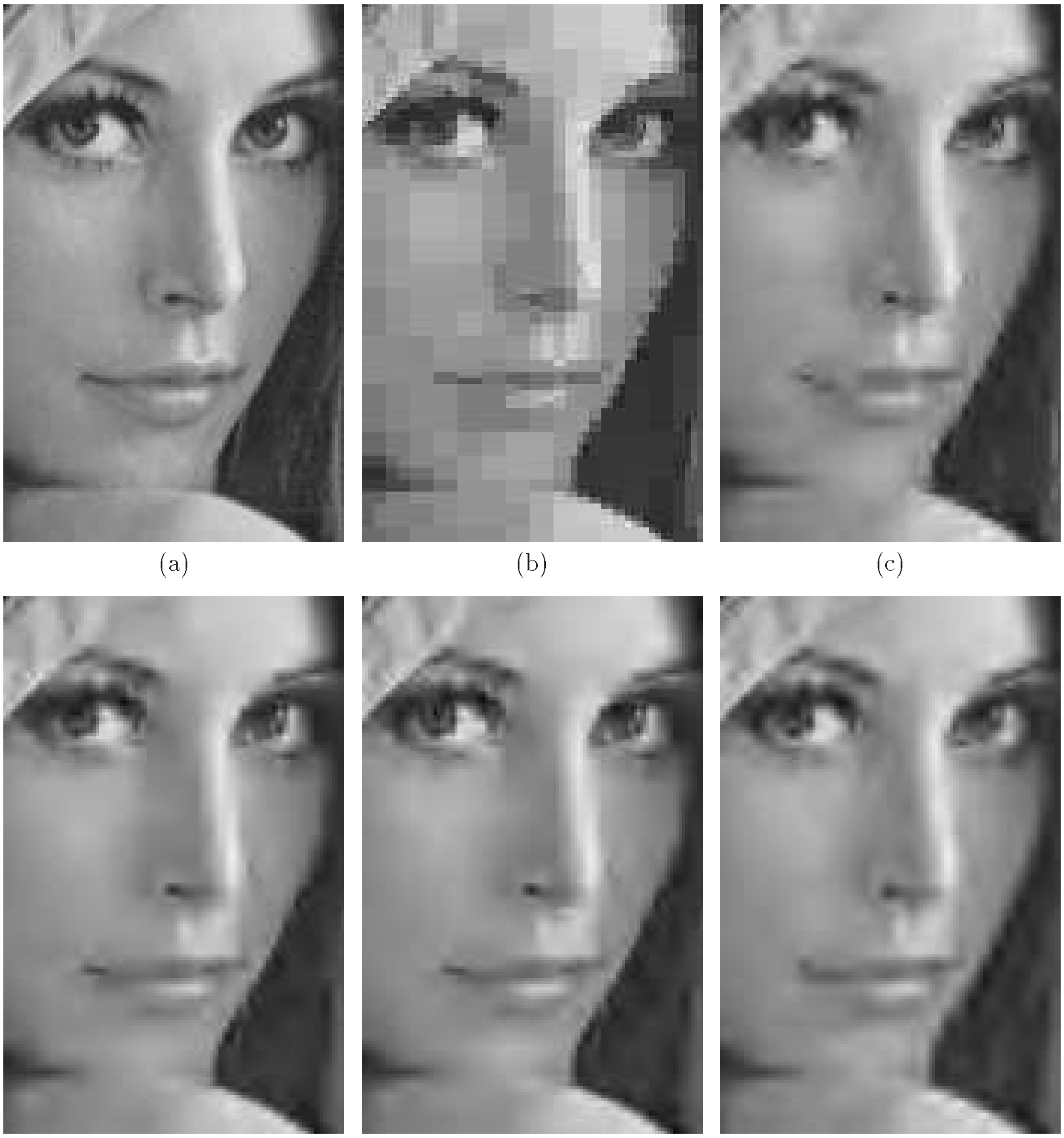
Table 4.6 shows lossy compression results for six of the test images. Again, the best result for each image has been highlighted. As can be seen, the above observations on transform effectiveness also hold for the most part in the case of lossy compression. Although at very low bit rates, i.e., very high compression ratios, the best transform in terms of PSNR is difficult to predict, subjective testing usually indicates little perceived difference in image quality in these cases since the distortion is so high. For these reasons, we are mainly concerned with the lossless compression results in our performance evaluation.

To further evaluate the lossy compression results, subjective image quality tests were performed.

Table 4.6. *Lossy compression results for 2-band transforms*

Image	CR	PSNR (dB)								
		Haar	TS	CDF22	CDF24	CDF97	V610	MIT97	BCW3	S+P
barb	8	32.78	34.63	34.40	34.61	35.28	35.13	35.15	35.34	34.89
	16	28.69	29.98	30.21	30.47	30.62	30.81	30.62	30.91	30.13
	32	25.43	26.36	26.46	26.76	26.93	26.99	26.59	26.83	26.36
	64	23.56	24.13	24.04	24.17	24.77	24.22	24.08	24.23	24.38
	128	22.60	23.19	23.40	23.51	23.39	23.49	23.27	23.39	23.27
chart_s	8	40.22	40.69	41.30	41.27	40.07	40.29	41.04	41.11	40.95
	16	33.21	34.17	34.02	33.81	34.98	33.88	34.30	34.39	35.09
	32	28.52	29.80	29.41	29.46	30.43	29.73	29.81	29.90	30.40
	64	25.05	26.98	26.62	26.65	27.53	26.72	26.97	27.07	27.42
	128	23.13	24.45	23.92	24.03	24.35	24.08	24.40	24.51	24.32
cmpnd1	8	45.98	41.01	41.31	40.74	39.40	39.46	39.88	39.73	40.97
	16	36.61	32.80	30.75	30.36	31.96	31.40	30.67	30.61	32.16
	32	25.42	25.06	24.43	24.43	24.98	24.50	24.29	24.41	24.75
	64	21.19	20.91	20.45	20.58	21.15	20.40	20.54	20.64	20.78
	128	18.32	18.60	19.42	19.66	19.23	19.34	19.43	19.45	18.90
finger	8	26.56	30.70	29.63	29.56	31.01	29.97	30.94	31.00	30.33
	16	23.74	26.93	26.08	26.17	27.06	26.55	26.80	26.94	26.81
	32	22.08	23.74	22.92	23.02	23.44	23.43	23.41	23.55	23.40
	64	20.28	21.02	21.21	21.20	21.16	21.74	21.36	21.53	21.04
	128	19.51	20.03	19.49	19.67	19.83	19.83	19.40	19.54	20.05
lena	8	37.07	38.76	39.03	39.02	38.51	38.63	39.15	39.22	39.02
	16	33.34	35.49	36.00	36.04	35.66	35.78	36.15	36.23	35.79
	32	30.50	32.31	33.03	33.14	32.79	32.93	33.17	33.27	32.84
	64	27.98	29.36	30.21	30.41	29.91	30.19	30.24	30.35	30.02
	128	25.59	26.95	27.52	27.72	27.54	27.49	27.45	27.57	27.60
molecule	8	43.49	48.43	47.86	47.85	44.89	45.31	47.94	47.97	48.24
	16	39.60	43.66	44.25	44.24	42.24	42.24	44.08	44.09	43.69
	32	35.12	40.14	41.00	41.00	39.94	38.94	41.05	41.09	40.66
	64	31.42	36.25	37.64	37.30	36.54	35.86	37.59	37.70	36.81
	128	28.23	32.55	33.04	33.01	33.00	32.32	34.00	33.99	32.86

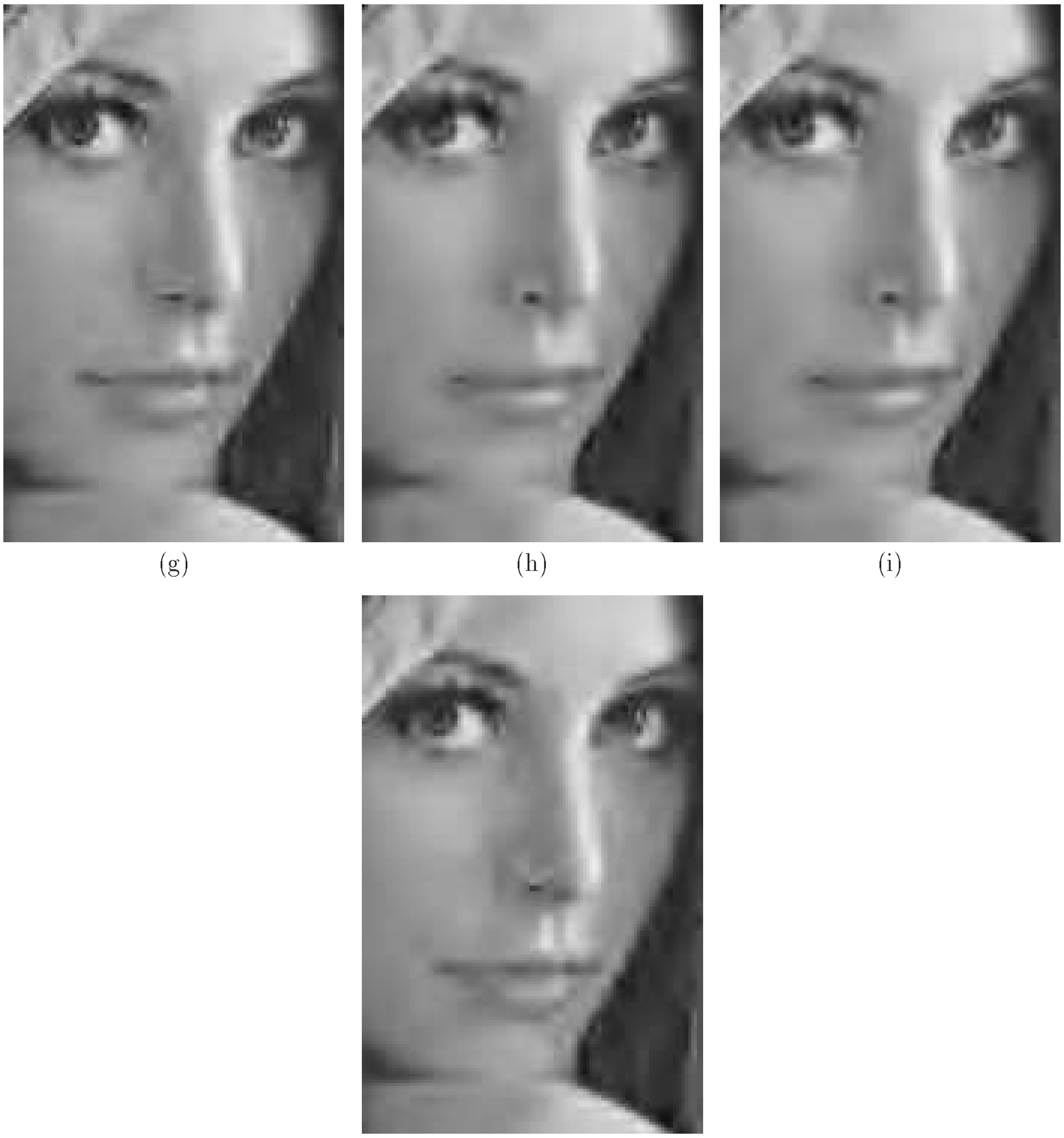
Of the transforms under consideration, no clear winner was identified. In most cases, the CDF22, CDF24, CDF97, V610, MIT97, and BCW3 transforms produce comparable results. The notable exceptions, however, are the Haar and TS transforms. The Haar transform often produces very disturbing blocking artifacts. Also, although not as bad as the Haar transform, the TS transform was frequently found to produce undesirable blocking artifacts. To illustrate the types of artifacts obtained with the various transforms, an example is now given. The `lena` image was compressed at a ratio of 64:1 using each of the transforms under consideration, and then reconstructed with the results shown in Figure 4.12. Only a portion of the image is shown in each case (enlarged slightly) in order that the differences between the results be more clearly visible. Notice the severe blocking artifacts obtained in the case of the Haar transform. Blocking artifacts are less pronounced in the TS transform case, but they are still clearly visible.



