LOSSLESS AND LOW-COST INTEGER-BASED LIFTING WAVELET TRANSFORM

A Thesis Submitted to the
College of Graduate and Postdoctoral Studies
in Partial Fulfillment of the Requirements
for the degree of Doctor of Philosophy
in the Department of Electrical and Computer Engineering
University of Saskatchewan
Saskatoon, Saskatchewan, Canada

By
Md. Mehedi Hasan

Permission to Use

In presenting this thesis in partial fulfillment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Electrical and Computer Engineering
University of Saskatchewan,
57 Campus Drive,
Saskatoon, Saskatchewan S7N 5A9, Canada

OR

Dean
College of Graduate and Postdoctoral Studies
University of Saskatchewan
116 Thorvaldson Building, 110 Science Place
Saskatoon, Saskatchewan S7N 5C9, Canada

Abstract

Discrete wavelet transform (DWT) is a powerful tool for analyzing real-time signals, including aperiodic, irregular, noisy, and transient data, because of its capability to explore signals in both the frequency- and time-domain in different resolutions. For this reason, they are used extensively in a wide number of applications in image and signal processing. Despite the wide usage, the implementation of the wavelet transform is usually lossy or computationally complex, and it requires expensive hardware. However, in many applications, such as medical diagnosis, reversible data-hiding, and critical satellite data, lossless implementation of the wavelet transform is desirable. It is also important to have more hardware-friendly implementations due to its recent inclusion in signal processing modules in system-on-chips (SoCs).

To address the need, this research work provides a generalized implementation of a wavelet transform using an integer-based lifting method to produce lossless and low-cost architecture while maintaining the performance close to the original wavelets. In order to achieve a general implementation method for all orthogonal and biorthogonal wavelets, the Daubechies wavelet family has been utilized at first since it is one of the most widely used wavelets and based on a systematic method of construction of compact support orthogonal wavelets. Though the first two phases of this work are for Daubechies wavelets, they can be generalized in order to apply to other wavelets as well. Subsequently, some techniques used in the primary works have been adopted and the critical issues for achieving general lossless implementation have solved to propose a general lossless method.

The research work presented here can be divided into several phases. In the first phase, low-cost architectures of the Daubechies-4 (D4) and Daubechies-6 (D6) wavelets have been derived by applying the integer-polynomial mapping. A lifting architecture has been used which reduces the cost by a half compared to the conventional convolution-based approach. The application of integer-polynomial mapping (IPM) of the polynomial filter coefficient with a floating-point value further decreases the complexity and reduces the loss in signal

reconstruction. Also, the “resource sharing” between lifting steps results in a further reduction in implementation costs and near-lossless data reconstruction.

In the second phase, a completely lossless or error-free architecture has been proposed for the Daubechies-8 (D8) wavelet. Several lifting variants have been derived for the same wavelet, the integer mapping has been applied, and the best variant is determined in terms of performance, using entropy and transform coding gain. Then a theory has been derived regarding the impact of scaling steps on the transform coding gain *(G_T)*. The approach results in the lowest cost lossless architecture of the D8 in the literature, to the best of our knowledge. The proposed approach may be applied to other orthogonal wavelets, including biorthogonal ones to achieve higher performance.

In the final phase, a general algorithm has been proposed to implement the original filter coefficients expressed by a polyphase matrix into a more efficient lifting structure. This is done by using modified factorization, so that the factorized polyphase matrix does not include the lossy scaling step like the conventional lifting method. This general technique has been applied on some widely used orthogonal and biorthogonal wavelets and its advantages have been discussed.

Since the discrete wavelet transform is used in a vast number of applications, the proposed algorithms can be utilized in those cases to achieve lossless, low-cost, and hardware-friendly architectures.

Acknowledgment

I would like to start praising the Almighty God, Allah, for His amazing grace, power, and presence that helped me to finalize this thesis.

I wish to express my gratitude and appreciation to my supervisor Prof. Khan Wahid, who provided me the guidance and advice during my Ph.D. program at the University of Saskatchewan. I also wish to thank the advisory committee member for their suggestions and advice. The assistance provided by the laboratory and secretarial staff of the Department of Electrical and Computer Engineering is gratefully recognized as well.

I would also to express my special thanks to my family in Canada and my parents in Bangladesh for their support. I must acknowledge my wife, Shohely Akkas, for her continuous support, without whose care, encouragement and help, I could not finish this thesis. My thanks also go to my son, Saad Hasan and my daughter, Samreen Hasan, I could not give them more family time.

I also thank "Multimedia Processing and Prototyping Laboratory" for using the different facilities, equipment and Canadian Microelectronics Corporation (CMC) for providing development tools and technical support.

I am also grateful for the financial support provided through different awards and scholarships by my supervisor, the Natural Sciences and Engineering Research Council of Canada, Department of Electrical and Computer engineering, College of Graduate and Postdoctoral Studies of the University of Saskatchewan and the province of Saskatchewan.

Dedication

Dedicated to

my dear children

Saad and Samreen Hasan

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CDF	Cohen–Daubechies–Feauveau wavelet
CR	Compression ratio
CSD	Canonical Signed Digit
CWT	Continuous wavelet transform
D4	Daubechies 4-tap wavelet, also expressed as db2
D6	Daubechies 6-tap wavelet, also expressed as db3
D8	Daubechies 8-tap wavelet, also expressed as db4
Daub	Daubechies wavelet
DCT	Discrete Cosine Transform
DWT	Discrete wavelet transform
ECG	Electrocardiogram
EEG	Electroencephalogram
FFT	Fast Fourier Transform
fMRI	Functional Magnetic resonance imaging
FPGA	Field-programmable-gate-array
GCD	Greatest common divisor
HPF	High-pass filter
IPM	Integer polynomial mapping
JPEG	Joint Photographic Experts Group
LPF	Low-pass filter
MRA	Multiresolution analysis
MRI	Magnetic resonance imaging
PSNR	Peak signal-to-noise ratio
SIPI	Signal and Image Processing Institute (University of Southern California)
SPIHT	Set partitioning in hierarchical trees
STFT	Short-term Fourier Transform
WFT	Windowed Fourier Transform

List of Symbols

$w(t)$	Window function
$\psi(t)$	Wavelet function
φ(t)	Scaling function
$\tilde{g}$	High-pass analysis filter
$\tilde{h}$	Low-pass analysis filter
g	High-pass synthesis filter
h	Low-pass synthesis filter
X_H	High-pass filter output
X_L	Low-pass fitter output
d	High-pass filter output
s	Low-pass fitter output
$P(z)$	Polyphase matrix
k	Scaling factor of the scaling step in a lifting scheme
$q_i(z)$	Quotient in each stage of the successive division
G_T	Transform coding gain
σ^2	Variance
μ	Mean

Chapter 1

Introduction

1.1 Introduction

Digital multimedia system is an inseparable part of daily life in the present world. From the advent of the telegraph, telephone, radio, internet-connected computer to the current smartphones, ubiquitous access to information signals has gradually become an important necessity of modern life [1]. In engineering terminology, a signal is a function that conveys information about the attributes of some phenomenon [2] or observable change in a measurable entity [3]. Some examples of types of signals are audio, video, speech, image, geophysical, sonar, radar, medical, and musical [4]. In order to make signals useful in different applications, they need to be processed by analyzing, coding, compressing and filtering. In signal processing, transforms like the Fast Fourier transform (FFT), windowed Fourier transform (WFT) or short-term Fourier transform (STFT), discrete cosine transforms (DCT), and, recently, discrete wavelet transform (DWT) are playing vital roles.

In day-to-day applications, most of the signals and related data used are aperiodic, irregular, noisy, and transient. When it comes to analyzing such signals and data in real time, the wavelet transform is particularly useful compared to its counterparts. This is because it has the capability of exploring signals concurrently in both the frequency and time domain [5] and analyzing the different frequency and time components in different resolutions which is called multiresolution analysis (MRA).

Due to its powerful characteristics of a concurrent time-frequency domain and multiresolution analysis, the wavelet transform has experienced rapid development in the last 35 years in both research and applications [6]. Initially, in the mid-1970, the wavelet transform was

proposed by Jean Morlet as an engineering solution to solve problems in analyzing geophysical data. These problems were not satisfactorily solvable by the classical tools, such as the Fourier transform and its derivatives like STFT. Later, wavelets were established based on powerful mathematical theories [6] which widely extended their application in many different fields. Due to their strong mathematical foundation in concurrent time-frequency domain analysis and multi-resolution analysis, they are extensively used in a wide number of fields in science, geophysics, astrophysics, biology, mathematics, computer science, medicine, finance, and engineering [5], [7], [8] and have already been adopted in the JPEG2000 image compression standard [9].

Because of the heavy usage of the wavelet transform, especially the DWT due to the necessity to analyze the abundant discrete time signals, its implementation is an important topic. While many implementation methods are explored for different purposes, they can be classified broadly into two categories: convolution based and lifting based. Lifting-based schemes are comparatively better in terms of cost. Integer-based lifting structures are more hardware-implementation friendly and allow lossless transform for a few select wavelets.

This research aims to propose a general low-cost and lossless architecture, and starts with an examination of lifting architectures. Existing implementations have some flaws which prevent losslessness for general wavelets. This research offers a solution to those flaws.

1.2 Motivation

The wavelet transform solves the incompatibility of the Fourier transform with the many practical signals which are non-stationary, and provides a means for MRA of time-frequency components of a signal. This capability makes the wavelet transform useful for many different applications. Despite its usefulness, the current implementation has a few issues. The usefulness of the wavelet transforms and appeal of better implementation by fixing the existing issues, as detailed in the following subsections, motivate this research.

1.2.1 Usefulness

The wavelet transform is able to harness a signal or data sequence and find its frequency and time information simultaneously enabling it to be used in a variety of applications. This

makes the wavelet transform very attractive for implementation. Besides, it is a powerful statistical tool which also makes it useful for a wide range of applications. Some of its uses are discussed below.

Compression

In the present world, the compression of image and video is of great interest due to the increasing number of high-definition images and videos, the limitation of the transmission bandwidth and the need to store more data in the cloud. Though the discrete cosine transform (DCT) is common for image compression, such as JPEG compression, the discrete wavelet transform is getting more attention in this area due to some limitations of the DCT, such as block artifacts. JPEG2000 has been implemented with the DWT and there have been a number of attempts in the literature to compress different types of signals using DWT. Wavelet-based compressions have been applied to single-dimensional (1D) signals or data including electroencephalograms (EEGs) [10], electrocardiograms (ECGs) [11], speech signals [12], two-spatial dimensional signals such as general images [13]–[18] and finger-prints by the FBI [19], three-dimensional signals or data like videos [16], remote-sensing data [20], multispectral images [21] or hyperspectral images [22]. The wavelet transform also has applications for lossless images or data compression [23], [24] which is vital for a few applications such as medical images, where traditional DCT based JPEG cannot be used due to its lossy nature.

General signal processing

Apart from compression, the wavelet transform is also used for a variety of other signal processing applications [25], [26]. Since it is able to decompose the signal into low-frequency approximations and high-frequency details, it can be used to denoise or reduce the trend [8]. There have been a significant number of works to denoise signals in general [27] and ECG signals in particular [28]. It has been used in feature extraction [29], texture analysis [30], and video watermarking[31]. In addition, it has been applied in audio and speech processing [32], speech recognition [33], pitch detection [34], segmentation [35], enhancement [36], pattern recognition [37], and detection of gear failure via acoustic signals [38].

Image processing is one of the classic applications [39]. The wavelet transform has been useful in image resolution enhancement [40], digital watermarking [31], and multisensor image fusion [41].

Top areas of application

The wavelet transform can be used effectively in the most, if not all, applications with non-stationary signals because of its ability to find both time and scale (an alternative to frequency) information with diverse resolution, and because it is not limited to only a single wavelet. As a result, there are many areas of applications where the wavelet transform is being heavily used. Some of the top areas are as follows.

(a) Medical application

Medical images play a vital role in clinical diagnosis and therapy[42]. The wavelet transform can be for speckle noise reduction [43], enhancement [44], and multimodal image fusion [45]. Other imaging areas where the wavelet transform can play a role are detection of brain tumors [46], detection of microcalcification in mammography [47], analysis of magnetic resonance imaging (MRI) [48] and X-rays [49], localization of patterns of activity in the brain using functional images such as positron emission tomography (PET) and functional MRI [50].

As discussed before, the wavelet transform has been applied in ECG compression and noise reduction. In addition, it is used to extract useful information from ECG signals. ECG is a measure of the electrical activity of the heart [51]. It is the standard method for the investigation of irregularities in heartbeats [5]. The duration of the electrical wave determines whether the heartbeat is normal, slow, or irregular, and the amount of electrical activity shows whether parts of the hearts are too large or overworked [52].

Wavelet analysis also has been applied to detect the characteristic features of ECG signals [29], to facilitate the automated diagnosis of cardiac health issues [53] or abnormalities [54], such as coronary artery diseases [55], sudden cardiac death [56] and myocardial ischemia [57]. Wavelet transforms also have been used for ECG signals for the enhancement of late potentials [58], ECG-beat classification [37], [59], and arrhythmia classification [60].

(b) Remote sensing information and satellite image processing

Satellite images and remotely sensed data are useful in numerous areas, including astronomy, geoscience studies, geographical information systems, land surveying, and most earth science disciplines [61]. Wavelet transforms have been utilized to enhance resolution [40] and contrast [62] of satellite images, and remote sensing image fusion [63].

(c) Geophysical applications

The wavelet transform originated because of the need for analysis of seismic signals and has been used in the analysis of many other geophysical processes [8], such as atmospheric turbulence [64], large-scale atmospheric circulation [65], ocean waves [66], seafloor bathymetry [67], earthquakes [68] and climate change [69].

(d) Power systems

Wavelet transforms have been utilized for the analysis of harmonic distortion in power systems as reviewed in [70]. Wavelet analysis is used to extract disturbances in power systems [71], to determine reference compensation current of shunt active power filters [72], and to control the DC-link voltage of a shunt active power filter [73] using the application of Daubechies 4-tap wavelets with 24 decomposition levels to a DC voltage error signal.

1.2.2 Appeal of lossless implementation

While in some applications, lossy implementation of the wavelet transform can be used, there are many applications, where lossless implementation is crucial. Such applications are discussed in the following sections.

Medical image/signal compression

Clinical picture archiving and communication systems (PACS) and telemedicine or teleradiology networks are used for the storage, retrieval, transmission, distribution, and visualization of large collections of medical image data, and compression of them is necessary [74]. Since the amount of three-dimensional images is increasing due to the increased usage of diagnostic techniques such as computed tomography (CT), magnetic resonance (MR), and positron emission tomography (PET), the efficient compression of these data is crucial.

Though lossy compression of medical images is usable in some cases, medical professionals prefer to use lossless methods [75]. This is because many physicians have concerns that lossy compression may result in errors in diagnosis. In addition, there are some legal and regulatory concerns which recommend lossless compression [76] and lossy compressions are not accepted for some cases like coronary angiogram [77].

In addition to medical images, for medical signals such as ECG or EEG signals, lossless compression is desirable for legal considerations and serious diagnoses [78].

Reversible data hiding and digital watermarking

Data hiding is used for covert communications and storage where the data is hidden in a cover media file such as image[79]. It has an additional advantage over cryptography because it hides the existence of secret information [79]. Digital watermarking is a similar process where a sequence of digital bits are embedded into a media file for copyright protection [80].

Many data hiding and watermarking techniques distort the cover media file in order to add the hidden data or watermark. However, in some applications, such as medical imaging and the legal enforcement field, irreversibility is not desirable due to legal considerations and sensitivity of the data [81]. In these applications, lossless or reversible data hiding, and watermarking are necessary.

For lossless data hiding and watermarking, the lossless wavelet transform is one of the most commonly used methods [79]. Some recent significant works on this subject are [81], [82] by Xuan *et al*. and [83]–[85] by other researchers.

Compression of satellite data

Satellite data are often collected and processed later to get information regarding deforestation or vegetation. If lossy compression is used for storage, the enhancement of the reconstructed data may result in enhancement of the errors, and important details can be missing. Therefore, it is preferable to use lossless compression for satellite images in storage [86] since it may be costly or not possible to get the same data again [87].

Text or document compression

Lossless compression is vital for text compression. In much text compression, very small losses or changes of information can change the whole purpose. A single-digit change in banking information or a small change in other important files can have a detrimental impact. In texting, it is unacceptable that a sentence like “Do not send money” to change into “Do now send money” because of using a lossy compression [87]. Similarly, scanned images of old handwritten manuscripts also demand lossless compression [86] to avoid loss of information.

Areas to be explored

There are many areas described in section 1.2.1 where lossy or the continuous/analog wavelet transform is used. Those areas can be explored for better quality processing since the disk-space capability and demand for quality processing are increasing. Such examples are an analog wavelet transform for a pacemaker application where a lossless digital wavelet transform needs to be studied.

1.2.3 Appeal of integer-based implementation

Integer-based implementations are preferred in many cases where discrete wavelet transforms need to be implemented in hardware. Floating-point calculations are expensive to implement in hardware and slow for software implementation. Many applications demand the algorithm be implemented in a field-programmable-gate-array (FPGA) in order to have the ability to update the algorithm when there may be a need to modify it. FPGA hardwires usually do not support floating-point calculations. For this reason, the wavelet algorithm needed for implementation in FPGA is should be integer-based. Moreover, since integer-based implementations are comparatively low-cost, they are preferable in many applications.

1.2.4 Issues in the existing implementation

There are two major techniques for implementing discrete wavelet transforms: the ladder-structured lifting scheme and convolution or matrix-based techniques (see Chapter 2 for details). The lifting scheme is preferred for hardware implementation for several reasons. It reduces the

cost by half, speeds up the transform, and allows for in-place implementation (i.e. it does not require any auxiliary memory). In addition, it can be used to build an integer-based wavelet transform that maps integer-to-integer. The integer-based lifting transform is hardware-implementation friendly and paves the way for lossless implementation.

Though the integer-based lifting wavelet transform allows lossless implementation for few wavelets, such as orthogonal Haar and biorthogonal CDF 5/3 wavelets, the implementation of most orthogonal and biorthogonal wavelets using the same technique cannot be lossless. They remain lossy even though most of the lifting steps become lossless in integer-based methods. This is because the last step known as the scaling step remains irreversible due to a division/multiplication operation by the scaling coefficient *k* (see chapter 2). This is a known issue and can be fixed by conversion of the scaling step into additional lifting steps, but it increases hardware cost and introduces more delay in the processing of the transform; that is one of the reasons why this solution is not used much in the literature. To best of the knowledge, to date, there are no other options.

Most researchers who need to implement lossless wavelets for different applications rely on only a few selective wavelets which inherently support lossless integer-based lifting wavelet transforms because the value of the scaling coefficient *k* is 1 for those wavelets. Therefore, a general approach is needed that can be utilized for any orthogonal or biorthogonal wavelets using low-cost lifting scheme in order to take advantage of diverse energy compactness of different wavelets and implement them in a lossless and low-cost manner.

1.2.5 Motivation summary

The wavelet transform is a mathematical or statistical toolbox, which can be used for a vast variety of applications as described in subsection 1.2.1. In many applications, lossless implementation of the discrete wavelet is necessary or desired, as outlined in subsection 1.2.2. But, as discussed in section 1.2.3, the status quo suggests that the scope of lossless implementation is very limited; only a few wavelets support purely lossless implementations. However, different wavelets have different levels of energy compactness; some work better in some applications than others. For this reason, it is important to take advantage of the diverse properties of all different orthogonal and biorthogonal wavelets in different applications. But the

implementation in a lossless manner with low-cost lifting scheme is not an easy option for most of them, and so, this research aims to overcome the obstacle and sets objectives as described in section 1.3.

1.3 Research Objectives

Based on the discussion in section 1.2 regarding the issues in existing conventional implementation and recent developments, this study proposes a general approach for an integer-based lossless lifting wavelet transform. The research objectives are as follows:

- The implementation should be lossless and reversible. To be able to generalize, the technique should be based on a mathematical foundation which would be experimentally examined. To facilitate this, in an integer-based implementation (i.e. a wavelet transform technique that maps integer-to-integer), the loss-causing scaling steps should be altered, based on a mathematical foundation, and the change should not affect performance.
- The implementation techniques or algorithms should result in a low-cost implementation of a DWT when compared with similar wavelets in the literature and with a similar capability, if it exists. Lower cost can be achieved with a combination of low-cost-proven lifting schemes that map integer-to-integer, the integer-polynomial mapping (i.e., integer conversion of the expensive floating-points polynomial coefficients of the filter), and low-cost management of the scaling steps.
- The proposed algorithm should be generalized and will be examined by an application to popular wavelets, such as the orthogonal Daubechies-4,-6,-8,-10 and biorthogonal wavelets such as the CDF-9/7 and CDF4.2. The effort will result in various efficient architectures, and some examples can be proposed.

1.4 Thesis Organization

This thesis is organized in a manuscript-based style. Published or submitted manuscripts are included in this thesis as chapters. In each chapter, a brief introduction precedes each

manuscript to connect it to the main context of the thesis. The organization of different chapters is as follows.

Chapter 2 presents some basic background knowledge regarding wavelet transform and how, why and from what it evolved. The chapter starts with the early development of the analysis of signals in the frequency domain and how they are useful for analyzing different types, such as stationary and non-stationary signals. The Fourier transform and its issues with analyzing non-stationary signals are discussed briefly as well as the short-term solution called the windowed or short-term Fourier transform (STFT). Next, the need of multiresolution analysis (MRA) and why STFT was not able to meet the need of MRA is discussed, and how this problem led to the idea of the Wavelet transform in the mid-1970s. Then a description of the wavelet transforms, both continuous and discrete is given as well as a discussion of the usage of subband coding in the discrete wavelet transform. Finally, the implementation techniques of wavelet transform and issues with existing implementations, which have motivated a proposal for a general low-cost lossless implementation method for orthogonal and biorthogonal wavelets in this study, are presented.

Following the background study, the first step in this research is to explore low-cost implementation methods. Daubechies-4 and 6 wavelets were used for this purpose. Chapter 3 describes this work and present the manuscript [88] titled “Low-cost Architecture of Modified Daubechies Lifting Wavelets Using Integer Polynomial Mapping.” In this work, a low-cost lifting scheme was used as the preferred implementation technique. Integer-polynomial mapping with resource sharing or coefficient elimination was applied to ensure even lower cost implementation.

Chapter 4 investigates the reasons for the loss in the lossy integer-based implementation and proposes a solution which suggests elimination of the scaling steps and the careful selection of factorizations of polyphase matrices. This solution has been applied to Daubechies-8 wavelet. This chapter also elaborates the non-unique process of factorization which would enable the reader or future researcher to follow the steps and find other options. In this chapter, pure losslessness is achieved.

Chapter 5 provides a more generalized solution. Though the technique used in chapter 4 can be utilized for other wavelets, it requires much manual work. In chapter 5, a change is made

to the method of factorization of the polyphase matrix in a lifting-based wavelet transform. Conventionally, it uses the Euclidian algorithm for Laurent polynomials. This chapter proposes a new factorization technique which does not produce any lossy scaling steps ensuring lossless integer-based implementations and incorporates the impact of those steps in the previous lifting steps. This also provides some formula for converting the existing factorization into the proposed factorization. This general technique is applied and tested on a number of orthogonal and biorthogonal wavelets.

Finally, chapter 6 summarizes the whole thesis, suggests research problems for future works and draws some conclusions.

1.5 References

[1] Liang-Gee Chen, *VLSI design of wavelet transform*. London: Imperial College Press, 2007.

[2] R. Priemer, *Introductory signal processing*, vol. 6. World Scientific Publishing Company, 1990.

[3] P. Chakravorty, "What Is a Signal? [Lecture Notes]," *IEEE Signal Process. Mag.*, vol. 35, no. 5, pp. 175–177, 2018.

[4] "Aims & Scope," *IEEE Transactions on Signal Processing*. .

[5] P. S. Addison, *The illustrated wavelet transform handbook: introductory theory and applications in science, engineering, medicine and finance*. CRC press, 2002.

[6] A. K. Louis, D. Maass, and A. Rieder, *Wavelets: Theory and Applications*. Wiley, 1997.

[7] M. Vetterli and C. Herley, "Wavelets and filter banks: Theory and design," *IEEE Trans. signal Process.*, vol. 40, no. 9, pp. 2207–2232, 1992.

[8] M. Misiti, Y. Misiti, G. Oppenheim, and J.-M. Poggi, *Wavelets and their Applications*. John Wiley & Sons, 2013.

[9] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression

standard," *IEEE Signal Process. Mag.*, vol. 18, no. 5, pp. 36–58, 2001.

[10] J. L. Cárdenas-Barrera, J. V Lorenzo-Ginori, and E. Rodríguez-Valdivia, "A wavelet-packets based algorithm for EEG signal compression," *Med. Inform. Internet Med.*, vol. 29, no. 1, pp. 15–27, 2004.

[11] S.-C. Tai, C. Sun, and W.-C. Yan, "A 2-D ECG compression method based on wavelet transform and modified SPIHT," *IEEE Trans. Biomed. Eng.*, vol. 52, no. 6, pp. 999–1008, 2005.

[12] G. Rajesh, A. Kumar, and K. Ranjeet, "Speech compression using different transform techniques," in *Computer and Communication Technology (ICCCT), 2011 2nd International Conference on*, 2011, pp. 146–151.

[13] P. Telagarapu, V. J. Naveen, A. L. Prasanthi, and G. V. Santhi, "Image compression using DCT and wavelet transformations," *Int. J. Signal Process. Image Process. Pattern Recognit.*, vol. 4, no. 3, pp. 61–74, 2011.

[14] H. B. Kekre, T. K. Sarode, and S. D. Thepade, "Inception of hybrid wavelet transform using two orthogonal transforms and it's use for image compression," *Int. J. Comput. Sci. Inf. Secur.*, vol. 9, no. 6, p. 80, 2011.

[15] M. Alemohammad, J. R. Stroud, B. T. Bosworth, and M. A. Foster, "High-speed all-optical Haar wavelet transform for real-time image compression," *Opt. Express*, vol. 25, no. 9, pp. 9802–9811, 2017.

[16] P. N. Topiwala, *Wavelet image and video compression*, vol. 450. Springer Science & Business Media, 2006.

[17] C.-L. Chang and B. Girod, "Direction-adaptive discrete wavelet transform for image compression," *IEEE Trans. Image Process.*, vol. 16, no. 5, pp. 1289–1302, 2007.

[18] V. V. S. Kumar and M. I. S. Reddy, "Image compression techniques by using wavelet transform," *J. Inf. Eng. Appl.*, vol. 2, no. 5, pp. 35–39, 2012.

[19] J. N. Bradley, C. M. Brislawn, and T. Hopper, "FBI wavelet/scalar quantization standard

for gray-scale fingerprint image compression," in *Visual Information Processing II*, 1993, vol. 1961, pp. 293–305.

[20] B. Li, R. Yang, and H. Jiang, "Remote-sensing image compression using two-dimensional oriented wavelet transform," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 1, pp. 236–250, 2011.

[21] P. L. Dragotti, G. Poggi, and A. R. P. Ragozini, "Compression of multispectral images by three-dimensional SPIHT algorithm," *IEEE Trans. Geosci. Remote Sens.*, vol. 38, no. 1, pp. 416–428, 2000.

[22] B. Penna, T. Tillo, E. Magli, and G. Olmo, "Transform coding techniques for lossy hyperspectral data compression," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 5, pp. 1408–1421, 2007.

[23] A. A. Kassim, P. Yan, W. S. Lee, and K. Sengupta, "Motion compensated lossy-to-lossless compression of 4-D medical images using integer wavelet transforms," *IEEE Trans. Inf. Technol. Biomed.*, vol. 9, no. 1, pp. 132–138, 2005.

[24] Z. Xiong, X. Wu, S. Cheng, and J. Hua, "Lossy-to-lossless compression of medical volumetric data using three-dimensional integer wavelet transforms," *IEEE Trans. Med. Imaging*, vol. 22, no. 3, pp. 459–470, 2003.

[25] A. Bruce, D. Donoho, and H.-Y. Gao, "Wavelet analysis [for signal processing]," *IEEE Spectr.*, vol. 33, no. 10, pp. 26–35, 1996.

[26] S. Mallat, *A wavelet tour of signal processing: the sparse way*. Academic press, 2008.

[27] R. Aggarwal, J. K. Singh, V. K. Gupta, S. Rathore, M. Tiwari, and A. Khare, "Noise reduction of speech signal using wavelet transform with modified universal threshold," *Int. J. Comput. Appl.*, vol. 20, no. 5, pp. 14–19, 2011.

[28] M. A. Kabir and C. Shahnaz, "Denoising of ECG signals based on noise reduction algorithms in EMD and wavelet domains," *Biomed. Signal Process. Control*, vol. 7, no. 5, pp. 481–489, 2012.

[29] S. Banerjee, R. Gupta, and M. Mitra, “Delineation of ECG characteristic features using multiresolution wavelet analysis method,” *Measurement*, vol. 45, no. 3, pp. 474–487, 2012.

[30] T. Chang and C.-C. Kuo, “Texture analysis and classification with tree-structured wavelet transform,” *IEEE Trans. image Process.*, vol. 2, no. 4, pp. 429–441, 1993.

[31] G. V Mane and G. G. Chiddarwar, “Review paper on video watermarking techniques,” *Int. J. Sci. Res. Publ.*, vol. 3, no. 4, pp. 1–5, 2013.

[32] R. Kronland-Martinet, “The wavelet transform for analysis, synthesis, and processing of speech and music sounds,” *Comput. Music J.*, vol. 12, no. 4, pp. 11–20, 1988.