(a) (b) (c)

(d) (e) (f)

Figure 4.12. *Lossy compression example for 2-band transforms. (a) Original image. Lossy reconstruction at compression ratio of 64:1 using (b) Haar transform, (c) TS transform, (d) CDF22 transform, (e) CDF24 transform, and (f) CDF97 transform.*



(g) (h) (i)

(j)

Figure 4.13. *Lossy compression example for 2-band transforms (continued). Lossy reconstruction at compression ratio of 64 : 1 using (g) V610 transform, (h) MIT97 transform, (i) BCW3 transform, and (j) S+P transform.*

Now, we consider the two 4-band transforms (i.e., V4 and A4). Table 4.7 shows the lossless compression results for six test images obtained with each of the two transforms. For each image, the best result has been highlighted. Evidently, the V4 transform is best for images with much nonsmooth intensity variation while the A4 transform is more effective for smooth images. These results are, again, consistent with the shape of the synthesis scaling and wavelet functions for the two transforms. The synthesis scaling and wavelet functions for the V4 transform are highly discontinuous and consequently the V4 transform is best suited to images with many sharp intensity changes (like `cmpnd1` and `eafbcmpnd`). Similarly, the A4 transform has relatively smoother scaling and wavelet functions, and is therefore more effective for smooth images (like `barb`, `finger`, `lena`, and `molecule`). Comparing the results in Tables 4.5 and 4.7, we can see that the two 4-band transforms are not as good as the best 2-band transforms.

Table 4.7. *Lossless compression results for 4-band transforms*

Image	CR V4	CR A4
`barb`	1.606	1.666
`cmpnd1`	2.699	2.328
`eafbcmpnd`	1.356	1.290
`finger`	1.409	1.430
`lena`	1.716	1.779
`molecule`	2.847	2.955

Table 4.8 shows the results obtained for the lossy compression of the same six images used above. Again, the best result for each image has been highlighted. Evidently, these lossy results tend to correlate reasonably well with the lossless compression results presented above.

Table 4.8. *Lossy compression results for 4-band transforms*

Image	CR	PSNR (dB) V4	PSNR (dB) A4
`barb`	8	34.02	35.75
	16	30.07	31.31
	32	26.67	27.62
	64	24.17	25.17
	128	23.00	23.72
`cmpnd1`	8	36.81	36.51
	16	28.51	28.45
	32	23.35	23.35
	64	20.63	20.73
	128	18.58	18.73
`eafbcmpnd`	8	22.97	22.91
	16	17.81	17.67
	32	14.15	14.33
	64	13.21	13.33
	128	12.17	12.32

Image	CR	PSNR (dB) V4	PSNR (dB) A4
`finger`	8	29.84	30.67
	16	25.80	26.65
	32	23.09	23.95
	64	20.90	21.47
	128	19.88	20.28
`lena`	8	36.41	38.05
	16	33.34	35.36
	32	30.71	32.32
	64	28.10	29.36
	128	25.72	26.91
`molecule`	8	41.62	42.55
	16	37.28	41.67
	32	33.03	38.95
	64	30.64	34.98
	128	28.18	31.63

In terms of subjective lossy compression performance, the A4 transform was found to perform better than the V4 transform. The oscillations in the scaling and wavelet functions associated with the V4 transform lead to very noticeable checkerboarding artifacts. To demonstrate the quality of lossy reconstructions obtained with these two transforms an example is now presented. The `lena` image was compressed at a ratio of 64:1 using both the V4 and A4 transforms, and then reconstructed yielding the results shown in Figure 4.14. The checkerboarding artifacts obtained from the V4 transform are clearly visible.

(a) (b) (c)

Figure 4.14. *Lossy compression example for 4-band transforms. (a) Original image. Lossy reconstruction at compression ratio of 64 : 1 using (b) V4 transform, and (c) A4 transform.*

4.13 Full Embedding versus Partial Embedding

The image compression system discussed in Section 4.7 always produces a partially embedded bitstream. That is, all but the three LSBs of the transform coefficients are coded using EZW, and then the remaining bits are coded in a very simple non-embedded fashion. In this section, we compare the performance of this scheme to the fully embedded case. As suggested previously, the partially embedded approach is not always the best, but in many cases it leads to improved compression performance in lossless (or near-lossless) operation.

Recall that in some cases, reversible transforms compute scaled versions of near-orthogonal transforms. In these instances, the transform coefficients must be weighted on a per subband basis in order to correspond to a near-orthogonal transform. Since it is the weighted coefficients that are actually coded, some subbands can never have coefficients with nonzero bits in their three LSB positions.

For purposes of analysis, three different reversible transforms were considered: Haar, CDF22, and BCW3. Table 4.9 shows the lossless compression results obtained for six images using both the partially embedded and fully embedded coding schemes. In each case, the best result (from either partial or full embedding) is highlighted. For all three transforms, partial embedding yields the best results for the `barb`, `finger`, and `lena` images while full embedding is most effective for the `chart_s`, `cmpnd1`, and `molecule` images. From this we can see that it is the image, and not the transform employed, that largely determines whether partial or full embedding performs best. Fortunately, there is a simple explanation for the above results. To see it, however, we must observe one key difference between the two methods.

It is important to note that the choice of partial or full embedding only affects the rate-distortion behavior at very high bit rates—typically at compression ratios less than 8. Since we are rarely interested in lossy compression at such low compression ratios, the difference in lossy results is of little concern.

In both the partially and fully embedded cases, the first part of the bitstream is coded using the EZW scheme. The difference in the two approaches is what happens once the third LSB of the transform coefficients is reached. In the fully embedded case, nothing changes, and the EZW scheme is used to code the rest of the coefficient bits. With the partially embedded approach, the remaining bits of the transform coefficients are coded explicitly. There is one fundamental difference between the two approaches: When partial embedding is employed, all of the remaining (i.e., uncoded) transform coefficient bits must be coded explicitly. In the case of full embedding, however, not all of the remaining bits necessarily need to be coded. This is because EZW only ever codes nonzero coefficients. Thus, if a large number of the remaining bits correspond to zero-valued coefficients, very little additional information needs to be output. Also, EZW has the property that it can very efficiently encode trees of zero bits. Thus, even if the remaining uncoded bits correspond to nonzero coefficient values, so long as these bits are mostly zero, EZW can encode them extremely efficiently. Thus, one might speculate that full embedding would perform better in cases where the three LSBs of the transform coefficients tend to be zero. Otherwise, the partially embedded approach would likely be more effective.

As will now be demonstrated, numerical results confirm the above hypothesis. In other words, full embedding does perform better than partial embedding in cases where the three LSBs of the transform coefficients are mostly all zero. Table 4.10 shows how many of the transform coefficients have their three LSBs equal to zero for each image/transform pair. Clearly, the images for which full embedding is best (i.e., `chart_s`, `cmpnd1`, `molecule`) have a large percentage of transform coefficients with their three LSBs all zero, typically, more than 40%. The other images (i.e., `barb`, `finger`, `lena`) for which partial embedding is more effective have a much lower percentage of such coefficients. This suggests a further refinement for the image coder. One could easily choose between full and partial embedding on the basis of how many transform coefficients have their three LSBs all equal to zero. If more than 40% of the coefficients satisfy this condition, full embedding would be selected; otherwise, partial embedding would be used. Although one additional bit in the output stream is required to differentiate between the partially and fully embedded cases, the potential savings are far more significant.

Table 4.9. *Lossless compression results for full versus partial embedding*

	CR					
	Haar		CDF22		BCW3	
Image	Full	Partial	Full	Partial	Full	Partial
barb	1.548	1.556	1.669	1.686	1.715	1.733
chart_s	2.366	2.320	2.449	2.433	2.395	2.385
cmpnd1	4.819	4.490	3.817	3.714	3.074	3.041
finger	1.270	1.272	1.435	1.439	1.472	1.475
lena	1.730	1.740	1.858	1.875	1.874	1.893
molecule	4.130	3.903	4.231	4.030	4.097	3.930

Table 4.10. *Fraction of transforms coefficients with all three LSBs equal to zero*

	"Zero" Coefficients (%)		
Image	Haar	CDF22	BCW3
barb	14.84	26.24	26.45
chart_s	44.78	50.14	48.05
cmpnd1	77.33	78.55	70.72
finger	12.52	25.34	25.35
lena	15.76	26.57	26.33
molecule	72.25	69.43	67.59

4.14 Periodic Extension versus Symmetric Extension

Periodic extension and symmetric extension were previously discussed as a means for handling finite-length signals. In the case of linear transforms, symmetric extension is known to generally yield better compression results than periodic extension. As one might expect, this behavior carries over to nonlinear reversible transforms. Since reversible transforms well approximate their parent linear transforms, this behavior is inherited from the linear case. Experimental results to support this assertion will now be presented.

For the purposes of this analysis, two symmetry-preserving reversible transforms were considered: CDF22 and BCW3. These transforms were used in conjunction with both periodic extension and symmetric extension to compress six test images in a lossy and lossless manner. The compression results were then used to assess the relative merits of the two extension techniques.

The six test images used consisted of the following: `barb`, `chart_s`, `cmpnd1`, `finger`, `lena`, and `molecule`. This set of images was chosen so that both smooth and nonsmooth image types were represented, and also so that some images had dimensions that would result in expansive transforms for the periodic extension case. Of the various images, `chart_s` and `molecule` result in expansive transforms in the periodic extension case while the others are nonexpansive.

At this point we recall the two primary advantages of symmetric extension over periodic extension:

1. Symmetric extension does not introduce jump discontinuities in the extended signal, while periodic extension does have the potential to do this. Such discontinuities are undesirable as they increase the amount of high frequency energy in the signal.
2. Symmetric extension allows for nonexpansive transforms for arbitrary length signals while periodic extension can result in expansive transforms.

Thus, there are two reasons we might expect to obtain better compression results with symmetric extension. Considering the test images used, we would expect that the smooth images (i.e., `barb`, `finger`, `lena`, `molecule`) would benefit from point (1) above, while the images that have "bad" dimensions (i.e., `chart_s` and `molecule`) would benefit from point (2). With these observations in mind, we now examine the compression results obtained with the two different extension methods.

The results for the lossless compression of the test images are given in Table 4.11. Symmetric extension performs significantly better than periodic extension for all of the test images except the `cmpnd1` image where the two methods yield essentially the same result.

The `chart_s` image is not particularly smooth and has an expansive transform in the periodic extension case. Thus, we would expect this image to benefit the most from point (2) above. Similarly, the `lena` image is very smooth, and does not have an expansive transform. Thus, we would expect this image to benefit mostly from point (1) above. Clearly, however, the `lena` image benefits much more from symmetric extension than the `chart_s` image. Therefore, it would appear that expansiveness does not significantly affect compression performance. In fact, smooth images tend to benefit much more from the use of symmetric extension than images with dimensions that lead to expansive transforms.

Lossy compression results for the same six test images are shown in Table 4.12. In the vast majority of cases, the results obtained with symmetric extension are either clearly better than, or comparable to those obtained with periodic extension. Again, we observe that smooth images such as `lena` tend to benefit much more from symmetric extension than images that have expansive transforms in the periodic extension case.

In terms of subjective image quality, symmetric extension very frequently leads to better results than periodic extension at low bit rates. In particular, the use of periodic extension often leads to disturbing artifacts at the edges of the reconstructed image. To demonstrate this effect, the `lena` image was reconstructed at a compression ratio of 64:1 using both periodic extension and symmetric extension. The results are shown in Figure 4.15. In order for the artifacts to be clearly visible, only part of the right edge of the image is shown under magnification. Clearly, in the periodic extension case we have a very annoying artifact that is not present at all in the symmetric extension case. This poor behavior at image edges is typical of periodic extension.

Smooth images would tend to benefit most from the first advantage of symmetric extension given above. Images with "bad" dimensions (i.e., dimensions that result in an expansive transform with periodic extension) benefit from the second advantage. Clearly, from the results we can see that smooth images (e.g., `lena`) tend to benefit the most from symmetric extension regardless of the expansiveness issue. One can therefore conclude that from a compression performance standpoint, expansiveness is not a serious problem. The introduction of undesirable high frequency components by periodic extension is far more problematic.

One further issue concerning expansiveness, however, must not be overlooked. Although expansiveness does not significantly degrade compression performance, it does have a number of detrimental effects on coder complexity. When periodic extension is used, additional rows and columns are added on an "as needed" basis in order to handle the expansive cases. This, however, results in padding entries in the matrix of transform coefficients. Due to these padding elements, the coefficients for the various subbands are more difficult to locate, and coder complexity suffers. Of course, one could instead perform all padding on the image initially, but this also leads to much more expansive transforms and is therefore undesirable. In addition, expansive transforms cannot generally be calculated in place (at least in a strict sense). Therefore, we have a compelling argument for wanting to use symmetric extension. By avoiding all of the complications caused by expansiveness, we are able to build much simpler image coders. Of course, all other things being equal, a less complex coder will also be faster.

Therefore, we conclude that even in the case of (nonlinear) reversible transforms, symmetric extension is clearly superior to periodic extension. Since symmetric extension can only be used in conjunction with symmetry-preserving transforms, this provides a very strong motivation for the use of such transforms. In fact, aside from the Haar transform (which is so simple to compute), it is hard to justify the use of any other reversible transform that cannot be used with symmetric extension. Thus, the design of reversible transforms with the symmetry-preserving property is preferred whenever possible.

Table 4.11. *Lossless compression results for periodic versus symmetric extension*

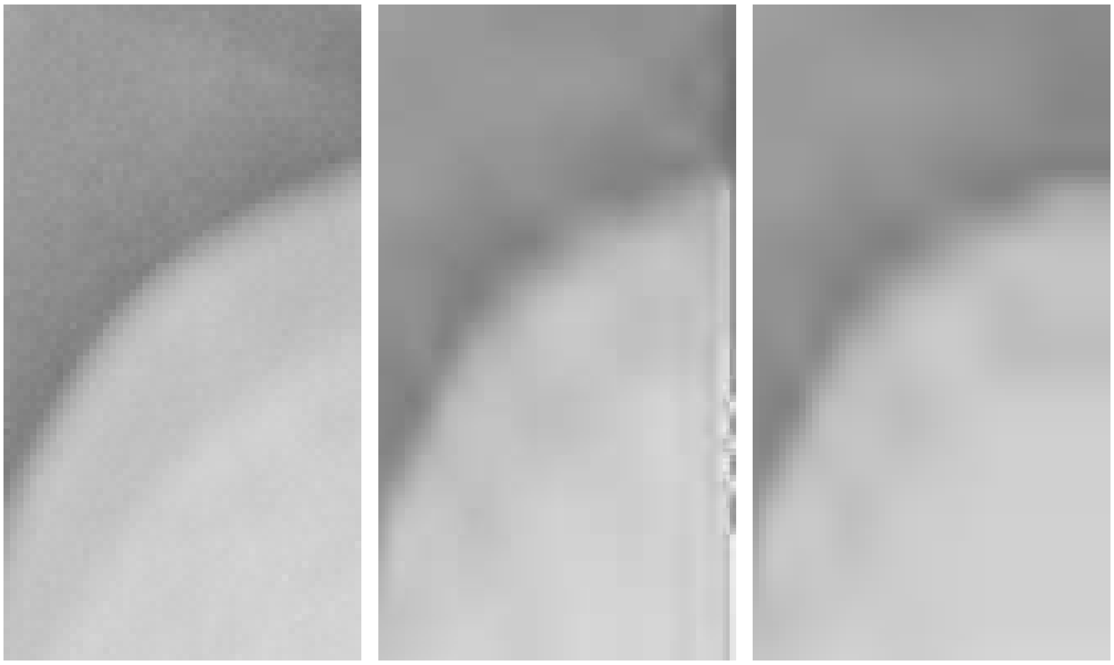
	CR			
	CDF22		BCW3	
Image	PE	SE	PE	SE
`barb`	1.680	1.686	1.726	1.733
`chart_s`	2.431	2.433	2.379	2.385
`cmpnd1`	3.715	3.714	3.041	3.041
`finger`	1.438	1.439	1.474	1.475
`lena`	1.865	1.875	1.881	1.893
`molecule`	3.978	4.030	3.868	3.930

Table 4.12. *Lossy compression results for periodic versus symmetric extension*

		PSNR (dB)			
		CDF22		BCW3	
Image	CR	PE	SE	PE	SE
`barb`	8	34.34	34.40	35.26	35.34
	16	30.15	30.21	30.81	30.91
	32	26.36	26.46	26.72	26.83
	64	23.93	24.04	24.12	24.23
	128	23.26	23.40	23.29	23.39
`chart_s`	8	41.30	41.30	41.08	41.11
	16	34.03	34.02	34.32	34.39
	32	29.41	29.41	29.85	29.90
	64	26.64	26.62	27.03	27.07
	128	23.92	23.92	24.47	24.51
`cmpnd1`	8	41.31	41.31	39.73	39.73
	16	30.76	30.75	30.62	30.61
	32	24.43	24.43	24.42	24.41
	64	20.46	20.45	20.64	20.64
	128	19.45	19.42	19.45	19.45
`finger`	8	29.61	29.63	30.98	31.00
	16	26.07	26.08	26.91	30.61
	32	22.92	22.92	23.56	24.41
	64	21.23	21.21	21.55	20.64
	128	19.52	19.49	19.56	19.45
`lena`	8	38.94	39.03	39.08	39.22
	16	35.91	36.00	36.10	36.23
	32	32.88	33.03	33.08	33.27
	64	29.95	30.21	30.06	30.35
	128	27.25	27.52	27.21	27.57
`molecule`	8	47.79	47.86	47.89	47.97
	16	44.20	44.25	43.99	44.09
	32	40.77	41.00	40.89	41.09
	64	36.84	37.64	37.39	37.70
	128	32.55	33.04	33.16	33.99

4.15 Multi-Transform Approach to Image Compression

In transform-based image compression systems, compression performance is highly dependent on the effectiveness of the decorrelating transform employed. Unfortunately, transform effectiveness is inherently signal-dependent, and consequently no single transform yields the best results for all classes of images. The results of the reversible transform performance evaluation presented earlier



(a) (b) (c)

Figure 4.15. *Lossy compression example. (a) Original image. Lossy reconstruction at compression ratio of 64 : 1 using (b) periodic extension, and (c) symmetric extension.*

well demonstrate this fact. In spite of the fact that no single transform is optimal for all classes of images, many image compression systems utilize a single fixed transform for decorrelation purposes.

Since no single transform is optimal for all classes of images, there are obvious benefits to utilizing more than one transform in an image coding system. By allowing the use of more than one transform, additional freedom exists to choose a transform that is well suited to a particular image. There are, however, two difficulties associated with such a scheme. The first problem is deciding upon a set of candidate transforms which can potentially be used during the coding process. The second problem, and also the more difficult one, is to find efficient and reliable methods for selecting an appropriate transform to use for a given image from a particular set of candidate transforms.

Before the development of lifting-based design techniques for reversible transforms, such transforms were constructed by ad hoc methods that are difficult to generalize. For this reason, reversible transforms were traditionally very difficult to design, and few good reversible transforms were known. This made a multi-transform approach unattractive as there were very few transforms from which to choose in constructing a candidate set to be used by the image coder. With lifting-based design techniques, however, reversible versions of most linear M-band subband transforms can be constructed. Thus, we are now able to exploit the large body of linear subband transforms that are known to work so well for image compression. With so many excellent reversible transforms at our disposal, is it possible to seriously consider a multi-transform approach to reversible embedded image compression. Since so many good linear subband transforms are known, it is relatively easy to produce reversible transforms that work well for particular classes of images. In fact, we can even custom design reversible transforms for particular classes of images, and then use these transforms to form the candidate set. With several performant transforms from which to choose, we can often select a better transform, thus, improving compression performance.

We now propose a simple multi-transform approach to reversible embedded image compression. With this scheme, the decorrelating transform employed by the image coder is selected on a per-image basis from a candidate set of transforms using image-specific characteristics. By using image-specific information in the selection process, a transform well suited to a particular image can be chosen. In this fashion, we can best exploit the strengths of each transform in the candidate set. Although this thesis is concerned only with reversible embedded image compression, the scope of applicability of the proposed method is potentially much larger. There is nothing to prevent these ideas from being used for purely lossless compression or even strictly lossy compression with linear versions of the transforms described.

Having decided to use a multi-transform approach, one must choose a candidate set of transforms to be employed by the codec. Initially, the ten reversible versions in Table 4.1 and the S+P transform were considered as potential members of the candidate set. With the exception of the S+P transform, all of these transforms were designed using the lifting-based method discussed earlier. The selection of the candidate set was driven by several considerations. For a transform to be accepted as a member of the candidate set, the transform should have some or all of the following desirable properties:

1. The transform should yield good lossless compression results and reasonably good lossy results both subjectively and objectively for some reasonably large class of images.
2. It should be possible to calculate the transform using only fixed-point arithmetic. Since fixed-point arithmetic is inherently simpler and faster than floating-point arithmetic, transforms that can be calculated using only fixed-point arithmetic have a clear cost/performance advantage. Also for reasons of cross-platform portability, the use of floating-point arithmetic is undesirable. For example, different platforms may have subtle differences (or even bugs) in their floating-point implementations that could cause lossless compression to become lossy.
3. The transform should be nonexpansive for images of arbitrary size. The nonexpansive property eliminates undesirable growth in the forward transform coefficient count, simplifies the process of locating subbands in the transformed image as no padding is present, and helps to facilitate in-place calculation. Moreover, numerous complications can arise when an expansive transform is employed in an image coding system. These complications often lead to increased coder complexity and can simply be avoided by using nonexpansive transforms.
4. The transform should allow in-place calculation. In-place calculation leads to improved memory efficiency and often better time efficiency as well.

The first two criteria were considered essential. The last two were considered highly desirable, but were not a requirement. Also, not only must the individual transforms chosen have desirable characteristics, but together the transforms of the candidate set must have good coverage of the types of images likely to be encountered in practice. That is, for any given image, there should be at least one transform in the candidate set that is reasonably well suited to the image.

In an earlier section, we saw that the smoothness and uniformity of an image largely determine transform effectiveness. Moreover, three different classes of images were identified along with the transform that is most effective for images from each class. To briefly restate these results, we found that

1. for images characterized by very smooth intensity variations, the BCW3 transform is usually most effective;
2. for images with moderate amounts of nonsmooth intensity variation and smoothly varying images with very large regions of uniform intensity, the CDF22 transform is typically best;
3. for images characterized by much nonsmooth intensity variation, the Haar transform is quite effective.

These results suggest that we might consider a candidate set consisting of the BCW3, CDF22, and Haar transforms. As luck would have it, the BCW3 and CDF22 transforms have all four desired properties for any transform to be used in the candidate set. The Haar transform, however, does not meet all of these criteria as it can be expansive and often yields poor subjective results for lossy compression. In spite of the disadvantages of the Haar transform, its simplicity and excellent lossless performance for highly nonsmooth images are felt to make its use justifiable. Together, the BCW3, CDF22, and Haar transforms provide good coverage of most types of images likely to be encountered in practice. That is, most images fall into one of the three categories handled by these transforms. All of the above facts taken together lead naturally to a candidate set consisting of the BCW3, CDF22, and Haar transforms.

Earlier observations made immediately suggest two characteristics of an image that could be used for transform selection:

1. The smoothness of intensity variations in an image.
2. The proportion of an image constituted by regions of uniform (i.e., constant) intensity.