[33] J. N. Gowdy and Z. Tufekci, “Mel-scaled discrete wavelet coefficients for speech recognition,” in *Acoustics, Speech, and Signal Processing, 2000. ICASSP’00. Proceedings. 2000 IEEE International Conference on*, 2000, vol. 3, pp. 1351–1354.

[34] S. Kadambe and G. F. Boudreaux-Bartels, “Application of the wavelet transform for pitch detection of speech signals,” *IEEE Trans. Inf. Theory*, vol. 38, no. 2, pp. 917–924, 1992.

[35] C. Wendt and A. P. Petropulu, “Pitch determination and speech segmentation using the discrete wavelet transform,” in *Circuits and Systems, 1996. ISCAS’96., Connecting the World., 1996 IEEE International Symposium on*, 1996, vol. 2, pp. 45–48.

[36] Y. Hu and P. C. Loizou, “Speech enhancement based on wavelet thresholding the multitaper spectrum,” *IEEE Trans. Speech Audio Process.*, vol. 12, no. 1, pp. 59–67, 2004.

[37] S. Banerjee and M. Mitra, “Application of cross wavelet transform for ECG pattern analysis and classification,” *IEEE Trans. Instrum. Meas.*, vol. 63, no. 2, pp. 326–333, 2014.

[38] N. Baydar and A. Ball, “Detection of gear failures via vibration and acoustic signals using wavelet transform,” *Mech. Syst. Signal Process.*, vol. 17, no. 4, pp. 787–804, 2003.

[39] J.-L. Starck, F. Murtagh, and J. M. Fadili, *Sparse image and signal processing: wavelets,*

curvelets, morphological diversity. Cambridge university press, 2010.

[40] H. Demirel and G. Anbarjafari, "Discrete wavelet transform-based satellite image resolution enhancement," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 6, pp. 1997–2004, 2011.

[41] K. Amolins, Y. Zhang, and P. Dare, "Wavelet based image fusion techniques—An introduction, review and comparison," *ISPRS J. Photogramm. Remote Sens.*, vol. 62, no. 4, pp. 249–263, 2007.

[42] A. Aldroubi, *Wavelets in medicine and biology*. Routledge, 2017.

[43] H. Rabbani, M. Vafadust, P. Abolmaesumi, and S. Gazor, "Speckle noise reduction of medical ultrasound images in complex wavelet domain using mixture priors," *IEEE Trans. Biomed. Eng.*, vol. 55, no. 9, p. 2152, 2008.

[44] Y. Yang, Z. Su, and L. Sun, "Medical image enhancement algorithm based on wavelet transform," *Electron. Lett.*, vol. 46, no. 2, pp. 120–121, 2010.

[45] V. Bhateja, H. Patel, A. Krishn, A. Sahu, and A. Lay-Ekuakille, "Multimodal medical image sensor fusion framework using cascade of wavelet and contourlet transform domains," *IEEE Sens. J.*, vol. 15, no. 12, pp. 6783–6790, 2015.

[46] A. Kharrat, N. Benamrane, M. Ben Messaoud, and M. Abid, "Detection of brain tumor in medical images," in *Signals, Circuits and Systems (SCS), 2009 3rd International Conference on*, 2009, pp. 1–6.

[47] R. N. Strickland and H. Il Hahn, "Wavelet transforms for detecting microcalcifications in mammograms," *IEEE Trans. Med. Imaging*, vol. 15, no. 2, pp. 218–229, 1996.

[48] K. Hackmack, F. Paul, M. Weygandt, C. Allefeld, J.-D. Haynes, and A. D. N. Initiative, "Multi-scale classification of disease using structural MRI and wavelet transform," *Neuroimage*, vol. 62, no. 1, pp. 48–58, 2012.

[49] S. Zhao, D. D. Robeltson, G. Wang, B. Whiting, and K. T. Bae, "X-ray CT metal artifact reduction using wavelets: an application for imaging total hip prostheses," *IEEE Trans.*

Med. Imaging, vol. 19, no. 12, pp. 1238–1247, 2000.

[50] M. Unser and A. Aldroubi, "A review of wavelets in biomedical applications," *Proc. IEEE*, vol. 84, no. 4, pp. 626–638, 1996.

[51] P. S. Addison, "Wavelet transforms and the ECG: a review," *Physiol. Meas.*, vol. 26, no. 5, p. R155, 2005.

[52] C. Saritha, V. Sukanya, and Y. N. Murthy, "ECG signal analysis using wavelet transforms," *Bulg. J. Phys*, vol. 35, no. 1, pp. 68–77, 2008.

[53] R. J. Martis, U. R. Acharya, K. M. Mandana, A. K. Ray, and C. Chakraborty, "Application of principal component analysis to ECG signals for automated diagnosis of cardiac health," *Expert Syst. Appl.*, vol. 39, no. 14, pp. 11792–11800, 2012.

[54] P. De Chazal, M. O'Dwyer, and R. B. Reilly, "Automatic classification of heartbeats using ECG morphology and heartbeat interval features," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 7, pp. 1196–1206, 2004.

[55] D. Giri, U.R. Acharya, R.J. Martis, S.V. Sree, T.C. Lim, T.A. VI, and J.S. Suri, "Automated diagnosis of coronary artery disease affected patients using LDA, PCA, ICA and discrete wavelet transform," *Knowledge-Based Syst.*, vol. 37, pp. 274–282, 2013.

[56] J. P. Amezquita-Sanchez, M. Valtierra-Rodriguez, H. Adeli, and C. A. Perez-Ramirez, "A Novel Wavelet Transform-Homogeneity Model for Sudden Cardiac Death Prediction Using ECG Signals," *J. Med. Syst.*, vol. 42, no. 10, p. 176, 2018.

[57] P. Ranjith, P. C. Baby, and P. Joseph, "ECG analysis using wavelet transform: application to myocardial ischemia detection," *ITBM-RBM*, vol. 24, no. 1, pp. 44–47, 2003.

[58] O. Meste, H. Rix, R. Jané, and P. Caminal, "Detection of late potentials by means of wavelet transform," in *Engineering in Medicine and Biology Society, 1989. Images of the Twenty-First Century., Proceedings of the Annual International Conference of the IEEE Engineering in*, 1989, pp. 28–29.

[59] R. J. Martis, U. R. Acharya, and L. C. Min, "ECG beat classification using PCA, LDA,

ICA and discrete wavelet transform," *Biomed. Signal Process. Control*, vol. 8, no. 5, pp. 437–448, 2013.

[60] J.A. Gutiérrez-Gnecchia, R. Morfin-Magañaa, D. Lorias-Espinozab, A.d.C. Tellez-Anguianoa, E. Reyes-Archundiaa, A. Méndez-Patiñoa, R. Castañeda-Mirandac, "DSP-based arrhythmia classification using wavelet transform and probabilistic neural network," *Biomed. Signal Process. Control*, vol. 32, pp. 44–56, 2017.

[61] H. Demirel and G. Anbarjafari, "Satellite image resolution enhancement using complex wavelet transform," *IEEE Geosci. Remote Sens. Lett.*, vol. 7, no. 1, pp. 123–126, 2010.

[62] H. Demirel, C. Ozcinar, and G. Anbarjafari, "Satellite image contrast enhancement using discrete wavelet transform and singular value decomposition," *IEEE Geosci. Remote Sens. Lett.*, vol. 7, no. 2, pp. 333–337, 2010.

[63] J. Cheng, H. Liu, T. Liu, F. Wang, and H. Li, "Remote sensing image fusion via wavelet transform and sparse representation," *ISPRS J. Photogramm. Remote Sens.*, vol. 104, pp. 158–173, 2015.

[64] L. Hudgins, C. A. Friehe, and M. E. Mayer, "Wavelet transforms and atmopsheric turbulence," *Phys. Rev. Lett.*, vol. 71, no. 20, p. 3279, 1993.

[65] S. Jevrejeva, J. C. Moore, P. L. Woodworth, and A. Grinsted, "Influence of large-scale atmospheric circulation on European sea level: results based on the wavelet transform method," *Tellus A Dyn. Meteorol. Oceanogr.*, vol. 57, no. 2, pp. 183–193, 2005.

[66] S. R. Massel, "Wavelet analysis for processing of ocean surface wave records," *Ocean Eng.*, vol. 28, no. 8, pp. 957–987, 2001.

[67] S. A. Little, "Wavelet analysis of seafloor bathymetry: An example," in *Wavelet Analysis and Its Applications*, vol. 4, Elsevier, 1994, pp. 167–182.

[68] K. Gurley and A. Kareem, "Applications of wavelet transforms in earthquake, wind and ocean engineering," *Eng. Struct.*, vol. 21, no. 2, pp. 149–167, 1999.

[69] K.-M. Lau and H. Weng, "Climate signal detection using wavelet transform: How to make

a time series sing," *Bull. Am. Meteorol. Soc.*, vol. 76, no. 12, pp. 2391–2402, 1995.

[70] J. Barros, R. I. Diego, and M. de Apraiz, "Applications of Wavelet Transform for Analysis of Harmonic Distortion in Power Systems: A Review," *IEEE Trans. Instrum. Meas.*, vol. 61, no. 10, pp. 2604–2611, 2012.

[71] A. Elnady, A. Goauda, and M. M. A. Salama, "Unified power quality conditioner with a novel control algorithm based on wavelet transform," in *Electrical and Computer Engineering, 2001. Canadian Conference on*, 2001, vol. 2, pp. 1041–1045.

[72] H. Liu, G. Liu, and Y. Shen, "A novel harmonics detection method based on wavelet algorithm for active power filter," in *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, 2006, vol. 2, pp. 7617–7621.

[73] M. Basu and B. Basu, "A wavelet controller for shunt active power filter," 2006.

[74] V. Sanchez, R. Abugharbieh, and P. Nasiopoulos, "Symmetry-based scalable lossless compression of 3D medical image data," *IEEE Trans. Med. Imaging*, vol. 28, no. 7, pp. 1062–1072, 2009.

[75] S. Wong, L. Zaremba, D. Gooden, and H. K. Huang, "Radiologic image compression-a review," *Proc. IEEE*, vol. 83, no. 2, pp. 194–219, 1995.

[76] A. Bilgin, G. Zweig, and M. W. Marcellin, "Efficient lossless coding of medical image volumes using reversible integer wavelet transforms," in *Data Compression Conference, 1998. DCC'98. Proceedings*, 1998, pp. 428–437.

[77] A. C. of Cardiology, "President's page: Developing standards for the digital age: The Dicom project." Journal of the American College of Cardiology, 1996.

[78] K. Srinivasan and M. R. Reddy, "Selection of optimal wavelet for lossless EEG compression for real-time applications," *Indian J. Biomech.*, pp. 241–245, 2010.

[79] R. Shaik, Ahmand ; Thanikaiselvan,V.; Amitharajan, "Data Security Through Data Hiding in Images: A Review," *J. Artif. Intell.*, vol. 10, no. 1, 2017.

[80] F. W. Alsaade, “Watermarking system for the security of medical image databases used in telemedicine,” *Res. J. Inform. Technol*, vol. 8, pp. 88–97, 2016.

[81] G. Xuan, J. Zhu, J. Chen, Y. Q. Shi, Z. Ni, and W. Su, “Distortionless data hiding based on integer wavelet transform,” *Electron. Lett.*, vol. 38, no. 25, pp. 1646–1648, 2002.

[82] G. Xuan, Q. Yao, C. Yang, J. Gao, P. Chai, Y. Q. Shi, and Z. Ni, “Lossless data hiding using histogram shifting method based on integer wavelets,” in *Digital Watermarking*, Springer, 2006, pp. 323–332.

[83] S. Sirsikar and J. Salunkhe, “Analysis of Data Hiding Using Digital Image Signal Processing,” in *2014 International Conference on Electronic Systems, Signal Processing and Computing Technologies*, 2014, pp. 134–139.

[84] W. Pan, G. Coatrieux, N. Cuppens, F. Cuppens, and C. Roux, “An additive and lossless watermarking method based on invariant image approximation and Haar wavelet transform,” in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, 2010, pp. 4740–4743.

[85] S. Lee, C. D. Yoo, and T. Kalker, “Reversible image watermarking based on integer-to-integer wavelet transform,” *IEEE Trans. Inf. Forensics Secur.*, vol. 2, no. 3, pp. 321–330, 2007.

[86] A. R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. B.-L. Yeo, “Lossless Image Compresion Using Integer to Integer Wavelet Transforms.,” in *in Proc. ICIP*, 1997, pp. 596–599.

[87] K. Sayood, *Introduction to data compression*. Morgan Kaufmann, 2017.

[88] M. M. Hasan and K. A. Wahid, “Low-Cost Architecture of Modified Daubechies Lifting Wavelets Using Integer Polynomial Mapping,” *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 64, no. 5, pp. 585–589, 2017.

Chapter 2

Background

2.1 Introduction

This chapter discusses the background of the theory of wavelet transforms starting with the early development of the Fourier series to find the harmonic content in a mathematical function or a signal. This discussion begins with the early concepts and follows through the elaborate steps to the modern developments of the continuous and discrete wavelet transform. The discussion also covers implementations of the discrete wavelet transform, including efficient integer-based implementations and their unsolved issues.

2.2 The Fourier Series and Transform

2.2.1 The Fourier series

The Fourier series is named after Jean-Baptiste Joseph Fourier (1768–1830). He introduced the series mainly to solve the heat equation that describes the distribution of heat in a metallic plate. In his 1822 book *Théorie Analytique de la Chaleur* (*Analytical theory of heat*) [1], [2], Fourier showed that the initial distribution of temperature can be expressed as a sum of many sine and cosine terms. This trigonometric series with its many developments has extended to many other fields and is commonly known as the Fourier series.

Presently the Fourier series is known as an expansion of a period function in terms of an infinite sum of sines and cosines. The Fourier series [3], [4] of a periodic function $f(t)$ of period T is represented as

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \left(a_k \cos \frac{2\pi kt}{T} + b_k \sin \frac{2\pi kt}{T} \right), \tag{2.1}$$

where $a_k = \frac{2}{T}\int_0^T f(t)\cos\frac{2\pi kt}{T}dt$, $b_k = \frac{2}{T}\int_0^T f(t)\sin\frac{2\pi kt}{T}dt$.

This is how the function $f(t)$ can be decomposed into a sine and cosine wave of $\frac{k}{T}$ frequencies where k=1,2, to ∞, and a_k or b_k determine the amplitudes of the waves.

2.2.2 The Fourier transform (FT)

While the Fourier series can analyze only periodic functions, the Fourier transform is the extension or application of the Fourier series which is able to analyze non-periodic functions or signals. The Fourier transform is the process of converting time-domain into frequency-domain signals. In other words, it decomposes a signal into its component frequencies and provides frequency-amplitude representation from time-amplitude representation. This transform is also called the frequency domain representation of a signal.