In order to use these two characteristics for transform selection, we require a quantitative measure of these characteristics. To measure both smoothness and uniformity, two simple statistics based on first-order differences were used. Suppose that we are given an image where the pixels may assume values from the set $\{0, 1, \ldots, R-1\}$. The differences between adjacent pixels in the horizontal and vertical directions are sampled as depicted in Figure 4.16. For convenience, denote these differences as d_i for $i = 0, 1, \ldots, N-1$. From these differences, we define the smoothness measure

$$s = 100N_s/N \tag{4.1}$$

where N_s is the number of d_i satisfying $|d_i| \geq R/2$. This metric can assume values on the interval $[0, 100]$ with smaller values corresponding to greater smoothness. Similarly, we define the uniformity measure

$$u = 100N_u/N \tag{4.2}$$

where N_u is the number of d_i satisfying $d_i = 0$. With this metric, a larger value corresponds to greater uniformity.

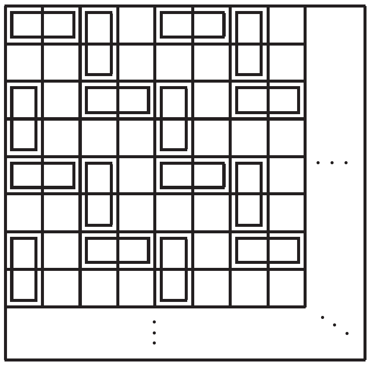


Figure 4.16. *Pairs of pixels used in difference calculations.*

Given a particular image to be compressed, we first calculate the smoothness metric s and uniformity metric u for the image as defined by (4.1) and (4.2), and then we select the decorrelating transform to use as indicated in Table 4.13. This selection process does not incur too much additional overhead as it requires only a fraction (approximately one-eighth) of the computation necessary for a single-level Haar wavelet decomposition.

Table 4.13. *Transform selection criteria*

Condition	Transform
$0 \leq u < 20$ and $0 \leq s < 0.25$ $20 \leq u < 40$ and $s < -0.01u + 0.45$ $40 \leq u < 75$ and $0 \leq s < 0.05$	BCW3
$0 \leq u < 25$ and $5 \leq s \leq 100$ $25 \leq u < 50$ and $2 \leq s \leq 100$ $50 \leq u \leq 100$ and $1 \leq s \leq 100$	Haar
otherwise	CDF22

As mentioned previously, the Haar transform is known for often introducing undesirable artifacts when used for lossy compression. If such artifacts are unacceptable for a particular application, the CDF22 transform can be used in place of the Haar transform. In almost all observed cases, if the Haar transform performs better than the BCW3 transform, the CDF22 transform will also lead to better results than the BCW3 transform.

In order to evaluate the effectiveness of the proposed method, it was employed in the reversible embedded image compression system described in Section 4.7. For test data, a suite of approximately 75 grayscale images was used, representing images with a wide assortment of sizes and of many different types (e.g., photographs, computer generated images, compound images, images with intermixed text and graphics, medical images, etc.). Here, the compression results obtained for a representative subset of these images are presented. This subset consists of twenty-five images taken from those listed in Table B.1. For more information about these images, see Appendix B.

Each of the 25 test images was coded in a lossless manner and the results obtained are shown in Table 4.14. The results for each individual transform (i.e., Haar, CDF22, CDF24, MIT97, BCW3, S+P) are given along with the results obtained with our proposed scheme. Since our method can be used both with and without the Haar transform, two sets of results are provided for this method. For each image, the result associated with the transform selected by our method is highlighted. As is evident from these numbers, the proposed method yields better results than those obtained with any of the other permissible transforms individually.

Table 4.14. *Lossless compression results for multi-transform scheme*

	Compression Ratio							
Image	Haar	CDF22	CDF24	CDF97	MIT97	BCW3	S+P	Proposed Method with/without Haar
`air1`	1.403	1.468	1.461	1.468	1.476	1.476	1.480	1.476
`air2`	1.726	1.869	1.860	1.819	1.891	1.889	1.881	1.889
`airplane`	2.317	2.409	2.401	2.264	2.385	2.389	2.389	2.409
`barb`	1.556	1.686	1.692	1.710	1.724	1.733	1.712	1.733
`bike3`	1.651	1.728	1.719	1.706	1.725	1.725	1.715	1.728
`chart_s`	2.320	2.433	2.412	2.272	2.388	2.385	2.402	2.433
`cmpnd1`	4.490	3.714	3.407	2.946	3.103	3.041	3.504	4.490/3.714
`cmpnd2`	4.272	3.713	3.395	2.949	3.117	3.057	3.431	4.272/3.713
`cr`	1.864	1.930	1.928	1.919	1.924	1.928	1.911	1.928
`ct`	2.495	2.975	2.941	3.037	3.191	3.179	3.120	3.179
`eafbcmpnd`	2.445	1.851	1.706	1.475	1.524	1.501	1.697	2.445/1.851
`express1`	3.133	2.515	2.388	2.003	2.187	2.161	2.263	3.133/2.515
`finger`	1.272	1.439	1.432	1.451	1.477	1.475	1.445	1.475
`france`	5.328	3.759	3.452	2.527	2.945	2.900	3.228	5.328/3.759
`gold`	1.588	1.675	1.670	1.655	1.675	1.676	1.662	1.676
`hotel`	1.623	1.687	1.684	1.667	1.686	1.687	1.672	1.687
`lax`	1.358	1.371	1.368	1.366	1.366	1.368	1.360	1.368
`lena`	1.740	1.875	1.870	1.875	1.891	1.893	1.881	1.893
`library`	1.460	1.401	1.382	1.337	1.352	1.347	1.359	1.460/1.401
`mandrill`	1.283	1.329	1.327	1.330	1.333	1.335	1.332	1.335
`medical_93`	2.588	2.941	2.920	2.706	3.004	2.998	2.953	2.998
`molecule`	3.903	4.030	4.004	3.610	3.927	3.930	3.960	4.030
`mri`	1.722	1.845	1.838	1.850	1.879	1.879	1.885	1.879
`us`	2.567	2.539	2.484	2.202	2.429	2.415	2.507	2.567/2.539
`xray1`	2.696	3.103	3.088	2.812	3.156	3.155	3.092	3.103
Mean	2.352	2.291	2.233	2.078	2.190	2.180	2.234	2.477/2.308

Although the results presented are for an EZW-based coder, similar results have also been obtained with the SPIHT coding scheme described in [37]. In this regard, the effectiveness of our method is somewhat independent of the coding scheme employed.

From the above results, we can see that the proposed method is quite effective despite its simplicity, yielding better compression results than are obtained by the use of a single fixed transform. By selecting the decorrelating transform on a per image basis using image-specific statistics, we are able to choose a more effective transform, thus, obtaining improved compression performance. Although some additional computational overhead is incurred by the transform selection process, this overhead is relatively small. Moreover, the selection algorithm itself is extremely easy to implement.

4.16 Summary

We began the chapter with a brief introduction to image compression. This led to the discussion of transform-based compression systems and the motivation for their use. Next, measures for lossy and lossless compression performance were presented. Reversible embedded image compression was introduced and its merits described. Next, the EZW scheme was briefly presented as an example of an embedded coding technique. This led to a brief discussion of the particular EZW-based image coder used to obtain most of the results in this thesis. Next, wavelet transforms were discussed in the context of image compression. The advantages of using wavelet transforms over other more traditional transforms such as the DCT were presented.

Several 2-band linear wavelet transforms known to be effective for image compression were presented along with two 4-band linear wavelet transforms—none of these transforms being reversible. Reversible versions of these transforms were then constructed using the lifting-based method described in the previous chapter. The reversible transforms were then employed in our EZW-based image compression system and their effectiveness evaluated. Both the cases of lossy and lossless compression were considered. The merits of full versus partial embedding and periodic versus symmetric extension were also studied. Based on the observations made previously in the chapter, a multi-transform approach to image compression was proposed.

4.17 Conclusions

This chapter presented numerous new results regarding reverisble embedded image compression. Here, we briefly summarize these contributions.

Several reversible transforms were evaluated in terms of their effectiveness for image compression. This led to many useful observations as to which transforms are most effective for various classes of images and what types of artifacts are obtained from the various transforms when employed for lossy compression. Some of these observations are as follows:

- For images characterized by very smooth intensity variations and lacking very large areas of uniform intensity, the BCW3 transform performs best overall for lossy and lossless compression.

- For images with moderate amounts of nonsmooth variation and smoothly varying images with very large regions of uniform intensity, the CDF22 transform is typically best for lossy and lossless compression.
- For images characterized by much nonsmooth variation and large regions of uniform intensity, the Haar transform is extremely effective for lossless compression. This said, however, the subjective quality of lossy compression results are often poor.

Of the two 4-band transforms studied, the A4 transform is comparable to some of the 2-band transforms, but not as good as the best of the 2-band transforms. The V4 transform fares very poorly due to the highly discontinuous synthesis basis functions associated with it.

Compression results obtained using periodic extension and symmetric extension were compared, and symmetric extension was found to be clearly superior (as is in the linear transform case). Symmetric extension has the advantages of not introducing jump discontinuities in the extended signal and also allowing the construction of nonexpansive transforms for signals of arbitrary length. Of these two advantages, the first was found to be more significant from a compression performance viewpoint, and the second was found to be more important from a coder complexity viewpoint. Since symmetric extension can only be used in conjunction with symmetry-preserving transforms, reversible transforms with this property are clearly very desirable.

The relative merits of full versus partial embedding were also studied. The fully embedded approach used the EZW scheme to completely code all of the transform coefficient bits. The partially embedded approach used the EZW scheme until the three LSBs of the transform coefficients were reached, and then the remaining bits were coded in a very simple nonembedded manner. Although some images benefit significantly from the use of a partially embedded bitstream, this was found not to be true in all cases, particularly for some noiseless computer-generated images. Furthermore, it was observed that one could use the transform coefficients in order to accurately predict which coding scheme is most effective. By measuring the fraction of transform coefficients with all zeros in the 3 LSB positions, one can easily predict whether full or partial embedding is likely to work best.

In this chapter, a multi-transform approach to image compression was proposed. With this scheme, the decorrelating transform employed by the image coder is selected from a set of candidate transforms based on image-specific characteristics. The transform selection algorithm incurs relatively little additional overhead, yet significantly improves compression results. This demonstrates that it is possible to achieve improved compression performance by judicious choice of decorrelating transform, and that such transform selection can be done without incurring objectionable amounts of additional computation.

Finally, the numerical results presented throughout this chapter demonstrate that the EZW coding scheme can be used to good effect for lossless image compression. Only a few minor modifications to the EZW coding scheme, were needed (as outlined previously) in order to accomplish this.

I was gratified to be able to answer promptly, and I did. I said I didn't know.

—Mark Twain

Chapter 5

Conclusions and Future Research

> I dread success. To have succeeded is to have finished one's business on earth, like the male spider, who is killed by the female the moment he has succeeded in his courtship. I like a state of continual becoming, with a goal in front and not behind.
>
> —George Bernard Shaw

5.1 Overview

This thesis has studied the design and implementation of reversible wavelet transforms and their application to reversible embedded image compression. In summary, the first part of the thesis focused mainly on the reversible transforms themselves, while the remaining portion considered their use in reversible embedded image compression systems. In accordance with this logical partitioning, the contributions made by this thesis can be categorized as either pertaining to reversible transforms or image compression. These contributions are summarized in the two sections that follow.

5.2 Reversible Transforms

By generalizing the lifting realization to the M-band case, we were able to extend the design method in [10] so that reversible versions of any M-band transform (for arbitrary M) can be produced. Also, for completeness, realizations based on both type 1/2 and type 3/4 polyphase forms are considered.

Using the extended design method, reversible versions of numerous transforms were constructed. Through numerical results it was demonstrated that this lifting-based design method can produce reversible transforms that well approximate their parent linear transforms. Similarly, it was shown that the type of filtering structure produced by this design method is ideally suited to implementation using only fixed-point arithmetic. This result is important because it implies that reversible transforms derived from lifting can be implemented much more efficiently using fixed-point instead of floating-point arithmetic.

The lifting-based reversible transform design method involves finding a lifting factorization of the polyphase matrix of a QMF bank. A simple, yet effective, software-based algorithm for finding good lifting factorizations in the M-band case was proposed. This technique addresses numerical ill-conditioning problems that can arise during the factorization process. Having a software algorithm for calculating lifting factorizations in the M-band case (for $M > 2$) is practically very useful since computation by hand would be extremely tedious for all but trivial, and therefore useless, sets of filters.

Some well-known reversible transforms such as the S+P transform do not quite fit into the lifting framework. Combining ideas from the S+P transform and lifting, a new and more general framework for reversible transform construction was proposed. This new framework is beneficial as it provides a more flexible approach for the design of reversible transforms and is capable of generating an even larger class of reversible transforms than those derived with lifting alone. For example, we can generate reversible versions of transforms based on QMF banks with IIR filters.

5.3 Reversible Embedded Image Compression

In this research, a reversible embedded image compression system based on the EZW coding scheme was used to obtain most results. Some modifications were made to the EZW coding scheme as originally proposed by Shapiro [41] in order to adapt it to handling M-band transforms, arbitrary-sized images, and reversible coding. Moreover, the modified coding scheme is partially embedded. At high bit rates, the coder switches to a simple residual coding method. This achieves better lossless compression ratios for many images. Further experimentation, however, suggested that even better performance could be achieved by choosing between full and partial embedding based on the values of the transform coefficients being coded. Through the numerical results presented, it is clear that the EZW scheme is highly effective for reversible coding. And, in fact, the price paid for reversibility is not too high as lossy compression results are still very close to those given in Shapiro's original paper on EZW [41].

Several reversible 2-band wavelet transforms were studied in the context of both lossless and lossy compression, namely, the Haar, TS, CDF22, CDF24, CDF97, V610, MIT97, and BCW3 transforms. Of the transforms considered, the CDF22 and BCW3 transforms were found to be most effective for lossy and lossless compression of nonsmooth and smooth images, respectively. The Haar transform is also extremely effective for the compression of highly nonsmooth images, but its subjective lossy performance is often very poor.

The observations made on the effectiveness of the various reversible transforms for different classes of images led to the proposal of a multi-transform approach to image compression. In this approach, the transform employed by the image codec is selected on a per image basis using simple image-based statistics. The selection method incurs very little additional overhead and is trivial to implement. By selecting the most appropriate transform for the image at hand, improved compression performance is achieved. Despite the simplicity of the technique, it works very well. Similar results were also obtained with a SPIHT based codec, so this multi-transform approach is at least somewhat independent of the coding scheme employed.

Two reversible 4-band wavelet transforms were also considered in the thesis, namely the V4 and A4 transforms. The results obtained with these two transforms tended to be inferior to those obtained with the best of the 2-band transforms considered. The 4-band transforms do not have very smooth synthesis scaling/wavelet functions, and this is believed to be a significant contributing factor to the poor results. The results obtained with these transforms, however, did demonstrate that the lifting-based reversible transform design method can be applied to M-band transforms to good effect. Whether M-band transforms can ultimately outperform 2-band transforms remains to be seen.

5.4 Future Research

Although this thesis has made some useful contributions towards improved reversible transforms and reversible embedded image compression systems, there are still many areas that could potentially benefit from further research. Some of these areas include the following:

- The motivation behind a multi-transform approach to image compression is simply to improve compression performance by selecting the transform best suited to a particular image. The multi-transform technique proposed earlier in this thesis applies the same transform to the entire image. Such an approach can not adequately handle images with characteristics that vary greatly in different regions (e.g., compound images). It is highly likely that improved compression performance can be achieved by segmenting an image (into homogeneous regions) and then applying (possibly) different transforms to each subregion. For such an approach to be feasible, fast and effective segmentation algorithms and transform selection schemes would need to be devised.
- Yet another possibility for improving transform effectiveness is to develop spatially varying transforms. In this case, a single transform is applied to the entire image, but the transform adapts itself to the spatially varying statistics of the image. Lifted transforms lend themselves particularly well to such a scheme.
- The transforms discussed in this thesis are most appropriate for multi-level images. Often, however, compression systems must also be able to handle bi-level images. Thus, building high-performance transforms for bi-level images would be a useful endeavor. As a starting point, one might consider some recent work by Swanson and Tewfik [47] which considers generalizations of wavelet transforms over finite fields for handling bi-level images.
- Most research efforts to date have focused on separable transforms. Although such transforms are computationally more efficient than their nonseparable counterparts, they have the disadvantage of having severe axial dependencies. By using nonseparable transforms, this problem can be reduced. Lifting may prove beneficial in this context as it may lead to improved design techniques and more efficient implementation strategies for nonseparable transforms. Moreover, it may be possible to extend the methods discussed in this thesis in order to construct reversible nonseparable transforms.
- While 2-band wavelet transforms have been exhaustively studied, M-band wavelet transforms (with $M > 2$) have not been as widely researched. There are, however, some good reasons to believe that M-band transforms may be more effective for signal compression. One desirable

feature of M-band wavelet systems is that it is possible to have orthogonality with symmetric, finitely-supported scaling/wavelet functions. This is not possible with 2-band systems except for the trivial case of the Haar and other Haar-like transforms. Futher work on developing better M-band wavelet transforms would be beneficial.

- Often a reversible transform is constructed to calculate a scaled version of its parent linear transform. It has been observed, however, that one particular scaling doesn't lead to the best compression results in all cases. Therefore, one might want to consider a scheme where the scaling is chosen based on the characteristics of a particular image.
- This thesis has dealt exclusively with the compression of grayscale images. An increasing number of applications today, however, use multi-component (e.g., color) images. Effective means for applying transform techniques to such images would be beneficial. Moreover, good embedded coding techniques for multi-component images is an area that would benefit from further research.

5.5 Closing Remarks

Reversible embedded image coding provides a convenient framework for building unified lossy/lossless image compression systems. Moreover, the embedded nature of the compressed bitstream is very attractive in many applications. This thesis has striven to make contributions in both the areas of reversible transforms and reversible embedded image compression. By exploiting the ideas presented herein, new reversible embedded image compression systems with improved compression performance can be built. Thus, this research can benefit the many applications in which images are stored and communicated, for example, image database retrieval and image archiving.

My pen is at the bottom of a page,
Which, being finished, here the story ends;
'Tis to be wished it had been sooner done,
But stories somehow lengthen when begun.

—Byron

Bibliography

[1] M. D. Adams and A. Antoniou. A comparison of new reversible transforms for image compression. In *Proc. of IEEE Pacific Rim Conference*, volume 1, pages 298–301, Victoria, BC, Canada, August 1997.

[2] M. D. Adams and A. Antoniou. Design of reversible subband transforms using lifting. In *Proc. of IEEE Pacific Rim Conference*, volume 1, pages 489–492, Victoria, BC, Canada, August 1997.

[3] M. D. Adams and A. Antoniou. A multi-transform EZW-based approach to reversible embedded image compression. Preprint, December 1997.

[4] M. D. Adams and A. Antoniou. A multi-transform approach to reversible embedded image compression. To appear in *Proc. of IEEE International Symposium on Circuits and Systems*, June 1998.

[5] O. Alkin and H. Caglar. Design of efficient m-band coders with linear-phase and perfect-reconstruction properties. *IEEE Transactions on Signal Processing*, 43(7):1579–1590, July 1995.

[6] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies. Image coding using wavelet transform. *IEEE Transactions on Image Processing*, 1(2):205–220, April 1992.

[7] C. M. Brislawn. Preservation of subband symmetry in multirate signal coding. *IEEE Transactions on Signal Processing*, 43(12):3046–3050, December 1995. Available from `ftp://ftp.c3.lanl.gov/pub/WSQ/documents/TranSP95.ps.Z`.

[8] C. M. Brislawn. Classification of nonexpansive symmetric extension transforms for multirate filter banks. Preprint, March 1996. Available from `ftp://ftp.c3.lanl.gov/pub/WSQ/documents/classify.ps.Z`.

[9] C. M. Brislawn, J. N. Bradley, R. J. Onyshczak, and T. Hopper. The FBI compression standard for digitized fingerprint images. Preprint, 1996. Available from `ftp://ftp.c3.lanl.gov/pub/WSQ/documents/spie96.ps.Z`.

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

[11] A. Cohen, I. Daubechies, and J.-C. Feauveau. Biorthogonal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, 45:485–560, 1992.

[12] R. E. Crochiere, S. A. Webber, and J. L. Flanagan. Digital coding of speech in sub-bands. *Bell System Technical Journal*, 55(8):1069–1085, October 1976.

[13] A. Croisier, D. Esteban, and C. Galand. Perfect channel splitting by use of interpolation/decimation/tree decomposition techniques. In *International Conference on Information Sciences and Systems*, pages 443–446, Patras, Greece, August 1976.

[14] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, 41(7):909–996, November 1988.

[15] I. Daubechies and W. Sweldens. Factoring wavelet transforms into lifting steps. Technical report, Bell Laboratories, Lucent Technologies, 1996. Available from `http://cm.bell-labs.com/who/wim/papers/factor.ps`.

[16] D. F. Delchamps. *State Space and Input-Output Linear Systems.* Springer-Verlag, New York, 1988.

[17] D. E. Dudgeon and R. M. Mersereau. *Multidimensional Digital Signal Processing.* Prentice-Hall, Englewood Cliffs, NJ, 1984.

[18] A. Grossman and J. Morlet. Decomposition of hardy functions into square integrable wavelets of constant shape. *SIAM Journal on Mathematical Analysis*, 15(4):723–736, July 1984.

[19] A. Haar. Zur theorie der orthogonalen funktionen-systeme. *Math. Annal.*, 69:331–371, 1910.

[20] M. Iwahashi, H. Kiya, and K. Nishikawa. Subband coding of images with circular convolution. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1356–1359, March 1992.

[21] N. Jacobson. *Basic Algebra I.* W. H. Freeman and Company, San Francisco, 1974.

[22] N. S. Jayant and P. Noll. *Digital Coding of Waveforms: Principals and Applications to Speech and Video.* Prentice-Hall, Englewood Cliffs, New Jersey, 1984.

[23] A. A. C. Kalker and I. A. Shah. Ladder structures for multidimensional linear phase perfect reconstruction filter banks and wavelets. In *Proceedings of SPIE: Visual Communications and Image Processing*, volume 1818, pages 12–20, Boston, MA, November 1992.

[24] T. A. C. M. Kalker and I. A. Shah. On ladder structures and linear phase conditions for multidimensional bi-orthogonal filter banks. Preprint, June 1993.

[25] G. Karlsson and M. Vetterli. Extension of finite length signals for sub-band coding. *Signal Processing*, 17:161–168, June 1989.

[26] G. G. Langdon. An introduction to arithmetic coding. *IBM Journal of Research and Development*, 28(2):135–149, March 1984.

[27] A. S. Lewis and G. Knowles. Image compression using the 2-D wavelet transform. *IEEE Transactions on Image Processing*, 1(2):244–250, April 1992.

[28] P. Lux. A novel set of closed orthogonal functions for picture coding. *Archiv fur Elektronik und Uebertragungstechnik*, 31(7):267–274, July 1977.

[29] S. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, July 1989.

[30] S. A. Martucci and R. M. Mersereau. The symmetric convolution approach to the nonexpansive implementation of FIR filter banks for images. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, number 5, pages 65–68, Minneapolis, MN, April 1993.

[31] F. Mintzer. Filters for distortion-free two-band multirate filter banks. *IEEE Transactions on Acoustics, Speech, and Signal Proceesing*, 33(3):626–630, June 1985.

[32] A. Moffat, R. Neal, and I. H. Witten. Arithmetic coding revisited. In *Proc. of IEEE Data Compression Conference*, pages 202–211, Snowbird, UT, 1995.

[33] K. Nishikawa, H. Kiya, and M. Sagawa. Property of circular convolution for subband image coding. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, number 4, pages 281–284, March 1992.

[34] M. Rabbani and P. W. Jones. *Digital Image Compression Techniques.* SPIE Optical Engineering Press, Bellingham, WA, 1991.