In mathematical term, the Fourier transform decomposes a signal $f(t)$ to complex exponentials of different frequencies. The Fourier transform $F(f)$ of the time-domain signal $f(t)$ can be found using the following equation where f and t represent frequency and time.

$$F(f) = \int_{-\infty}^{\infty} f(t)e^{-j2\pi ft}dt . \tag{2.2}$$

On the other hand, the inverse transform, which reproduces the original time domain signal $f(t)$, is as follows:

$$f(t) = \int_{-\infty}^{\infty} F(f)e^{j2\pi ft}df . \tag{2.3}$$

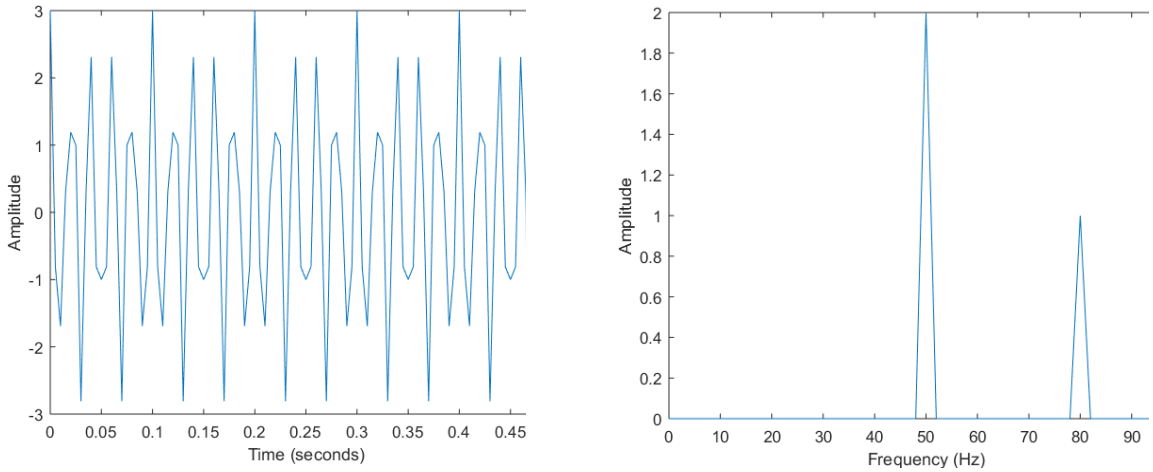
2.2.3 The Fourier transform's issue with non-stationary signals

If the frequency contents of a signal do not change over time, the signal is known as a stationary signal. In contrast, if it has varying frequencies, it is called a non-stationary signal. Examples of stationary signals are white noise, which is defined as a random signal with equal intensity at different frequencies [5], the sound generated by the smooth operation of a motor, and the noise voltage signals on the resistance of electronic equipment during stable operation.

On the other hand, many practical signals are non-stationary; two examples are the sound of a fireworks display, and a hammer blow[6].

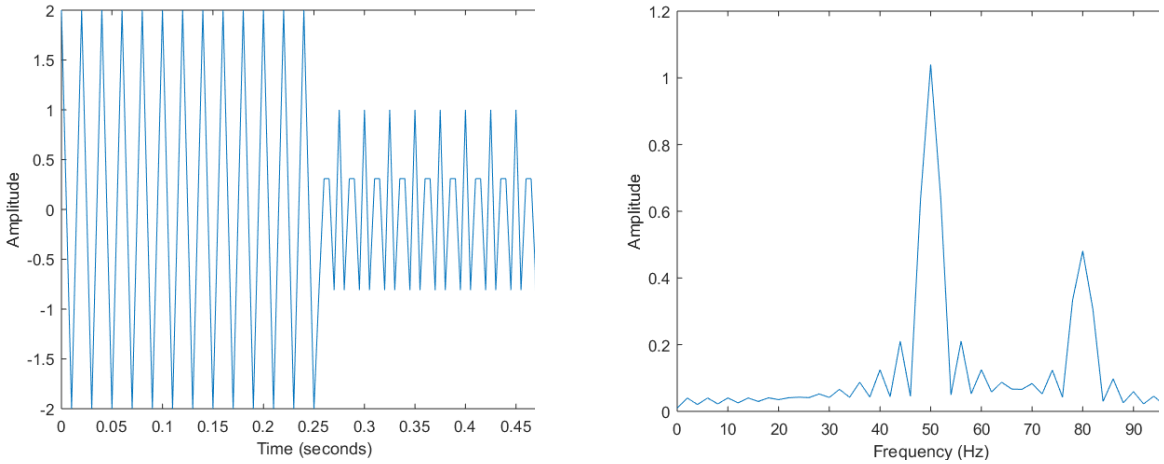
As evident from the equation (2.2), a specific frequency f_s component of a signal is the integral of the signal multiplied by a cosine and sine wave of that frequency f_s over the entire duration. This requires the presence of f_s over the entire duration in order to have the correct amplitude for f_s. Moreover, since the FT determines whether a certain frequency exists or not and does not say when that specific frequency appears [7], it is impossible to reproduce the original time-domain non-stationary signals.

This issue is explained in Fig. 2.1 with a stationary signal ($2\cos 2\pi 50t + \cos 2\pi 80t$) and in Fig. 2.2 with a non-stationary signal ($\begin{matrix} 2\cos 2\pi 50t \text{ when } t < 0.25 \\ \cos 2\pi 80t \quad \text{when } t \geq 0.25 \end{matrix}$) of the same frequency contents. Both signals have two frequencies, 50 Hz and 80 Hz. The Fourier transform is obtained by computation of the single-sided spectrum after the Fast Fourier transform (FFT) of the signals. Note that both spectrums peak at 50 and 80 Hz and the frequency content can be found in both cases, though the non-stationary spectrum at Fig. 2.2(b) has many small peaks. The small peaks represent some frequency contents which are not intended but they do exist in the input time-domain signal in Fig. 2.2 (a) and are due to the sudden changes between frequencies. If the minor frequencies are filtered out in the non-stationary spectrum in Fig. 2.2(b), the spectrum has the expected frequency contents. Therefore, the frequency information is apparent from the Fourier spectrum even for non-stationary signals, but there is no information regarding the time when the frequency changes. In other words, in the Fourier transform spectrum, there is no time information for each frequency component of the signal. Because of this, the input non-stationary signals cannot be reproduced exactly. Therefore, Fourier transform is not suitable for the analysis of non-stationary signals if time information is required and many of the practical signals are non-stationary. When the time localization of different frequencies is required for a non-stationary signal, a transform which has better time-frequency localization is needed. A possible potential answer to this issue, which is discussed in the next section, is the short-term Fourier transform.



(a) (b)

Fig. 2.1 (a) A stationary signal and (b) Fourier transform of the stationary signal.

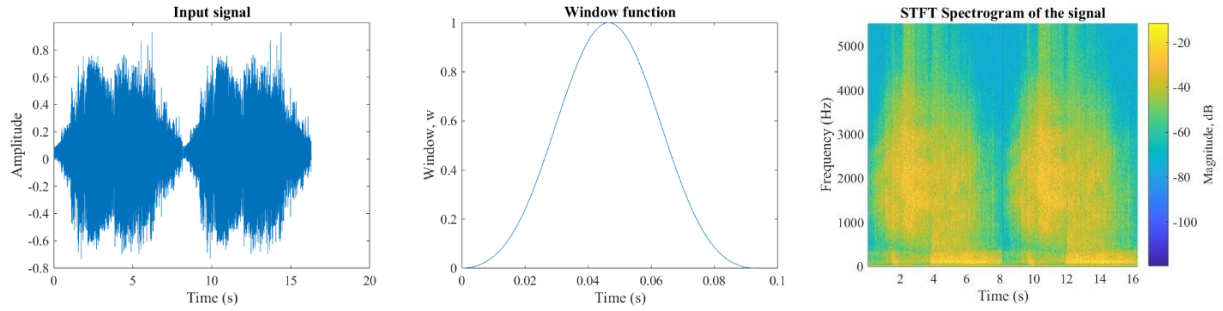


(a) (b)

Fig. 2.2 (a) A non-stationary signal with same frequency contents as Fig. 2.1 and (b) Fourier transform of the non-stationary signals (a).

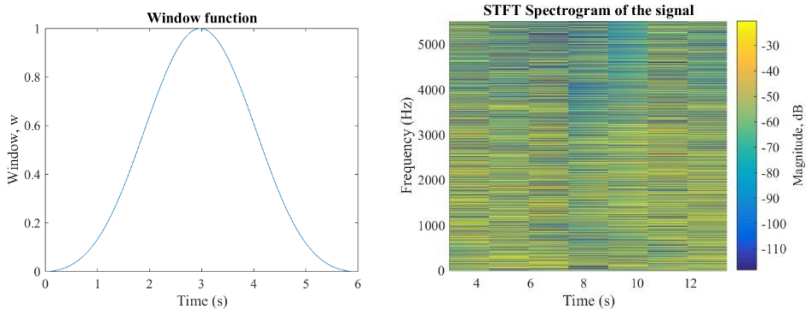
2.2.4 The Windowed Fourier transform (WFT) or Short-term Fourier transform (STFT)

The previous subsection discussed the reasons the Fourier transform is not suitable for non-stationary signals if time information is needed. However, a part of the non-stationary signal may be considered as stationary or close to stationary. In the windowed or short-term Fourier transform, the signal is divided into many small parts where each part is narrow enough to be considered as stationary. A window function $w(t)$ is used to divide the signal into parts. In 1946, D. Gabor [8] presented this windowed transform to analyze "frequency variations" of a sound [9].



(a) (b) (c)

Fig. 2.3 (a) Input signal (jet airplane sound), (b) window function, $w(t)$ and (c) short-term Fourier transform of the input sound signal. The time-axis values represent the midpoints of the parts of the signals selected by the window function; it does not show every timepoint available in the input signals.



(a) (b)

Fig. 2.4 Short-term Fourier transform of the same input signal as in Fig. 2.3, but with a larger window function. (a) Window function, (b) STFT spectrum.

Mathematically, the short-term Fourier transform of a signal $f(t)$ is given by

$$F(\tau, f) = \int_{-\infty}^{\infty} f(t) w(\tau - t) e^{-j2\pi ft} dt \,. \tag{2.4}$$

Fig 2.3 shows the short-term Fourier transform (discrete) spectrum of the input sound signal of a jet airplane sound collected from [10] with the shown window function, $w(t)$. The window function is traversed along the time axis to get different parts of the signal and Fourier transform is applied to get frequency and magnitude of that frequency for each small time-window. The results in the STFT spectrum is shown in Fig. 2.3. Thus, the short-term Fourier transform solves the issue of the time-locations of practical non-stationary signals and provides a time-frequency representation of the signals.

2.2.4 Issues of the short-term Fourier transform (STFT)

Fig. 2.3 shows the STFT spectrum with a narrow window. With a larger window as shown in Fig. 2.4(a), the spectrum in Fig. 2.4(b) shows a higher frequency resolution compared to Fig. 2.3(c) and very low time resolution. Therefore, while Fig. 2.3 spectrum has more time information and less frequency information, Fig. 2.4 one has more frequency information and less time information.

This issue is similar to the Heisenberg Uncertainty Principle[11] which deals with the momentum and location of moving particles. For the time-frequency information of a signal, as with the Heisenberg Uncertainty Principle, both time and frequency information cannot be determined perfectly; which exact spectral component exists at which exact instance of time cannot be known. However, the time interval in which a certain band of frequencies exists[7] can be determined. So, if the time resolution is higher (i.e. narrower window function) like Fig. 2.3, the frequency resolution is lower and vice versa. A fixed narrow or wide window function provides two different types of results, not the complete picture. Therefore, the issue is choosing a specific window size and using it for the entire analysis.

In the mid-1970s, Jean Morlet faced this issue while working at an oil company analyzing acoustic echo signals which had very high frequencies with short durations and low-frequency components with long durations [12], [13]. STFT could be used to analyze only high frequencies with a narrow window or only low frequencies with a wide window, but not both. As

a solution, he experimented with the idea of varying the width of the window, i.e., scaling of the window function [14]. This work led to the concept of the wavelet transform, which will be discussed in the next section.

2.3 The Continuous Wavelet Transform (CWT)

2.3.1 Concept of multiresolution analysis (MRA)

Section 2.2 presented how in a short-term Fourier transform, if a better time resolution with a narrower window function is obtained, the frequency resolution will be worse, and vice versa. It is not possible to get both time and frequency resolution exactly or precisely; one of them needs to be sacrificed if the other is expected to be better. This issue exists regardless of the transform used [7].

In the short-term Fourier transform, a specific window function is selected which provides a fixed single time and frequency resolution for the entire signal. It can be any of the time-frequency resolution combinations, such as better time with worse frequency, worse time with better frequency, or mediocre time and frequency resolutions for the entire signal.

Instead of analyzing the entire signal with a fixed single time-frequency resolution set, as done in the STFT), the concept of multiresolution analysis suggests analyzing different frequency components or parts in different sets of time-frequency resolutions. Usually, high-frequency components are analyzed with a narrower window, i.e., a higher time resolution and lower frequency resolution while low-frequency components are analyzed with a wider window or lower time resolution, that is better frequency resolution. This approach is useful to extract more information from the signal especially since many practical signals [7], such as images have higher frequency components for short durations or instances and lower frequency components for long durations or instances.

In summary, in multiresolution analysis, unlike in a fixed single resolution in the STFT, different frequency components are analyzed in different time-frequency resolution sets, which makes it more useful for extracting necessary information from the signals.

MRA was employed by Cosserat [15] in the early 1900s, whose work incorporated couple stresses and local rotation of points [16]. In 1986, Mallat first introduced the concept of multiresolution analysis for the wavelet transform [12].

2.3.2 The continuous wavelet transform

The fixed resolution issue of the short-term Fourier transform (STFT) was the main reason for developing the wavelet transform [12]. Unlike the STFT where the window size is fixed, the wavelet transform enables variable window sizes (i.e., changes in the width of the window) for analyzing different frequency components in a signal [9] and provides varying time-frequency resolutions for analyzing a signal.

The window of varying width is called a "wavelet" because that the function of the wavelet is a short-duration finite energy function [17] and is oscillatory. All varying wavelets, represented by window functions, are generated by the translation and scaling of a fixed window function called the mother wavelet.

In the continuous wavelet transform (CWT), a mother wavelet function $\psi(t)$ has two properties or conditions: it is a function with a zero mean and it is normalized [9] i.e., it has unit energy [17]. It is expressed mathematically as

$$\int_{-\infty}^{+\infty} \psi(t)\,dt = 0 \quad \text{and} \quad \int_{-\infty}^{+\infty} |\psi(t)|^2\,dt = 1. \tag{2.5}$$

The mother wavelet function $\psi(t)$ can be of various types depending on the type of wavelet transform. For example, Fig. 2.5 shows the mother wavelet for a Mexican hat wavelet where $\psi(t) = (1-x^2)e^{-x^2/2}$.

Varying window functions (i.e., wavelets) are realized by scaling, namely dilation and contraction, of the mother wavelet $\psi(t)$ by s and shifting or translating it across the time axis by τ. So, the wavelets can be described as

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right). \tag{2.6}$$

As introduced in [18], the continuous wavelet transform of a signal *f*(*t*) can be expressed as

$$W(s,\tau) = \int_{-\infty}^{+\infty} f(t)\psi^*_{s,\tau}(t)dt = \frac{1}{\sqrt{s}}\int_{-\infty}^{+\infty} f(t)\psi^*(\frac{t-\tau}{s})dt\,, \qquad (2.7)$$

where τ is the shifting or translation parameter and corresponds to the time, and s is the scaling parameter and corresponds to the inverse of the frequency (i.e., a higher frequency means a lower scale, and vice versa).

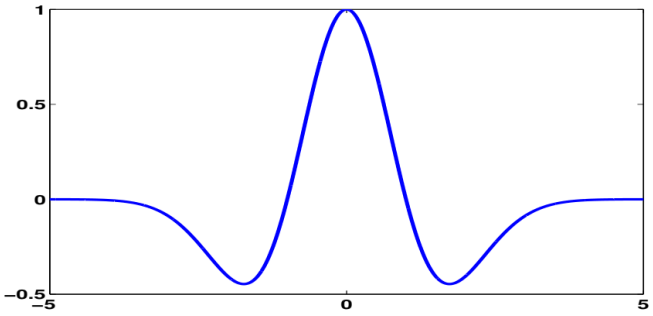


Fig. 2.5 Mexican hat wavelet [9]

The wavelet transform can also be expressed as the inner product of the input signal $f(t)$ and mother wavelet function $\psi(t)$ [9]. Since $\psi \in \mathbf{L}^2$, $f \in \mathbf{L}^2$, where $\mathbf{L}^2$ space is the space of the square-integrable functions, i.e., $< x, x > < \infty$ if $x \in \mathbf{L}^2$. According to the properties of the square-integrable function, the inner product of $f(t)$ and $\psi(t)$ is given by

$$< f, \psi_{s,\tau} >= \int_{-\infty}^{+\infty} f(t)\psi^*_{s,\tau}(t)dt = W(s,\tau)\ .$$

The wavelet transform can also be rephrased as a convolution product:

If $\bar{\psi}_s(t) = \frac{1}{\sqrt{s}}\psi^*(\frac{-t}{s})$, then $\bar{\psi}_s(\tau - t) = \frac{1}{\sqrt{s}}\psi^*(\frac{t-\tau}{s})$.

From the definition of convolution,

$$(f * \bar{\psi}_s)(\tau) = \int_{-\infty}^{+\infty} f(t)\bar{\psi}_s(\tau - t)dt = \int_{-\infty}^{+\infty} f(t)\frac{1}{\sqrt{s}}\psi^*(\frac{t-\tau}{s})dt\,. \qquad (2.8)$$

Using the equation (2.7) in (2.8),

$$W(s,\tau) = (f * \bar{\psi}_s)(\tau)\,. \qquad (2.9)$$

Therefore, the convolution of the input $f(t)$ with $\bar{\psi}_s(t)$ can provide the transformed outputs.

2.3.3 The discretization of the continuous wavelet transform

In discretization of the CWT, the scaling and translation parameters (s, τ) are sampled [19]–[25]. This is done by using $s = 2^j$ and $\tau = k2^j$ in (2.6) and (2.7) while keeping time continuous. The wavelet function, as defined in [19], becomes

$$\psi_{j,k}(t) = 2^{-j/2}\psi(2^{-j}t - k) \quad , \quad j,k \in Z \, . \tag{2.10}$$

The sampled CWT coefficients, also known as wavelet series coefficients [19], are as follows [7]:

$$W_{j,k} = \int_{-\infty}^{+\infty} f(t)\psi_{j,k}^{*} dt \, . \tag{2.11}$$

2.4 The Discrete Wavelet Transform (DWT)

Discrete wavelet transforms are based on the concept of the continuous wavelet transform. In many cases, DWT coefficients can be sampled from CWT counterpart, as shown in equation (2.10), although it is not the only option [7]. The DWT can be studied independently of its continuous counterpart [17]. Unlike the wavelet series or sampled/discretized CWT, time and parameters, both scaling and translation, are discrete [19]. The DWT also was considered as a natural wavelet transform for discrete-time signals by several early authors [19], [26], [27]. The additional advantage with the DWT compared to the CWT is the significant reduction of redundancy and computation time.

In the CWT, the filter coefficients are calculated using the equation (2.11) or (2.7) which varies the size of the window by time-shifting and scaling to provide variable time-frequency resolution, i.e., multiresolution. However, with the DWT, multiresolution is achieved using a distinct approach called subband coding which requires a low-pass and high-pass filter.

2.4.1 Scaling function

Before examining subband coding, some discussion of the concept of scaling function introduced by Mallat [24] is required. In equation (2.10), the wavelets can not be shifted and scaled continuously, they can be shifted and scaled only in discrete steps. Every time the wavelet is stretched in the time domain by a factor of 2, according to Fourier theory, its bandwidth in the

frequency domain is halved [28] (Fig. 2.6). Therefore, an infinite number of scaling and shifting is required to cover the whole transform. However, instead of an infinite number of wavelets, the lower part of the spectrum can be covered by a low-pass filter known as a scaling function.

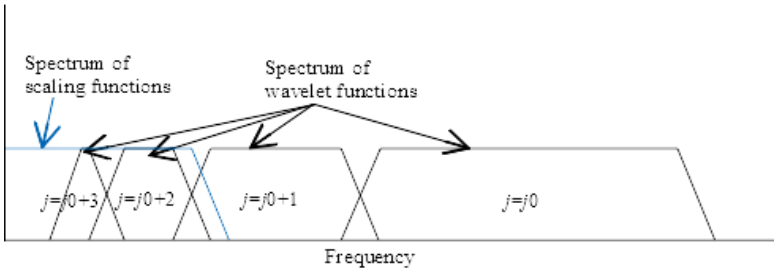


Fig. 2.6 Usage of scaling function instead of an infinite number of wavelets.

The scaling function can be formed by using the weighted version of the wavelet function in (2.10) in such a way that its spectrum fits the lower part of the spectrum in Fig. 2.6 which is left open by the wavelets. This can be expressed [28] by the following equation:

$$\phi(t) = \sum_{j,k} \gamma(j,k)\psi_{j,k}(t), \tag{2.12}$$

where $\gamma(j,k)$ is the necessary weight to cover the lower part of the spectrum.

2.4.2 Subband coding of the discrete-time signal

Subband coding is any type of transform that decomposes a discrete-time signal into different frequency bands. One of the common methods of subband coding that is adopted in the wavelet transform is to split the signal spectrum into two equal parts, a low-frequency and high-frequency part, through a low-pass and a high-pass filter [28] as shown in Fig. 2.7. Following this, the two types of processing are done.

a) The frequency band of each part is now half of the input signal. According to the Nyquist rule [29], the minimum sampling rate needed for each part is half of the sampling rate of the input signal. Thus, every other sample can be dropped. In other words, it can be subsampled by 2 as shown in Fig. 2.7.
b) The high-frequency part contains the smallest details, while the low-frequency part contains the average approximation of the signal with some small details. The low-

frequency part is passed through a high-pass and low-pass filter again after subsampling by 2.

These two steps complete the first level of decomposition. These steps are done iteratively until a target level is reached as shown in Fig. 2.7. It should be noted that subsampling in each step changes the scale resolution since removing samples decreases the resolution in time, which in turn means better resolution in frequency. As a result, with each level of decomposition, the lower frequency bands are filtered, and better frequency resolution is achieved by subband coding. In this way, MRA is realized in the discrete wavelet transform.

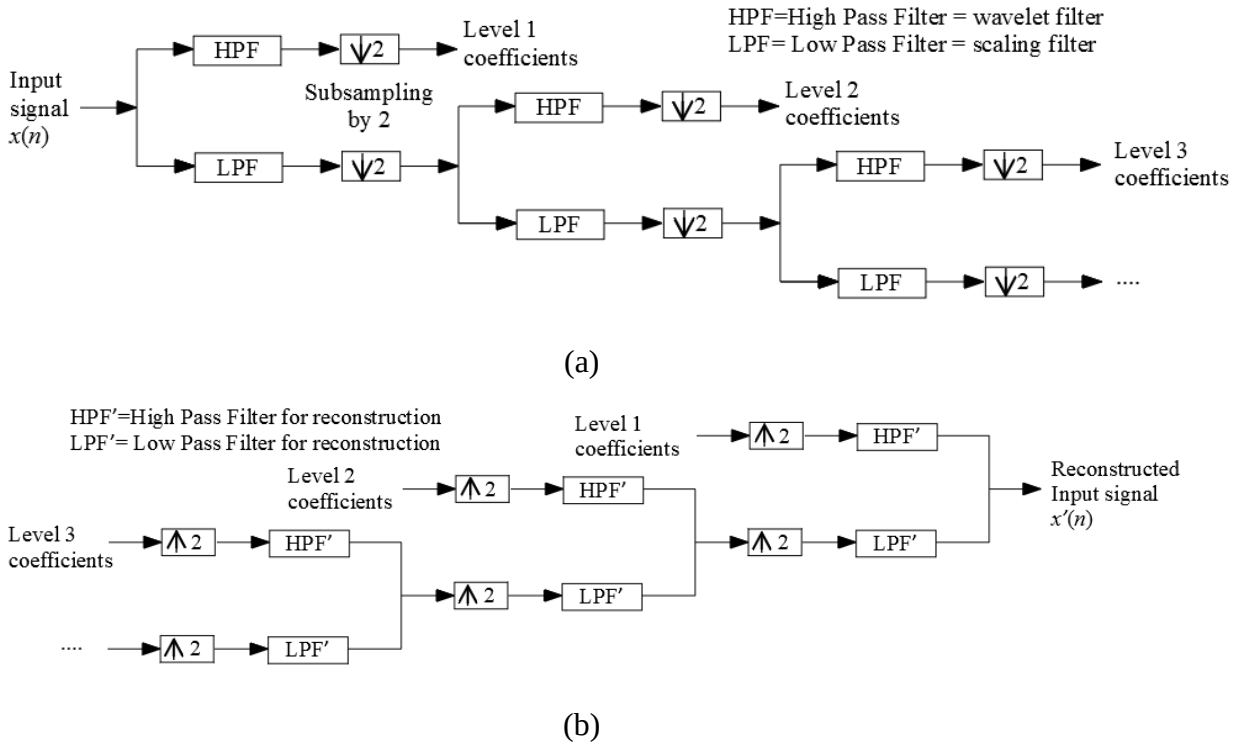


Fig. 2.7 Subband coding technique used in the DWT: (a) forward transform, (b) backward or inverse transform

2.4.3 The discrete wavelet transform

The discrete wavelet transform uses the subband coding technique shown in Fig. 2.7. The high-pass filter (HPF) is implemented by the wavelet function of the wavelet transform, while the low-pass one (LPF) is done by the scaling function. The HPF (i.e., wavelet filter) and LPH

(i.e., scaling filter) are denoted by $\tilde{g}$ and $\tilde{h}$, respectively, while the reconstruction HPF and LPH are denoted by g and h, respectively. The forward transform filters $\tilde{g}$ and $\tilde{h}$ are also known as analysis filters while inverse transform (reconstruction) filters g and h are known as synthesis filters. Fig. 2.8 shows the structure of the DWT including both forward and inverse transform with a single level of decomposition and reconstruction.

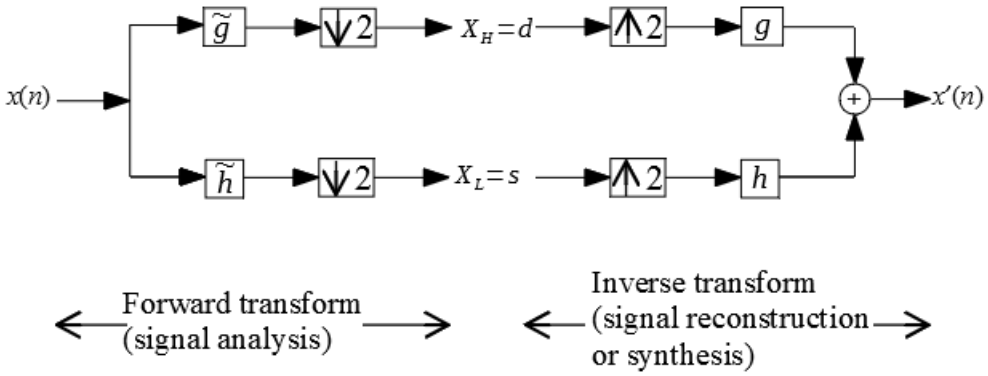


Fig. 2.8 Basic structure of the discrete wavelet transforms (single level of decomposition and reconstruction): forward transform uses the analysis filters $\tilde{g}$ and $\tilde{h}$, while the inverse transform uses the synthesis filter

Similar to the CWT in the equation (2.9), where the wavelet transform can be expressed as the convolution of the input and a function of the mother wavelet function, in the DWT, the output sequence can be expressed as the convolution of input and filters. The output sequence after the decomposition of the input sequence $x(n)$ is expressed as follows[7], [12]:

$$X_H(n) = \sum_k x(k)\tilde{g}(2n-k) = x * \tilde{g}(2n)$$
$$X_L(n) = \sum_k x(k)\tilde{h}(2n-k) = x * \tilde{h}(2n) \quad (2.13)$$

where

$\tilde{h}(N-1-k) = (-1)^n \tilde{g}(k)$ for the orthogonal wavelets.

Here, $x(n)$ is the input sequence, N is the filter length, and $X_H(n)$ and $X_L(n)$ are the high-pass and low-pass filter output sequences, respectively, after subsampling by 2; sometimes they

also are denoted by *d* and *s*, respectively, which refer to detail and approximations or the average component of the signal.

2.4.4 Constructing wavelets filters

To show how scaling and wavelet filters are derived, as an example, Daubechies orthogonal wavelets will be constructed.

For orthogonal wavelets, the relations among the filters $g(k)$ and $h(k)$ with the scaling and wavelet function are expressed as [17]

$$\phi(t) = \sum_{k=0}^{N-1} h(k)\sqrt{2}\phi(2t-k) = \sum_{k=0}^{N-1} c_k \phi(2t-k), \tag{2.14}$$

$$\psi(t) = \sum_{k=0}^{N-1} g(k)\sqrt{2}\phi(2t-k) = \sum_{k=0}^{N-1} c'_k \phi(2t-k). \tag{2.15}$$

Here, $h(k)$ and $g(k)$ are normalized coefficients, and $c_k = h(k)\sqrt{2}$ and $c'_k = g(k)\sqrt{2}$ are the un-normalized coefficients.

There are certain conditions required for the orthogonal wavelets to satisfy originality and other desirable properties for different kinds of applications. A summary of these conditions are (see [17] for details):

1. The scaling function should be normalized. This implies that the unit area under the scaling function must be 1. Therefore,

   $$\int \phi(t)dt = 1. \tag{2.16}$$

   From (2.14) and (2.16), we obtain

   $$\sum_{k=0}^{N-1} c_k = 2. \tag{2.17}$$

   For the Daubechies-8, N=8 and the value of k ranges from 0 to 7.