[35] F. Rubin. Arithmetic stream coding using fixed precision registers. *IEEE Transactions on Information Theory*, 25(6):672–675, November 1979.

[36] A. Said and W. A. Pearlman. An image multiresolution representation for lossless and lossy compression. *IEEE Transactions on Image Processing*, 5(9):1303–1310, September 1996. Available from `ftp://ipl.rpi.edu/pub/EW_Code/progloss.ps.gz`.

[37] A. Said and W. A. Pearlman. A new fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3):243–250, June 1996. Available from `ftp://ipl.rpi.edu/pub/EW_Code/SPIHT.ps.gz`.

[38] I. A. Shah, O. Akiwumi-Assani, and B. Johnson. A chip set for lossless image compression. *IEEE Journal of Solid-State Circuits*, 26(3):237–244, March 1991.

[39] J. M. Shapiro. An embedded wavelet hierarchical image coder. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, number 4, pages 657–660, San Francisco, CA, March 1992.

[40] J. M. Shapiro. Application of the embedded wavelet hierarchical image coder to very low bit rate image coding. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, number 5, pages 558–561, Minneapolis, MN, April 1993.

[41] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, 41(12):3445–3462, December 1993.

[42] J. M. Shapiro. A fast technique for identifying zerotrees in the EZW algorithm. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages 1455–1458, Atlanta, GA, May 1996.

[43] M.J.T. Smith and T.P. Barnwell. Exact reconstruction techniques for tree-structured subband coders. *IEEE Transactions on Acoustics, Speech, and Signal Proceesing*, 34(3):434–441, June 1986.

[44] A. K. Soman, P. P. Vaidyanathan, and T. Q. Nguyen. Linear phase paraunitary filter banks: Theory, factorizations, and designs. *IEEE Transactions on Signal Processing*, 41(12):3480–3496, December 1993.

[45] G. Strang. Creating and comparing wavelets. Preprint, October 1996. Available from `http://www-math.mit.edu/~gs/papers/dundee.ps.gz`.

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

[47] M. D. Swanson and A. H. Tewfik. A binary wavelet decomposition of binary images. Preprint, 1997.

[48] W. Sweldens. The lifting scheme: A construction of second generation wavelets. Preprint, July 1995. Available from `ftp://ftp.math.sc.edu/pub/imi_95/imi95_6.ps`.

[49] W. Sweldens. The lifting scheme: A new philosophy in biorthogonal wavelet constructions. In *Proceedings of SPIE Wavelet Applications in Signal and Image Processing III*, pages 68–79, 1995. Available from `ftp://ftp.math.sc.edu/pub/wavelet/papers/varia/spie95.ps.gz`.

[50] W. Sweldens. Wavelets and the lifting scheme: A 5 minute tour. Preprint, August 1995. Available from `ftp://ftp.math.sc.edu/pub/wavelet/papers/varia/iciam95.ps.gz`.

[51] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and Computational Harmonic Analysis*, 3(2):186–200, 1996. Available from `ftp://ftp.math.sc.edu/pub/imi_94/imi94_7.ps`.

[52] J. Tian. *The Mathematical Theory and Applications of Biorthogonal Coifman Wavelet Systems*. PhD thesis, Department of Mathematics, Rice University, February 1996.

[53] P. P. Vaidyanathan. Theory and design of m-channel maximally decimated quadrature mirror filters with arbitrary m, having the perfect-reconstruction property. *IEEE Transactions on Acoustics, Speech, and Signal Proceesing*, 35(4):476–492, April 1987.

[54] P. P. Vaidyanathan. *Multirate Systems and Filter Banks*. Prentice-Hall, Englewood Cliffs, New Jersey, 1993.

[55] M. Vetterli. Filter banks allowing perfect reconstruction. *Signal Processing*, 10(3):219–244, April 1986.

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

[57] J. D. Villasenor, B. Belzer, and J. Liao. Wavelet filter evaluation for image compression. *IEEE Transactions on Image Processing*, 4(8):1053–1060, August 1995.

[58] D. Wei, J. Tian, R. O. Wells, and C. S. Burrus. A new class of biorthogonal wavelet systems for image transform coding. Technical report, Computational Mathematics Laboratory, Rice University, October 1995. Technical Report CML TR 95-14.

[59] I. H. Witten, R. M. Neal, and J. G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987.

[60] A. Zandi, J. D. Allen, E. L. Schwartz, and M. Boliek. CREW: Compression with reversible wavelets. In *Proc. of IEEE Data Compression Conference*, pages 212–221, Snowbird, UT, 1995. Available from `ftp://ftp.crc.ricoh.com/pub/CREW/DCC95.ps.gz`.

[61] H. Zou and A. H. Tewfik. Discrete orthogonal m-band wavelet decompositions. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, number 4 in 4, pages 605–608, San Francisco, CA, 1992.

The great consulting room of a wise man is a library.
—George Dawson

Appendix A

Proofs

It is by the fortune of God that, in this country, we have three benefits: freedom of speech, freedom of thought, and the wisdom never to use either.

—Mark Twain

A.1 Proof of Theorem 2.1

First, let us consider the effects of transforming the analysis filters. We begin by making the following observations:

1. For both type 1/2 and type 3/4 systems, multiplying the analysis filter transfer functions by β_0 multiplies the analysis polyphase matrix by β_0.
2. For type 1/2 systems, multiplying the analysis filter transfer functions by z^{-1} circularly shifts the analysis polyphase matrix one column to the right and multiplies the wrapped column by z^{-1}. This is equivalent to postmultiplying the analysis polyphase matrix by

$$\begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}$$

   Multiplying the analysis filter transfer functions by z instead of z^{-1} simply postmultiplies the analysis polyphase matrix by the inverse of the above matrix.
3. For type 3/4 systems, multiplying the analysis filter transfer functions by z^{-1} circularly shifts the analysis polyphase matrix one column to the left and multiplies the wrapped column by z^{-1}. In other words, the analysis polyphase matrix is postmultiplied by

$$\begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}$$

   Multiplying the analysis filter transfer functions by z instead of z^{-1} simply postmultiplies the analysis polyphase matrix by the inverse of the above matrix.

Using (1) and by repeatedly using (2) and (3) L_0 times, we find that multiplying the analysis filter transfer functions by $\beta_0 z^{-L_0}$ postmultiplies the analysis polyphase matrix by

$$\begin{cases} \beta_0 \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_0} & \text{type 1/2} \\ \beta_0 \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_0} & \text{type 3/4} \end{cases}$$

Thus, the old and new analysis polyphase matrices are related as specified in the theorem.

Now, let us consider the effects of transforming the synthesis filters. We start by making the following observations:

1. For both type 1/2 and type 3/4 systems, multiplying the synthesis filter transfer functions by β_1 multiplies the synthesis polyphase matrix by β_1.
2. For type 1/2 systems, multiplying the synthesis filter transfer functions by z^{-1} circularly shifts the synthesis polyphase matrix upwards by one row and multiplies the wrapped row by z^{-1}. This is equivalent to premultiplying the synthesis polyphase matrix by

   $$\begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}$$

   Multiplying the synthesis filter transfer functions by z instead of z^{-1} simply premultiplies the synthesis polyphase matrix by the inverse of the above matrix.
3. For type 3/4 systems, multiplying the synthesis filter transfer functions by z^{-1} circularly shifts the synthesis polyphase matrix one row downwards and multiplies the wrapped column by z^{-1}. In other words, the synthesis polyphase matrix is premultiplied by

   $$\begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}$$

   Multiplying the synthesis filter transfer functions by z instead of z^{-1} simply premultiplies the synthesis polyphase matrix by the inverse of the above matrix.

Using (1) and by repeatedly using (2) and (3) L_1 times, we have that multiplying the synthesis filter transfer functions by $\beta_1 z^{-L_1}$ premultiplies the synthesis polyphase matrix by

$$\begin{cases} \beta_1 \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_1} & \text{type 1/2} \\ \beta_1 \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_1} & \text{type 3/4} \end{cases}$$

Thus, the old and new synthesis polyphase matrices are related as specified in the theorem. This completes the proof. ∎

A.2 Proof of Theorem 2.2

By definition, a QMF bank has the PR property if and only if it is alias-free with a distortion function that is a pure delay. A QMF bank is alias-free, however, if and only if $\boldsymbol{P}(z)$ has the form

$$\boldsymbol{P}(z) = \begin{cases} \displaystyle\sum_{k=0}^{M-1} P_k(z) \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^k & \text{type 1/2} \\ \displaystyle\sum_{k=0}^{M-1} P_k(z) \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^k & \text{type 3/4} \end{cases} \tag{A.1}$$

That is, $\boldsymbol{P}(z)$ is a right pseudocirculant matrix for a type 1/2 system and a left pseudocirculant matrix for a type 3/4 system. Furthermore, such an alias-free system has the distortion function

$$T(z) = \begin{cases} z^{-(M-1)} \displaystyle\sum_{k=0}^{M-1} z^{-k} P_k(z^M) & \text{type 1/2} \\ \displaystyle\sum_{k=0}^{M-1} z^{k} P_k(z^M) & \text{type 3/4} \end{cases} \tag{A.2}$$

If we now assume that the system is PR, we also have that $T(z)$ is of the form

$$T(z) = z^{-\kappa}$$

where κ is an integer. This implies that in (A.2) exactly one of the $P_k(z)$, say $P_\eta(z)$, is nonzero and has the form

$$P_\eta(z) = z^{-L}$$

where L is an integer. Using this fact, (A.1) simplifies to

$$\boldsymbol{P}(z) = \begin{cases} z^{-L}\boldsymbol{I}_M \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^\eta & \text{type 1/2} \\ z^{-L}\boldsymbol{I}_M \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^\eta & \text{type 3/4} \end{cases} \tag{A.3}$$

Now we observe that

$$z^{-L}\boldsymbol{I}_M = \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{ML} = \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{ML}$$

So from (A.3) we have

$$\boldsymbol{P}(z) = \begin{cases} \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{ML+\eta} & \text{type } 1/2 \\ \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{ML+\eta} & \text{type } 3/4 \end{cases}$$

$$= \begin{cases} \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{K} & \text{type } 1/2 \\ \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{K} & \text{type } 3/4 \end{cases}$$

where $K = ML + \eta$. Thus, if the QMF bank has the PR property, $\boldsymbol{P}(z)$ must have the form stated in the theorem. ■

A.3 Proof of Corollary 2.3

From Theorem 2.2, we know that $\boldsymbol{P}(z)$ has the form

$$\boldsymbol{P}(z) = \begin{cases} \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{K} & \text{type } 1/2 \\ \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{K} & \text{type } 3/4 \end{cases}$$

where K is an integer. Taking the determinant of $\boldsymbol{P}(z)$ simply yields

$$\det \boldsymbol{P}(z) = (-1)^{K(M-1)} z^{-K} \tag{A.4}$$

Thus, $\det \boldsymbol{P}(z)$ must be of the form stated in the theorem.

Using both (A.4) and the fact that $\boldsymbol{P}(z) = \boldsymbol{R}(z)\boldsymbol{E}(z)$, we have

$$\det \boldsymbol{R}(z) \det \boldsymbol{E}(z) = (-1)^{K(M-1)} z^{-K} \tag{A.5}$$

Assume now that the analysis and synthesis filters are of the FIR type. This implies that $\det \boldsymbol{E}(z)$ and $\det \boldsymbol{R}(z)$ are Laurent polynomials. The only way that the product in (A.5) can be a monomial is if both $\det \boldsymbol{E}(z)$ and $\det \boldsymbol{R}(z)$ are themselves monomials. Thus, the last part of the theorem is proven. ■

A.4 Proof of Lemma 3.1

In the case of both identities, there are two possibilities to consider:

1. The integers a and b have the same parity. That is, either both integers are odd or both are even.
2. The integers a and b have opposite parity. That is, one integer is odd and the other is even.

In the first case, $a \pm b$ is an even integer. Thus, we can write

$$\left\lfloor \tfrac{1}{2}(a+b) \right\rfloor + \left\lfloor \tfrac{1}{2}(a-b+1) \right\rfloor = \tfrac{1}{2}(a+b) + \tfrac{1}{2}(a-b) = a$$

and

$$\left\lfloor \tfrac{1}{2}(b+a) \right\rfloor - \left\lfloor \tfrac{1}{2}(b-a) \right\rfloor = \tfrac{1}{2}(b+a) - \tfrac{1}{2}(b-a) = a$$

In the second case, $a \pm b$ is an odd integer. Thus, we can write

$$\left\lfloor \tfrac{1}{2}(a+b) \right\rfloor + \left\lfloor \tfrac{1}{2}(a-b+1) \right\rfloor = \tfrac{1}{2}(a+b-1) + \tfrac{1}{2}(a-b+1) = a$$

and

$$\left\lfloor \tfrac{1}{2}(b+a) \right\rfloor - \left\lfloor \tfrac{1}{2}(b-a) \right\rfloor = \tfrac{1}{2}(b+a-1) - \tfrac{1}{2}(b-a-1) = a$$

Combining the results from the first and second cases, we have that the two identities hold for any integers a and b. ■

A.5 Proof of Theorem 3.2

Suppose we have an $M \times M$ Laurent polynomial matrix $\boldsymbol{U}(z)$ with $\det \boldsymbol{U}(z) = \Delta(z)$ where $\Delta(z)$ is not identically zero. We can arbitrarily choose any row of $\boldsymbol{U}(z)$ and divide it by $\Delta(z)$ to obtain a new matrix $\boldsymbol{V}(z)$ with $\det \boldsymbol{V}(z) = 1$. We can, however, express $\boldsymbol{U}(z)$ in terms of $\boldsymbol{V}(z)$ as

$$\boldsymbol{U}(z) = \boldsymbol{S}_0(z)\boldsymbol{V}(z) \tag{A.6}$$

where $\boldsymbol{S}_0(z)$ is a Type S elementary matrix of the form $\mathcal{S}(\cdot; \Delta(z))$. Now, any Laurent polynomial matrix $\boldsymbol{V}(z)$ with $\det \boldsymbol{V}(z) = 1$ can be decomposed using Euclid's algorithm (see [21], [16]) into the factors

$$\boldsymbol{V}(z) = \boldsymbol{A}_0(z)\boldsymbol{A}_1(z)\cdots\boldsymbol{A}_{P-1}(z) \tag{A.7}$$

where the $\boldsymbol{A}_i(z)$ are Type A elementary matrices. Combining equations (A.6) and (A.7), we have that $\boldsymbol{U}(z)$ admits a decomposition of the form

$$\boldsymbol{U}(z) = \boldsymbol{S}_0(z)\boldsymbol{A}_0(z)\boldsymbol{A}_1(z)\cdots\boldsymbol{A}_{P-1}(z)$$

Therefore, $\boldsymbol{U}(z)$ must also permit a factorization of the form

$$\boldsymbol{U}(z) = \boldsymbol{S}_0(z)\boldsymbol{S}_1(z)\cdots\boldsymbol{S}_{Q-1}(z)\boldsymbol{A}_0(z)\boldsymbol{A}_1(z)\cdots\boldsymbol{A}_{P-1}(z)$$

From the previous equation, we have

$$\begin{aligned}\det \boldsymbol{U}(z) &= \left(\prod_{i=0}^{Q-1} \det \boldsymbol{S}_i(z)\right)\left(\prod_{i=0}^{P-1} \det \boldsymbol{A}_i(z)\right)\\ &= \prod_{i=0}^{Q-1} \det \boldsymbol{S}_i(z)\end{aligned}$$

Therefore, if $\prod \det \boldsymbol{S}_i(z) \neq \Delta(z)$, then the factorization cannot be completed. ■

A.6 Proof of Theorem 3.3

From Theorem 2.1, we have

$$\boldsymbol{E}'(z) = \begin{cases} \beta_0 \boldsymbol{E}(z) \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_0} & \text{type } 1/2 \\ \beta_0 \boldsymbol{E}(z) \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_0} & \text{type } 3/4 \end{cases} \tag{A.8}$$

and

$$\boldsymbol{R}'(z) = \begin{cases} \beta_1 \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_1} \boldsymbol{R}(z) & \text{type } 1/2 \\ \beta_1 \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_1} \boldsymbol{R}(z) & \text{type } 3/4 \end{cases} \tag{A.9}$$

From equation (A.8), it follows that

$$\begin{aligned}\det \boldsymbol{E}'(z) &= \beta_0^M (-1)^{L_0(M-1)} z^{-L_0} \det \boldsymbol{E}(z)\\ &= \alpha_0 \beta_0^M (-1)^{L_0(M-1)} z^{-(L_0+K_0)}\end{aligned}$$

Thus, we have the first identity stated in the theorem. Now, solving for $\boldsymbol{E}^{-1}(z)$ in (A.8) yields

$$\boldsymbol{E}^{-1}(z) = \begin{cases} \beta_0 \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_0} [\boldsymbol{E}'(z)]^{-1} & \text{type } 1/2 \\ \beta_0 \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_0} [\boldsymbol{E}'(z)]^{-1} & \text{type } 3/4 \end{cases} \tag{A.10}$$

Since the system is PR, we have from Theorem 2.2 that

$$\boldsymbol{R}(z) = \begin{cases} \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{K} \boldsymbol{E}^{-1}(z) & \text{type } 1/2 \\ \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{K} \boldsymbol{E}^{-1}(z) & \text{type } 3/4 \end{cases} \tag{A.11}$$

Substituting (A.11) in (A.9) and then using (A.10), we obtain

$$
\begin{aligned}
\boldsymbol{R}'(z) &= \begin{cases} \beta_1 \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_1+K_1} \boldsymbol{E}^{-1}(z) & \text{type } 1/2 \\ \beta_1 \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_1+K_1} \boldsymbol{E}^{-1}(z) & \text{type } 3/4 \end{cases} \\
&= \begin{cases} \beta_0\beta_1 \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_1+K_1} \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_0} [\boldsymbol{E}'(z)]^{-1} & \text{type } 1/2 \\ \beta_0\beta_1 \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_1+K_1} \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_0} [\boldsymbol{E}'(z)]^{-1} & \text{type } 3/4 \end{cases} \\
&= \begin{cases} \beta_0\beta_1 \begin{bmatrix} \mathbf{0} & \boldsymbol{I}_{M-1} \\ z^{-1} & \mathbf{0} \end{bmatrix}^{L_0+L_1+K_1} [\boldsymbol{E}'(z)]^{-1} & \text{type } 1/2 \\ \beta_0\beta_1 \begin{bmatrix} \mathbf{0} & z^{-1} \\ \boldsymbol{I}_{M-1} & \mathbf{0} \end{bmatrix}^{L_0+L_1+K_1} [\boldsymbol{E}'(z)]^{-1} & \text{type } 3/4 \end{cases}
\end{aligned}
$$

Thus, the second identity stated in the theorem is obtained. This completes the proof. ■

Writing is easy. All you have to do is cross out the wrong words.
—Mark Twain

Appendix B

Test Images

> After [Benjamin] Franklin came a herd of Electrical Pioneers whose names have become part of our electrical terminology: Myron Volt, Mary Louise Amp, James Watt, Bob Transformer, etc. These pioneers conducted many important electrical experiments. For example, in 1780 Luigi Galvani discovered (this is the truth) that when he attached two different kinds of metal to the leg of a frog, an electrical current developed and the frog's leg kicked, even though it was no longer attached to the frog, which was dead anyway. Galvani's discovery led to enormous advances in the field of amphibian medicine. Today, skilled veterinary surgeons can take a frog that has been seriously injured or killed, implant pieces of metal in its muscles, and watch it hop back into the pond just like a normal frog, except for the fact that it sinks like a stone.
>
> —Dave Barry, "What is Electricity?"

B.1 Overview

The various test images used in this thesis are listed in Table B.1. Many of these images are taken from the the ISO JPEG test set and USC image database as indicated in the table. The nonstandard test images used and a few of the more important standard test images are shown in Figures B.1 to B.13.

The images were originally obtained in a variety of formats (e.g., GIF, Sun Rasterfile, raw). Those which were not already in Sun Rasterfile format were converted to this format using a collection of software tools (i.e., `xv`, `pbmplus`, and software specially written by the author). The images were required to be in Sun Rasterfile format as this is the only format currently supported by the image codec software. As some of the images were originally color, conversion to grayscale was also necessary in some cases.

Table B.1. *Test images*

Image	Size	Bits/Pixel	Description
`air1`†	1024×1024	8	aerial photograph
`air2`†	720×1024	8	aerial photograph
`airplane`	768×512	8	airplane
`barb`‡	512×512	8	woman
`bike3`†	781×919	8	motorcycle
`chart_s`†	1688×2347	8	scanned chart
`cmpnd1`†	512×768	8	computer generated compound
`cmpnd2`†	1024×1400	8	computer generated compound
`cr`†	1744×2048	10	computer radiology
`ct`†	512×512	12	computer tomography
`eafbcmpnd`	496×495	8	mixed text and graphics
`express1`	559×505	8	screen capture
`finger`†	512×512	8	fingerprint
`france`	672×496	8	transparency
`gold`†	720×576	8	houses and countryside
`hotel`†	720×576	8	hotel
`lax`‡	512×512	8	aerial view of airport
`lena`‡	512×512	8	woman
`library`	464×352	8	compound
`mandrill`‡	512×512	8	face of mandrill
`medical_93`	1228×920	8	screen capture
`molecule`	909×823	8	computer generated 3D model
`mri`†	256×256	11	magnetic resonance
`us`†	512×448	8	ultrasound
`xray1`	423×600	8	medical x-ray of hand