2. The integer translates of scaling function must be orthonormal. This implies that

   $$\int \phi(t)\phi(t-k)dt = \delta_{0,k}, \tag{2.18}$$

   where $\delta_{0,k} = \begin{cases} 1 if\ k=0 \\ 0 if\ k\neq 0 \end{cases}$.

   From (2.14) and (2.18), we obtain

$$\sum_{l=0}^{N-1} c_l^{\,2} = 2 \text{ for } \; k = 0, \tag{2.19}$$

$$\sum_{k=0}^{N-1} c_l c_{l-2k} = 0 \text{ for } \; k \neq 0. \tag{2.20}$$

3. An approximation of a signal using the scaling function is required in many applications. If the degree of smoothness is *p*, the smoothness or approximation conditions are expressed as

$$\sum_{k=0}^{N-1} (-1)^k c_k = 0 \text{ for } \; p=0, \tag{2.21}$$

$$\sum_{k=0}^{N-1} (-1)^k k^p c_k = 0 \text{ for } p>0 \tag{2.22}$$

Usage of different values of *p* (0, 1, 2 …) ensures we have enough number equations to solve the values of c_k. For example, for Daubechies-8 wavelet, *N*=8 and we need 8 unique equations to solve 8 c_k values. The condition 1, 2 and 3 with different values of *p* and *k* are able to provide a necessary number of equations.

4. The scaling and wavelet function are orthogonal. This implies that

$$\int \phi(t)\psi(t)dt = 0. \tag{2.23}$$

With the use of (2.14) and (2.15) in (2.21), it can be found that

$$c'_k = (-1)^k c_{N-k-1}. \tag{2.24}$$

This equation provides the relation between c_k and c'_k. This relation along with $h(k) = c_k / \sqrt{2}$ and $g(k) = c'_k / \sqrt{2}$ are sufficient to deduce the *h*(*k*) and *g*(*k*) filters from the values of c_k.

Solving the equations in the first three conditions provides the value of filter coefficients c_k and the corresponding scaling filter $h(k) = c_k / \sqrt{2}$. As an example, to find the filter of the Daubechies 8-tap wavelet transform, *N*=8 should be used in these relations. So, for Daubechies-8, using the equations (2.17), (2.19), (2.20), (2.21) and (2.22), we get the following equations.

$$
\begin{aligned}
&c_0+c_1+c_2+c_3+c_4+c_5+c_6+c_7=2\\
&c_0{}^2+c_1{}^2+c_2{}^2+c_3{}^2+c_4{}^2+c_5{}^2+c_6{}^2+c_7{}^2=2\\
&c_0c_2+c_1c_3+c_2c_4+c_3c_5+c_4c_6+c_5c_7=0\\
&c_0c_4+c_1c_5+c_2c_6+c_3c_7=0\\
&c_0c_6+c_1c_7=0\\
&c_0-c_1+c_2-c_3+c_4-c_5+c_6-c_7=0\\
&-c_1+2c_2-3c_3+4c_4-5c_5+6c_6-7c_7=0\\
&-c_1+4c_2-9c_3+16c_4-25c_5+36c_6-49c_7=0\\
&-c_1+2^3c_2-3^3c_3+4^3c_4-5^3c_5+6^3c_6-7^3c_7=0
\end{aligned}
\qquad (2.25)
$$

There are enough equations in the equation set (2.25) to solve 8 unknowns and find the value of the filter coefficients. After these equations are solved, the values of the coefficients of c_k and then the values of h_k can be determined by using the following relation:

$$h(k)=c_k/\sqrt{2}\,.$$

The derived values of h_k with a high degree of precision are shown in table 2.1.

The scaling filter (*h*-filter) in the Z-domain is:

$$h(z)=h(0)+h(1)z^{-1}+h(2)z^{-2}+h(3)z^{-3}+h(4)z^{-4}+h(5)z^{-5}+h(6)z^{-6}+h(7)z^{-7}. \qquad (2.26)$$

Since $g(k)=(-1)^k h(N-k-1)$, the wavelet filter would be

$$g(z)=-h(7)z^{6}+h(6)z^{5}-h(5)z^{4}+h(4)z^{3}-h(3)z^{2}+h(2)z^{1}-h(1)+h(0)z^{-1}. \qquad (2.27)$$

2.4.5 Common wavelets

There are many wavelet families such as Daubechies, Haar, Biorthogonal, Reverse Biorthogonal, Symlets, and Coiflets. The Daubechies and Biorthogonal wavelet families are the two most widely used.

Fig. 2.9 shows the scaling and wavelet function of Daubechies 8-tap wavelets [30] as an example, since the derivation of the *h*-filter of this wavelet has already been shown.

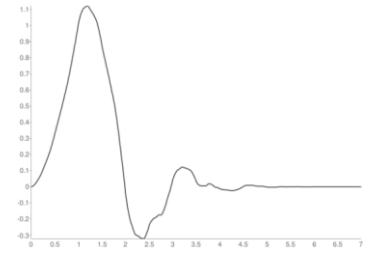
Daubechies wavelets are perfectly orthogonal and so the inverse wavelet transform is the adjoint of the forward transform. This family is based on Ingrid Daubechies's work on a systematic method to construct a compact support orthogonal wavelet [31]. It includes members ranging from the highly localized (D2, D4) to the highly smooth (D20) [32]. A higher tap or order wavelet filter offers comparatively better frequency localization and increased energy

compaction [33] along with increased implementation costs and more errors. As a result, the implementation of low-order filters, such as the Daubechies-4 (D4), D6, and D8, are more common in the literature [32], [34]–[39] while the implementation of higher order filters, like the D18 and D20, is very rare. For the Daubechies-N, the low-pass filter (h) and high-pass filter (g) are as follows:

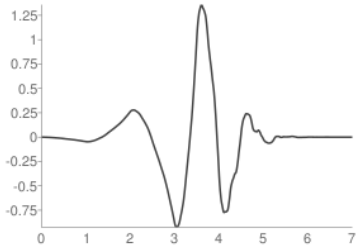
$$h(z) = h_0 + h_1 z^{-1} + \ldots + h_{N-1} z^{-(N-1)},$$

$$g(z) = -h_{N-1} z^{(N-2)} + \ldots - h_1 + h_0 z^{-1}.$$

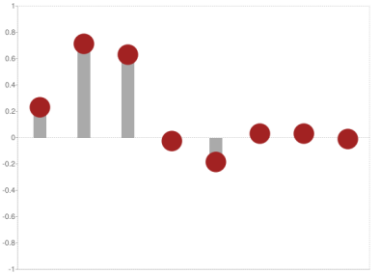
On the other hand, a biorthogonal wavelet is one where the transform may not be orthogonal but perfectly invertible. The two most popular biorthogonal wavelets are the Cohen-Daubechies-Feauveau wavelet 5/3 (CDF 5/3) and CDF 9/7, the latter of which has been adopted in the JPEG2000 compression standard. Table 2.2 shows the analysis filter coefficients of both the CDF 5/3 and CDF 9/7.



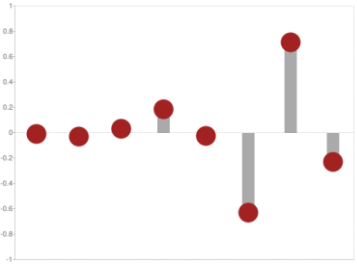
Scaling function



Wavelet function



h-filter coefficients (low-pass reconstruction)



g-filter coefficients (high-pass reconstruction)

Fig. 2.9 Daubechies 8-tap scaling function, wavelet function, and digital filters [30]

Table 2.1 Daubechies Wavelets filter coefficients

Daubechies-4		Daubechies-6		Daubechies-8	
h_0	$(1+\sqrt{3})/(4\sqrt{2})$	h_0	0.332670552950..	h_0	0.2303778133088960..
h_1	$(3+\sqrt{3})/(4\sqrt{2})$	h_1	0.806891509313...	h_1	0.7148465705529090..
h_2	$(3-\sqrt{3})/(4\sqrt{2})$	h_2	0.459877502119...	h_2	0.6308807679298580…
h_3	$(1-\sqrt{3})/(4\sqrt{2})$	h_3	-0.135011020010...	h_3	0.0279837694168599…
		h_4	-0.085441273882...	h_4	-0.1870348117190930…
		h_5	0.035226291882...	h_5	0.0308413818355607…
				h_6	0.0328830116668851..
				h_7	-0.0105974017850690…

Table 2.2 CDF-5/3 and 9/7 Analysis Filter Coefficients

	CDF 5/3		CDF 9/7	
K	Low-pass filter (h_k)	High-pass filter (g_k)	h_k	g_k
0	6/8	1	0.852698679008893	0.788485616405582
± 1	2/8	½	0.377402855612830	-0.4180922732216172
± 2	-1/8		-0.110624404418437	-0.0406894176091640
± 3			-0.023849465019556	0.06453888262869705
± 4			0.0378284555072640	-

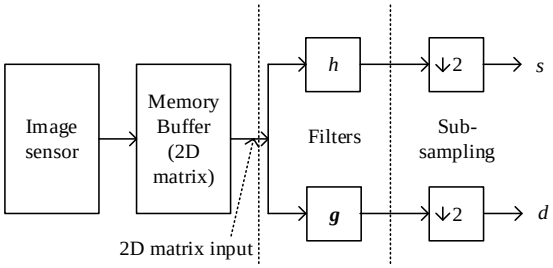
2.5 Implementation of the Discrete Wavelet Transform (DWT)

The decomposition and reconstruction of a wavelet filter can usually be implemented in two ways, by a filter bank or convolution or by the lifting method. Fig. 2.10 shows these methods and their differences in an image or two-dimensional signal processing application.

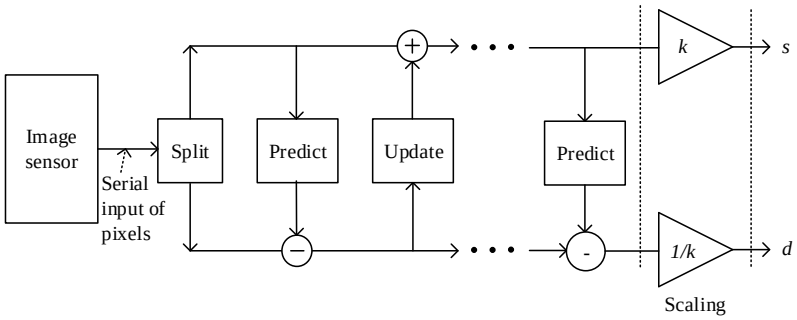
2.5.1 Standard filter bank vs Lifting steps-based implementations

The filter-bank- or convolution-based methods are the standard implementations which are the direct or indirect implementations of the convolution operation of the filters. For 2D signals, the methods are usually the 2D block-based technique (generally, 4x4, 8x8, or 16x16 blocks)

which usually implements matrix multiplication techniques or their variants. These methods usually require additional buffer memory, and hence, the hardware cost is often high. Moreover, the processing needed in the buffer memory affects the real-time processing of image sensor data adversely. Different variants of filter-bank-based implementations are proposed or explored in [40]–[46].



(a)



(b)

Fig. 2.10 Implementation of discrete wavelet transform (shown with image input source). (a) General filter-bank- or convolution-based approach and (b) lifting-based approach.

The lifting-based technique implements the same thing but without direct or indirect convolution or matrix multiplications. Instead, it is a ladder-like structure derived from the filters (see discussion in 2.5.2). As shown in Fig. 2.10, the lifting-based technique is progressive in nature using raster scanning and allows in-place implementation. Lifting schemes bring several benefits compared to the standard filter-bank implementation[47]:

a) The computational complexity decreases by about a half.

b) The speed of the transform increases.

c) It allows for an in-place implementation of the wavelet transform, and therefore,

does not require any auxiliary memory (as shown in Fig. 2.10).

d) It allows the building non-linear wavelet transform, such as wavelet transforms that maps integers-to-integers [48]. The integer-based transform is hardware-implementation friendly and allows lossless implementation of most steps in the lifting schemes.

The purpose of this research is to propose a lossless and low-cost hardware implementation of discrete wavelet transforms. To accomplish this goal and because of its advantages, a lifting-based technique will be used.

2.5.2 Conversion of filter-bank into lifting steps

Daubechies and Sweldon [47], [49] showed that a new structure of wavelet transform can be created from any orthogonal and biorthogonal filters. This new scheme is named the lifting scheme.

This new scheme begins with a well-known set of filters (h, g), and the filters are split into even and odd parts forming a polyphase matrix which can be expressed as

$$P(\mathrm{z}) = \begin{bmatrix} h_e & g_e \\ h_o & g_o \end{bmatrix}. \tag{2.28}$$

This polyphase matrix is then factorized using the successive division approach i.e., Euclidean algorithm of the greatest common divisor (GCD) and choosing the appropriate Laurent polynomials. The aim of the factorization is to represent the polyphase matrix as a set of upper and lower triangular matrices [35].

At first, the first column of the polyphase matrix is factorized which results in the following matrix decomposition:

$$\begin{bmatrix} h_e(z) \\ h_o(z) \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k \\ 0 \end{bmatrix}$$

Here, k is the greatest common divisors and $q_i(z)$ is the quotient in the $i-\text{th}$ step of the successive division approach of finding the greatest common-divisors.

A complementary polyphase filter g^0 can be found where another polyphase matrix $\hat{P}(z)$ is as follows:

$$\hat{P}(z)=\begin{bmatrix} h_e(z) & \hat{g}_e(z) \\ h_o(z) & \hat{g}_o(z) \end{bmatrix}=\prod_{i=1}^{n}\begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}$$

We can find a polynomial $S(z)$ such that $P(z)=\hat{P}(z)\begin{bmatrix} 1 & S(z) \\ 0 & 1 \end{bmatrix}$

So, the final factorization matrix is given as:

$$P(z)=\prod_{i=1}^{n/2}\begin{bmatrix} 1 & q_{2i-1}(z) \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ q_{2i}(z) & 1 \end{bmatrix}\begin{bmatrix} 1 & s(z) \\ 0 & 1 \end{bmatrix}\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}$$

where $s(z)=k^2S(z)$.

The above factorized polyphase matrix can be written in the following format:

$$P(z)=\prod_{i=1}^{m}\begin{bmatrix} 1 & s_i(z) \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ t_i(z) & 1 \end{bmatrix}\begin{bmatrix} k & 1 \\ 1 & 1/k \end{bmatrix}. \tag{2.29}$$

where k is a non-zero constant and the Laurent polynomials, $s_i(z)$ and $t_i(z)$ are the primal and dual lifting stages, respectively.