† ISO test image

‡ USC test image

Figure B.1. `airplane` *(768×512, 8 bpp)*

Figure B.2. `barb` *(512×512, 8 bpp)*

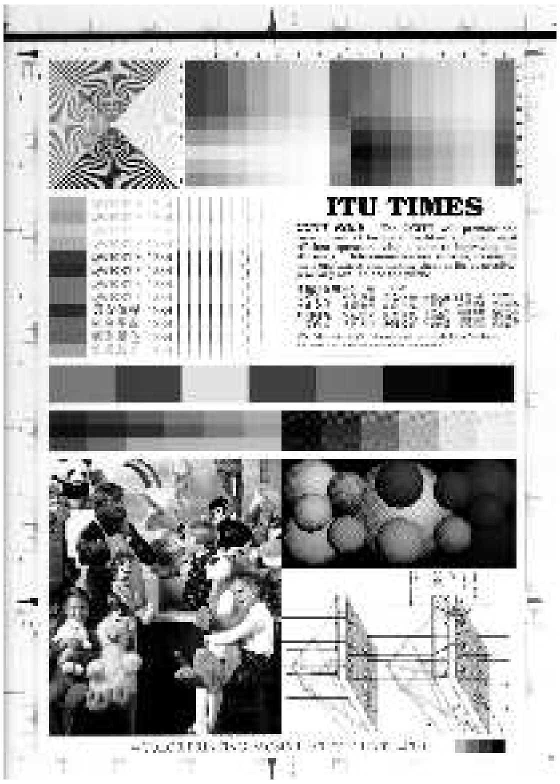


Figure B.3. `chart_s` *(1688×2347, 8 bpp)*

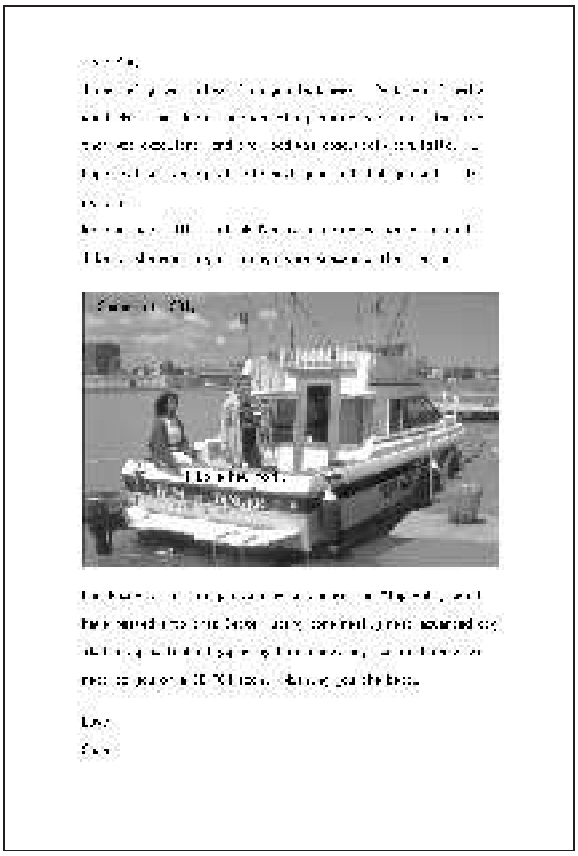


Figure B.4. `cmpnd1` *(512×768, 8 bpp)*

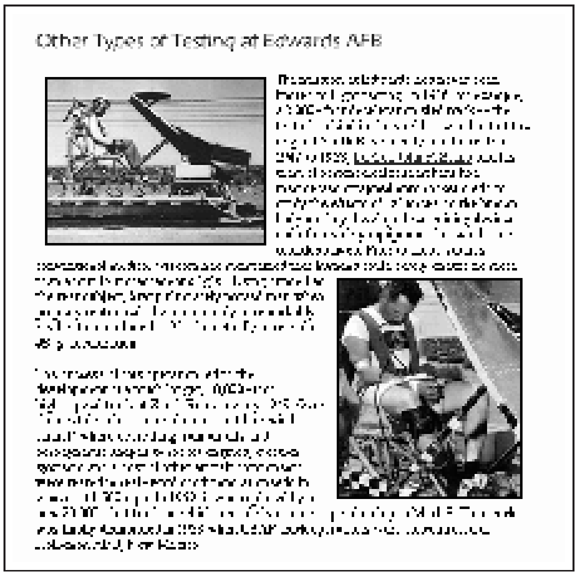


Figure B.5. `eafbcmpnd` *(496×495, 8 bpp)*

Figure B.6. `express1` *(559×505, 8 bpp)*

Figure B.7. `finger` *(512×512, 8 bpp)*

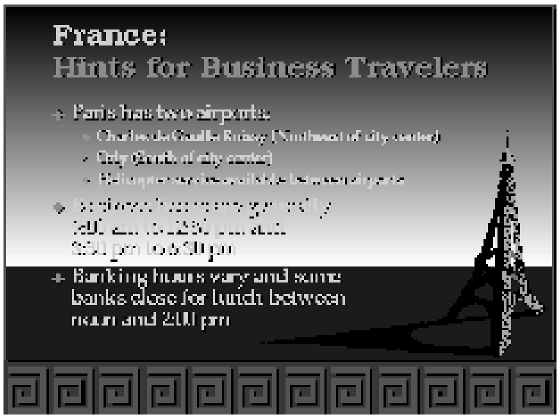


Figure B.8. `france` *(672×496, 8 bpp)*

Figure B.9. `lena` *(512×512, 8 bpp)*

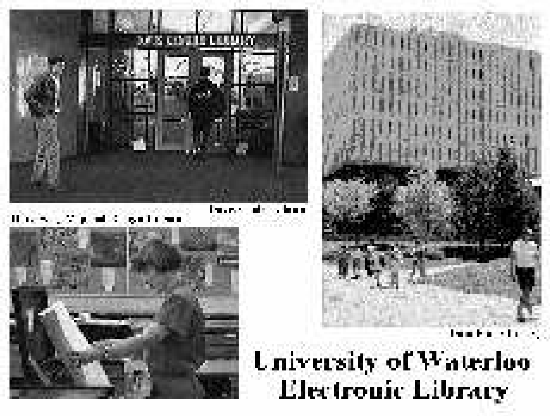


Figure B.10. `library` *(464×352, 8 bpp)*

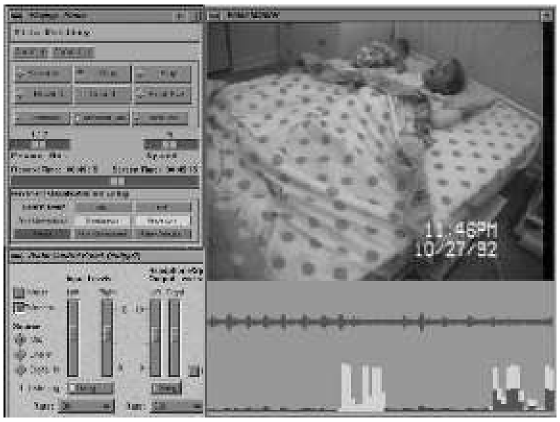


Figure B.11. `medical_93` *(1228×920, 8 bpp)*

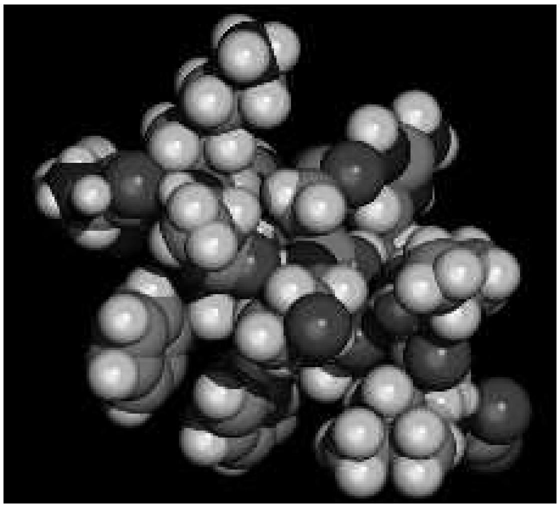


Figure B.12. `molecule` *(909×823, 8 bpp)*

Figure B.13. `xray1` *(423×600, 8 bpp)*

Appendix C

Image Compression Software

> I cannot overemphasize the importance of good grammar... What a crock. I could easily overemphasize the importance of good grammar. For example, I could say: "Bad grammar is the leading cause of slow, painful death in North America," or "Without good grammar, the United States would have lost World War II."
>
> —Dave Barry, "An Utterly Absurd Look at Grammar"

C.1 Overview

The image codec software consists of the following program:

1. The `rastoim` program which performs image compression.
2. The `imtoras` program which performs image decompression.
3. The `ratedist` program which performs rate-distortion analysis.

The codec is based on the EZW coding scheme used in conjunction with reversible wavelet transforms. The software is capable of handling grayscale images of arbitrary size (e.g., non-square and/or non-dyadic dimensions) up to 65535×65535 with as many as 16 bits/pixel. (The compression of color images is not currently supported.) Sun Rasterfile format is used for all image data external to the codec. That is, the image compression program `rastoim` expects its input to be in Sun Rasterfile format, and the image decompression program `imtoras` produces its output in the same format. The compressed data stream generated by the coder is in a format which is referred to as IM format.

The image compression program `rastoim` reads image data assumed to be in Sun Rasterfile format from standard input and writes the compressed image data in IM format to standard output. Command line options allow a specific reversible transform to be used and the number of scales in the wavelet decomposition to be specified. The entire coded bitstream required for lossless reconstruction is always produced. If a lossy reconstruction is desired, the output may be truncated at some appropriate point.

The image decompression program `imtoras` reads compressed image data in IM format from standard input and writes the uncompressed image data in Sun Rasterfile format to standard output. By default, the entire compressed bitstream is used during decoding. This default behavior can be overridden with a command line option that specifies the maximum number of bytes to be read by the decoder. This feature is useful when lossy reconstruction is desired.

C.2 The `rastoim` Command

Name

`rastoim` - image coder for reversible embedded image codec

Synopsis

`rastoim` [-v] [-c *coding_scheme*] [-t transform] [-n *max_scales*]

Description

The `rastoim` command reads an image in Sun Rasterfile from standard input and writes the compressed image to standard output in IM format. The input image must be grayscale. The complete bitstream required for lossless decoding is always generated. If only lossy compression is desired, other software can be used to truncate the coded bitstream.

Options

- -c *coding_scheme*. Select the transform coefficient quantization and coding scheme to be used by the image codec. The parameter *coding_scheme* must be an integer. The value 0 selects SPIHT and 1 selects EZW.
- -n *max_scales*. Set the maximum allowable number of scales in the wavelet decomposition to *max_scales*.
- -t *transform*. Select the decorrelating transform to be used by the image codec. The value 0 selects the built-in S+P transform. Other values select user-defined are transforms.
- -v. Enable verbose mode.

Examples

Assume that we have an image in Sun Rasterfile format stored in the file `lena.ras`. To compress the image (losslessly) using transform 0 and the SPIHT coding scheme, and write the result to the file `lena.im`, type:

```
rastoim -c 0 -t 0 -n 6 < lena.ras > lena.im
```

C.3 The `imtoras` Command

Name

`imtoras` - image decoder for reversible embedded image codec

Synopsis

`imtoras` [-v] [-m *max_bytes*]

Description

The `imtoras` command reads an image in IM format from standard input and writes the decompressed image to standard output in Sun Rasterfile format.

Options

- -m *max_bytes*. Only use the first *max_bytes* of coded bitstream during decoding.
- -v. Enable verbose mode.

Examples

Assume that we already have a compressed image in IM format stored in the file `lena.im`. To decompress the entire encoded bitstream and write the result to another file called `lena_new.ras`, type:

```
imtoras < lena.im > lena_new.ras
```

C.4 The `ratedist` Command

Name

`ratedist` - calculate rate-distortion characteristic

Synopsis

`ratedist` -o *original_image* -c *compressed_image* -Z *compression_ratio* [-b basename]

Description

The `ratedist` program reads the original image in Sun Rasterfile format and the compressed version of the image in IM format and calculates the requested rate-distortion values.

Options

- -o *original_image*. Specify the name of the file containing the original image in Sun Rasterfile format.
- -c *compressed_image*. Specify the name of the file containing the compressed image in IM format.
- -Z *compression_ratio*. Set the compression ratio for which to calculate PSNR/MSE.

- -b *basename*. Specify the basename to be used to save lossy reconstructions of the original image (in Sun Rasterfile format).

Examples

Assume that we have the original image stored in Sun Rasterfile format in `lena.ras`, and the compressed image in IM format in `lena.im`. To calculate the rate-distortion performance for the lossy compression of the image, type:

```
ratedist -o lena.ras -c lena.im
```

This is the end... my only friend, the end.
–Jim Morrison

VITA

Surname:	Adams	***Given Names:***	Michael David
Place of Birth:	Kitchener, Canada	***Date of Birth:***	Feb. 6, 1969

Educational Institutions Attended

University of Victoria	1995 to 1997
University of Waterloo	1988 to 1993

Degrees Awarded

B.A.Sc.	University of Waterloo	1993

Honors and Awards

Research Assistantship	1995 to 1997
Graduate Teaching Fellowship	Jun. 1996
Graduate Teaching Fellowship	Jan. 1996

Conference Publications

1. M. D. Adams and A. Antoniou. "A multi-transform approach to reversible embedded image compression," to appear in *Proc. of IEEE International Symposium on Circuits and Systems*, Monterey, CA, USA, June 1998.

2. M. D. Adams and A. Antoniou. "Design of reversible subband transforms using lifting," in *Proc. of IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, Victoria, B.C., Canada, Aug. 1997, vol. 1, pp. 489–492.

3. M. D. Adams and A. Antoniou. "A comparison of new reversible transforms for image compression," in *Proc. of IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, Victoria, B.C., Canada, Aug. 1997, vol. 1, pp. 298–301.

PARTIAL COPYRIGHT LICENSE

Title of Thesis: Reversible Wavelet Transforms and Their Application to Embedded Image Compression.

Author: ______________________________

MICHAEL DAVID ADAMS
January 20, 1998