For example, for the Daubechies 4-tap wavelet transform (D4), shown in Table 2.1, using a successive division approach, the polyphase matrix for D4 wavelets is found to be

$$P(z)=\begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ b+cz^{-1} & 1 \end{bmatrix}\begin{bmatrix} 1 & z \\ 0 & 1 \end{bmatrix}\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}, \tag{2.30}$$

where $a=-\sqrt{3}$, $b=\sqrt{3}/4$, $c=(\sqrt{3}-2)/4$, $k=(\sqrt{3}+1)/\sqrt{2}$, and $1/k=(\sqrt{3}-1)/\sqrt{2}$.

For orthogonal wavelets, the dual polyphase matrix is the same as $P(z)$.

$$\tilde{P}(z)=P(z) \tag{2.31}$$

The analysis polyphase matrix should be

$$\tilde{P}(1/z)^T=\prod_{i=1}^{m}\begin{bmatrix} k & 1 \\ 1 & 1/k \end{bmatrix}\begin{bmatrix} 1 & t_i(z^{-1}) \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ s_i(z^{-1}) & 1 \end{bmatrix}. \tag{2.32}$$

The analysis polyphase matrix gives the direct implementation of the forward transform of the lifting scheme. This reduces the computational complexity when compared to the traditional

wavelet implementation.

If the input sequence to the transform is x, and the even and odd components are x_e and x_o, respectively, the outputs (low-pass component x_L and high-pass component x_H) are given by

$$\begin{bmatrix} X_L \\ X_H \end{bmatrix} = \tilde{P}(1/z)^T \begin{bmatrix} x_e \\ x_o \end{bmatrix}. \tag{2.33}$$

This corresponds to the following steps.

$$\begin{aligned}
&x_{o1} = s_1(z^{-1})x_e + x_0, \\
&x_{e1} = x_e + t_1(z^{-1})x_{o.1}, \\
&x_{o2} = s_2(z^{-1})x_{e1} + x_{01}, \\
&x_{e2} = x_{e1} + t_2(z^{-1})x_{o2}, \\
&\ldots \\
&\ldots \\
&X_L = s = kx_{em}, \\
&X_H = d = x_{om}/k.
\end{aligned} \tag{2.34}$$

The equation-set (2.34) can be used to deduce the structure of the lifting implementation of a forward wavelet transform which is shown in Fig. 2.11. The last step with the multiplier-pair (k,1/k) is known as the scaling step.

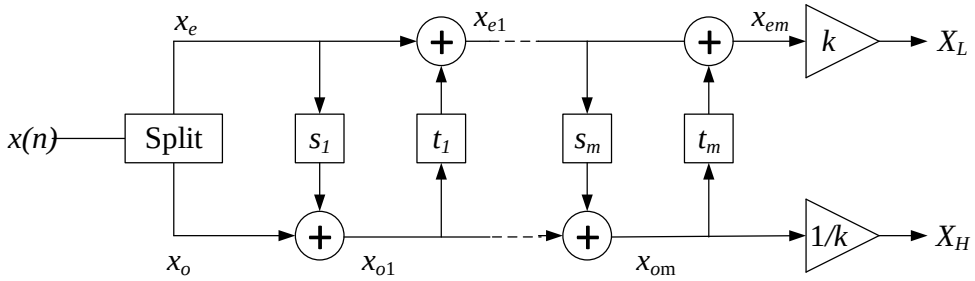


Fig. 2.11 Structure of lifting implementation of the forward part of the wavelet transform

The inverse transform can be found by the back-calculation of the equation set (2.34), shown in the equation set (2.35) and illustrated in Fig. 2.12.

$$
\begin{aligned}
&x_{em} = X_L / k, \\
&x_{om} = kX_H, \\
&\ldots \\
&\ldots \\
&x_{01} = x_{o2} - s_2(z^{-1})x_{e1}, \\
&x_{e1} = x_{e2} - t_2(z^{-1})x_{o2}, \\
&x_0 = x_{o1} - s_1(z^{-1})x_e, \\
&x_e = x_{e1} - t_1(z^{-1})x_{o.1}.
\end{aligned}
\tag{2.35}
$$

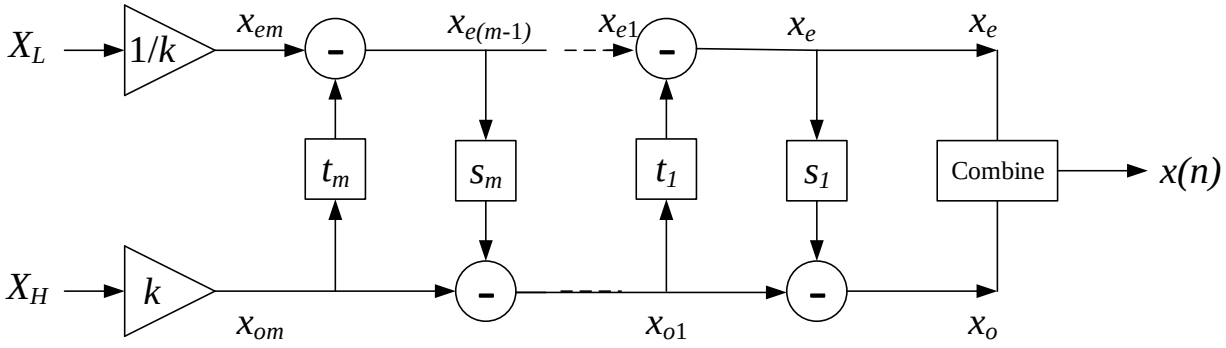


Fig. 2.12 Structure of lifting implementation of the inverse wavelet transform

This thesis aims to propose the efficient and lossless implementation of the lifting wavelet transform. Chapters 3, 4, and 5 discuss specific details of lifting implementation, conventional factorization techniques, related issues, and the proposed solution.

2.6 Existing Lifting-based Implementations

Daubechies and Sweldens introduced the lifting-step-based implementations in [47], [49]. These works show how the factorization of the polyphase matrix shown in (2.28) is performed to reach to (2.29). Many other lifting implementations such as [50]–[54] are based on this work while some other works such as [55]–[61] propose variant architectures of implementations.

2.6.1 The integer-based lifting wavelet transform and issues

Lifting-based implementations of [47], [49] can provide an excellent opportunity for lossless and lower cost hardware implementations as described in section 2.5. This opportunity is pursued in [48], [62], where Calderbank and Daubechies proposed another set of wavelet transforms based on lifting wavelets which map integer-to-integer.

In that work, the lifting step $x_{o1} = s_1(z^{-1})x_e + x_0$ is approximated by $x_{o1} = \lfloor s_1(z^{-1})x_e + 1/2 \rfloor + x_0$ to reach an integer value in each lifting step. This method can provide accurate reconstructions of the lifting step by using $x_0 = x_{o1} - \lfloor s_1(z^{-1})x_e + 1/2 \rfloor$. However, an issue remains with the scaling-steps sine if $X_L = s = \lfloor kx_{em} + 1/2 \rfloor$ is applied. x_{em} cannot be perfectly recovered with $x_{em} = \lfloor X_L / k + 1/2 \rfloor$. Therefore, the approach shown in [48] is not lossless when the scaling steps are present in a wavelet transform, and this is true for most orthogonal and many biorthogonal wavelet transforms.

The implementation algorithm based on this approach has the following problems:

(a) As explained above, the scaling steps are not reversible. The conventional lifting-based algorithm of most wavelets has scaling steps, which makes the transform irreversible. Therefore, the main wavelet-transform engine of many systems become lossy, and the data or signal cannot be reconstructed properly.

(b) Hardware implementation is costly since the values of the lifting-step filter coefficient s_i and t_i are irrational in many cases. For example, the polyphase matrix of the D4 contains irrational values like $-\sqrt{3}$, $\sqrt{3}/4$ (see equation (2.30)). Since here the irrational coefficients are multiplied, the cost is high. If these coefficients are truncated to a certain precision, the cost depends on the precision. But truncation leads to outputs which are different from the standard outputs. These changed outputs are often of higher entropy which indicates lower performance of the transform. Coefficients truncation should be based on the optimization of both performance and cost.

(c) Floating point operations in each step make the system expensive. Filter coefficients are not mapped into an integer in [48], [51], [62]. Though the truncated coefficients can be mapped the integers, adding $1/2$ can be the additional burden.

Most of the previous work regarding integer-based lifting wavelet uses the same or a variant of the algorithm proposed in [48], [62]. Some of these works are [63], [64], [65] , [66]. The work [63] constructed the integer-to-integer versions of several wavelets using the technique of [48] and showed that integer-to-integer and classic version often yield comparable quality at low bit rates. The research [64] also utilized the integer-based wavelet transform of [48] to present its usefulness in the distortionless data hiding. In [65], various integer wavelet transforms were presented based on [48] and they were utilized with zero-tree coding for thee-dimensional compression. Another work [66] also used the integer-based wavelet transform based on [48],[62] in steganography to utilize the perfect reconstruction ability in some wavelets.

2.6.2 Existing solution of the scaling-step issue

Regarding the scaling problems, one of the suggested solutions is converting the scaling steps into four additional lifting steps using either of the following formulas [47]:

$$\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} = \begin{bmatrix} 1 & k-k^2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1/k & 1 \end{bmatrix} \begin{bmatrix} 1 & k-1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \tag{2.36}$$

$$\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1-1/k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ k & 1 \end{bmatrix} \begin{bmatrix} 1 & 1/k^2-1/k \\ 0 & 1 \end{bmatrix} \tag{2.37}$$

There are some problems with this approach. First, the additional scaling steps need more hardware. Second, since the coefficients need to be truncated if they are mapped to integers in order to achieve low-cost implementation, truncation in each step incurs more errors. Therefore, this approach generally is not used.

2.6.3 Other integer-based lifting wavelet transform

In [67], the authors also used the conventional algorithm found in [48], [62] (see discussion 2.6.1), skipping the addition of $1/2$ in each lifting steps. Ding *et al*. [68] proposed an adaptive directional lifting for image coding which still maintains the scaling steps. Andra *et al*.

[51] proposed an integer-based lifting architecture, but it does not solve the scaling issues, and hence shows a lossy algorithm when the scaling is present. This algorithm converts filter coefficients into an integer by multiplication by 256, followed by rounding.

Recent work in the same area is [69], where Balster *et al*. proposed an integer-based lifting algorithm for CDF 9/7 wavelets. Mentioning the issues with the scaling steps, this work removes the scaling steps from the wavelet transform and include these steps in quantization; thus, the algorithm still remains lossy. In addition to the conventional algorithm in [48], [62], this algorithm converts the floating point coefficients into integers by multiplication by the power of two followed by a division and a floor operation.

To best of our knowledge, one of the previous works presents an efficient solution to reconstruction errors occurring in integer-based lifting architectures for most wavelets where scaling steps are present.

2.7 Conclusion

This chapter summarizes the background necessary to understand the objective of this research. A brief overview of the Fourier transform provides the basics of converting a time-domain signal to its frequency domain counterpart. Like many inventions, the Fourier transform has its limitation, such as incompatibility with non-stationary signals, which paved the way to further developments such as windowed or short-term Fourier transform. The windowed Fourier transform provides both frequency and time information but is limited by a fixed single resolution with a fixed window size. The need for multiresolution analysis was the driving force to find an alternative transform. The wavelet transform with a variable window-size was the solution.

Lifting-based method of the wavelet transform is an efficient alternative to the traditional filter-bank- or convolution-based implementation, decreasing the cost by almost half and creating an opportunity for efficient integer-based hardware implementation. However, there are a few issues, including non-invertible scaling steps, with an integer-based implementation which are discussed in this chapter.

2.8 Reference

[1] J. B. J. Fourier, "The Analytical Theory of Heat," *Cambridge Library Collection - Mathematics*. Cambridge University Press, Cambridge, 2009.

[2] J.-B. J. Fourier, *Théorie Analytique de la Chaleur (Analytical theory of heat)*. Paris: Chez Firmin Didot, père et fils, 1822.

[3] A. Graps, "An introduction to wavelets," *IEEE Comput. Sci. Eng.*, vol. 2, no. 2, pp. 50–61, 1995.

[4] M. DeCross, S. Philosophist, and J. Khim, "Fourier Series." [Online]. Available: https://brilliant.org/wiki/fourier-series/. [Accessed: 10-Jun-2018].

[5] B. Carter and R. Mancini, *Op Amps for everyone*. Newnes, 2017.

[6] "Non-Stationary Signals." [Online]. Available: http://www.azimadli.com/vibman/nonstationarysignals.htm. [Accessed: 11-Nov-2018].

[7] R. Polikar, "The wavelet tutorial," 1996. [Online]. Available: http://person.hst.aau.dk/enk/ST8/wavelet_tutotial.pdf. [Accessed: 29-Dec-2016].

[8] D. Gabor, "Theory of communication. Part 1: The analysis of information," *J. Inst. Electr. Eng. III Radio Commun. Eng.*, vol. 93, no. 26, pp. 429–441, 1946.

[9] S. Mallat, *A wavelet tour of signal processing*. Elsevier, 1999.

[10] "Sample Audio Files: Jet airplane." [Online]. Available: https://www.mathworks.com/help/audio/ug/sample-audio-files.html. [Accessed: 11-Nov-2018].

[11] L. Debnath and F. A. Shah, *Lecture Notes on Wavelet Transforms*. Springer, 2017.

[12] R. Polikar, "The story of wavelets," *Phys. Mod. Top. Mech. Electr. Eng.*, pp. 192–197, 1999.

[13] R. X. Gao and R. Yan, "From Fourier transform to wavelet transform: a historical perspective," in *Wavelets*, Springer, 2011, pp. 17–32.

[14] D. Mackenzie, "Wavelets: seeing the forest and the trees," Washingtion, DC, 2001.

[15] E. Cosserat and F. Cosserat, "Théorie des corps déformables," 1909.

[16] J. A. Moore, Y. Li, D. T. O'Connor, W. Stroberg, and W. K. Liu, "Advancements in

multiresolution analysis," *Int. J. Numer. Methods Eng.*, vol. 102, no. 3–4, pp. 784–807, Jan. 2015.

[17] K. P. Soman, *Insight into wavelets: From theory to practice*. PHI Learning Pvt. Ltd., 2010.

[18] P. Goupillaud, A. Grossmann, and J. Morlet, "Cycle-octave and related transforms in seismic signal analysis," *Geoexploration*, vol. 23, no. 1, pp. 85–102, 1984.

[19] O. Rioul and P. Duhamel, "Fast algorithms for discrete and continuous wavelet transforms," *IEEE Trans. Inf. theory*, vol. 38, no. 2, pp. 569–586, 1992.

[20] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Commun. pure Appl. Math.*, vol. 41, no. 7, pp. 909–996, 1988.

[21] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE Trans. Inf. theory*, vol. 36, no. 5, pp. 961–1005, 1990.

[22] C. E. Heil and D. F. Walnut, "Continuous and discrete wavelet transforms," *SIAM Rev.*, vol. 31, no. 4, pp. 628–666, 1989.

[23] S. Mallat and S. Zhong, "Signal characterization from multiscale edges," in *Pattern Recognition, 1990. Proceedings., 10th International Conference on*, 1990, vol. 1, pp. 891–896.

[24] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 674–693, 1989.

[25] M. Vetterli and C. Herley, "Wavelets and filter banks: Theory and design," *IEEE Trans. signal Process.*, vol. 40, no. 9, pp. 2207–2232, 1992.

[26] M. Vetterli and C. Herley, "Wavelets and filter banks: Relationships and new results," in *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, 1990, pp. 1723–1726.

[27] G. Evangelista, "Orthogonal wavelet transforms and filter banks," in *Multidimensional Signal Processing Workshop, 1989., Sixth*, 1989, p. 100.

[28] C. Valens, "A really friendly guide to wavelets," *ed. Clemens Val.*, 1999.

[29] D. Marshall, "Nyquist's Sampling Theorem," 2001. [Online]. Available: https://users.cs.cf.ac.uk/Dave.Marshall/Multimedia/node149.html. [Accessed: 31-Dec-2018].

[30] "Wavelet Daubechies 4 (db4)." [Online]. Available: http://wavelets.pybytes.com/wavelet/db4/ . [Accessed: 01-Jan-2019].

[31] L. Chun-Lin, "A tutorial of the wavelet transform," *Dept. Elect. Eng., Nat. Taiwan Univ., Taipei, Taiwan, Tech. Rep.*, 2010. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6942141.

[32] M. M. Hasan and K. A. Wahid, "Low-cost Architecture of Modified Daubechies Lifting Wavelets using Integer Polynomial Mapping," *IEEE Trans. Circuits Syst. II*, vol. PP, no. 89, pp. 1–2, 2016.

[33] M. K. Mandal, S. Panchanathan, and T. Aboulnasr, "Choice of wavelets for image compression," in *Information Theory and Applications II*, Springer, 1996, pp. 239–249.

[34] S. K. Madishetty, A. Madanayake, R. J. Cintra, S. Member, and V. S. Dimitrov, "Precise VLSI Architecture for AI Based 1-D / 2-D Daub-6 Wavelet Filter Banks With Low Adder-Count," *IEEE Trans. Circuits Syst. I*, pp. 1–10, 2014.

[35] P. Balakrishnan, M. M. Hasan, and K. A. Wahid, "An Efficient Algorithm for Daubechies Lifting Wavelets Using Algebraic Integers," *Can. J. Electr. Comput. Eng.*, vol. 37, no. 3, pp. 127–134, 2014.

[36] K. Wahid, V. Dimitrov, and G. Jullien, "VLSI architectures of Daubechies wavelet transforms using algebraic integers," *J. Circuits, Syst., Comput.*, vol. 13, no. 06, pp. 1251–1270, 2004.

[37] P. Sangeetha, M. Karthik, and T. KalavathiDevi, "VLSI architectures for the 4-tap and 6-tap 2-D Daubechies wavelet filters using pipelined direct mapping method," in *Proc. ICIIECS*, 2015, pp. 1–6.

[38] A. M. Al-Haj, "Fast discrete wavelet transformation using FPGAs and distributed arithmetic," *Int. J. Appl. Sci. Eng.*, vol. 1, no. 2, pp. 160–171, 2003.

[39] K. A. Wahid, M. A. Islam, and S.-B. Ko, "Lossless implementation of Daubechies 8-tap wavelet transform," *Proc. ISCAS*. pp. 2157–2160, 2011.

[40] G. Knowles, "VLSI architecture for the discrete wavelet transform," *Electron. Lett.*, vol. 26, no. 15, pp. 1184–1185, 1990.

[41] A. S. Lewis and G. Knowles, "VLSI architecture for 2D Daubechies wavelet transform without multipliers," *Electron. Lett.*, vol. 27, no. 2, pp. 171–173, 1991.

[42] K. K. Parhi and T. Nishitani, “VLSI architectures for discrete wavelet transforms,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 1, no. 2, pp. 191–202, 1993.

[43] M. Vishwanath, R. M. Owens, and M. J. Irwin, “VLSI architectures for the discrete wavelet transform,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 42, no. 5. pp. 305–316, 1995.

[44] A. Grzeszczak, M. K. Mandal, and S. Panchanathan, “VLSI implementation of discrete wavelet transform,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 4, no. 4, pp. 421–433, 1996.

[45] C. Yu and S.-J. Chen, “VLSI implementation of 2-D discrete wavelet transform for real-time video signal processing,” *IEEE Trans. Consum. Electron.*, vol. 43, no. 4, pp. 1270–1279, 1997.

[46] T. Acharya and P.-Y. Chen, “VLSI implementation of a DWT architecture,” in *ISCAS ’98. Proceedings of the 1998 IEEE International Symposium on Circuits and Systems (Cat. No.98CH36187)*, 1998, vol. 2, pp. 272–275 vol.2.

[47] I. Daubechies and W. Sweldens, “Factoring wavelet transforms into lifting steps,” *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 247–269, 1998.

[48] A. R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, “Wavelet Transforms That Map Integers to Integers,” *Appl. Comput. Harmon. Anal.*, vol. 5, no. 3, pp. 332–369, Jul. 1998.

[49] W. Sweldens, “The Lifting Scheme: A Construction of Second Generation Wavelets,” *SIAM J. Math. Anal.*, vol. 29, no. 2, pp. 511–546, Mar. 1998.

[50] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, “Efficient VLSI architectures of lifting-based discrete wavelet transform by systematic design method,” in *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, 2002, vol. 5, pp. V–V.

[51] K. Andra, C. Chakrabarti, and T. Acharya, “A VLSI architecture for lifting-based forward and inverse wavelet transform,” *IEEE Trans. Signal Process.*, vol. 50, no. 4, pp. 966–977, 2002.

[52] J. M. Jou, Y.-H. Shiau, and C.-C. Liu, “Efficient VLSI architectures for the biorthogonal wavelet transform by filter bank and lifting scheme,” in *Circuits and Systems, 2001. ISCAS 2001. The 2001 IEEE International Symposium on*, 2001, vol. 2, pp. 529–532.

[53] H. Olkkonen, J. T. Olkkonen, and P. Pesola, “Efficient lifting wavelet transform for microprocessor and VLSI applications,” *IEEE Signal Process. Lett.*, vol. 12, no. 2, pp. 120–122, 2005.

[54] X. Lan, N. Zheng, and Y. Liu, “Low-power and high-speed VLSI architecture for lifting-based forward and inverse wavelet transform,” *IEEE Trans. Consum. Electron.*, vol. 51, no. 2, pp. 379–385, 2005.

[55] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, “Flipping structure: An efficient VLSI architecture for lifting-based discrete wavelet transform,” *IEEE Trans. signal Process.*, vol. 52, no. 4, pp. 1080–1089, 2004.

[56] P.-Y. Chen, “VLSI implementation for one-dimensional multilevel lifting-based wavelet transform,” *IEEE Trans. Comput.*, vol. 53, no. 4, pp. 386–398, 2004.

[57] K. G. Oweiss, A. Mason, Y. Suhail, A. M. Kamboh, and K. E. Thomson, “A scalable wavelet transform VLSI architecture for real-time signal processing in high-density intra-cortical implants,” *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 54, no. 6, pp. 1266–1278, 2007.

[58] Y.-H. Seo and D.-W. Kim, “VLSI architecture of line-based lifting wavelet transform for motion JPEG2000,” *IEEE J. Solid-State Circuits*, vol. 42, no. 2, pp. 431–440, 2007.

[59] C. Cheng and K. K. Parhi, “High-speed VLSI implementation of 2-D discrete wavelet transform,” *IEEE Trans. signal Process.*, vol. 56, no. 1, pp. 393–403, 2008.

[60] X. Tian, L. Wu, Y.-H. Tan, and J.-W. Tian, “Efficient multi-input/multi-output VLSI architecture for two-dimensional lifting-based discrete wavelet transform,” *IEEE Trans. Comput.*, vol. 60, no. 8, pp. 1207–1211, 2011.

[61] W. Zhang, Z. Jiang, Z. Gao, and Y. Liu, “An Efficient VLSI Architecture for Lifting-Based Discrete Wavelet Transform,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, no. 3. pp. 158–162, 2012.

[62] A. R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. B.-L. Yeo, “Lossless Image Compresion Using Integer to Integer Wavelet Transforms.,” in *in Proc. ICIP*, 1997, pp. 596–599.

[63] M. D. Adams and F. Kossentni, “Reversible integer-to-integer wavelet transforms for image compression: performance evaluation and analysis,” *Image Process. IEEE Trans.*,

vol. 9, no. 6, pp. 1010–1024, 2000.

[64] G Xuan, J Zhu, J Chen, YQ Shi, Z Ni, and W Su, "Distortionless data hiding based on integer wavelet transform," *Electron. Lett.*, vol. 38, no. 25, pp. 1646–1648, 2002.

[65] A. Bilgin, G. Zweig, and M. W. Marcellin, "Three-dimensional image compression with integer wavelet transforms," *Appl. Opt.*, vol. 39, no. 11, pp. 1799–1814, 2000.

[66] M. F. Tolba, M. A. Ghonemy, I. A. Taha, and A. S. Khalifa, "Using integer wavelet transforms in colored image Steganography," *Int. J. Intell. Coop. Inf. Syst.*, vol. 4, no. 2, pp. 230–235, 2004.

[67] P. Chandrasekar and V. Kamaraj, "Integer lifting wavelet transform based hybrid active filter for power quality improvement," *2011 1st International Conference on Electrical Energy Systems*. pp. 103–107, 2011.

[68] W. Ding, F. Wu, X. Wu, S. Li, and H. Li, "Adaptive Directional Lifting-Based Wavelet Transform for Image Coding," *IEEE Transactions on Image Processing*, vol. 16, no. 2. pp. 416–427, 2007.

[69] E. J. Balster, B. T. Fortener, and W. F. Turri, "Integer Computation of Lossy JPEG2000 Compression," *IEEE Transactions on Image Processing*, vol. 20, no. 8. pp. 2386–2391, 2011.

Chapter 3

Low-cost Transform using Integer Polynomial Mapping

(case-study with Daubechies-4 and -6 wavelets)

Published as:

Md. Mehedi Hasan and K. A. Wahid, "Low-Cost Architecture of Modified Daubechies Lifting Wavelets Using Integer Polynomial Mapping," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 64, no. 5, pp. 585–589, 2017.

Implementation of discrete wavelet transforms in a low-cost and lossless/reversible manner is the primary objective of this thesis. As the first step of full-filling our objective, the options of low-cost implementations techniques were investigated. We explored how the discrete wavelet transform can be implemented in a hardware-friendly manner and applied the techniques on two popular wavelets, the Daubechies-4 (D4) and -6 (D6). Since achieving losslessness is another objective, though it is not achieved in this work, with careful selection of architecture, the proposed method achieves near-losslessness.

The previous chapter discussed the reasons that the lifting scheme is preferred over the traditional convolution- or matrix-based technique, namely, that cuts the cost by half and allows for making wavelets that map integer-to-integer which are hardware friendly and part of the lifting steps becomes lossless. For these reasons, in this chapter, the lifting based technique is selected for implementation. Then, integer-polynomial mappings for the filter polynomials, which result in a lower cost than usual floating-point based lifting transform, are applied. A low-cost and near-lossless algorithm with experiments on the D4 and D6 is achieved and described in this chapter. Based on the advantages of integer-based techniques learned in this chapter and related issues, chapter 4 presents the improvements to make the wavelet completely lossless.

Low-cost Architecture of Modified Daubechies Lifting Wavelets using Integer Polynomial Mapping[1]

Abstract

The paper proposes a modified version of the popular lifting algorithm of Daubechies-4 (D4) and Daubechies-6 (D6) wavelets and their efficient implementation using integer polynomial mapping (IPM). At first, an improved polyphase matrix for D4 is presented that eliminates one filter coefficient completely without losing any accuracy. Then IPM is applied to encode the remaining irrational coefficients. As a result, computation error due to irrational numbers in the conventional method is significantly reduced, resulting in better image reconstruction. For D6, a 2-level optimization scheme combined with resource sharing of coefficients is applied that results in simplified hardware architecture with much fewer resources.

Index Terms

Daubechies wavelet transform, lifting scheme, image compression, integer mapping.

---

[1]Manuscript received April 29, 2016; revised June 21, 2016; accepted June 21, 2016. Date of publication June 24, 2016; date of current version April 28, 2017. This work was supported in part by the Natural Science and Engineering Research Council of Canada, by the Canada Foundation for Innovation, and by the Western Economic Diversification Canada. This brief was recommended by Associate Editor J. M. de la Rosa.

3.1 Introduction

Image compression with high ratio is vital in the digital image and video transmission and storages. Among many image compression techniques in literature, Discrete Wavelet Transform (DWT) has been proven to be powerful due to its time-frequency characteristics [1][2]. Biorthogonal wavelets such as 5/3 and 9/7 are quite popular since they have been employed in JPEG 2000 compression standard [4]. Another wavelet family, called Daubechies wavelets, are also widely used because of its orthogonal nature [6]. Daubechies family includes members ranging from highly localized to highly smooth. Among all members of the family, Daubechies-4 (D4) and Daubechies-6 (D6) are the two most commonly found members in medical image compression and texture analysis due to their ability to achieve better compression with lesser complexity [4][5].

There are two ways a wavelet transform basis can be implemented: conventional matrix-based and lifting-based. The former is a block-based technique (generally, 4x4, 8x8 or 16x16 blocks) that requires additional buffer memory, and hence hardware cost is often high. Moreover, the need of buffer memory affects real-time processing of image sensor data adversely. The lifting based technique, on the other hand, is progressive in nature (i.e., raster scan) and, therefore, has half of computational complexity compared with the other [6]. While implementing lifting based Daubechies wavelets, truncation error accumulation inside the encoder takes place due to the lossy implementation of the irrational filter coefficients. The situation is worse at higher order decomposition level that affects the reconstruction quality of the image adversely.

This work addresses the issue in two steps. At first, we propose an improved version of the conventional lifting matrix that eliminates completely one irrational filter coefficient. Secondly, we apply a near-lossless mapping of the remaining coefficients using integer polynomial mapping (IPM). As a result, the hardware resource is greatly reduced. It also results in improved image reconstitution quality.

3.2 Conventional Lifting-Based Wavelets

The DWT is based on small wavelets with limited duration to analyze both frequency and time component of a signal. The conventional or classic method of implementing wavelets uses a well-known set of filters: *h* and *g*, where *h* is a low-pass and *g* is a high-pass filter. These filters are split into even and odd sequences. The polyphase matrix is assembled as:

$$P(\mathrm{z}) = \begin{bmatrix} h_e(\mathrm{z}) & g_e(z) \\ h_o(\mathrm{z}) & g_o(z) \end{bmatrix} \tag{3.1}$$

The matrix is factorized by using successive division approach and choosing appropriate Laurent polynomials. It can be expressed as shown below:

$$P(z) = \prod_{i=1}^{m} \begin{bmatrix} 1 & s_i(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_i(z) & 1 \end{bmatrix} \begin{bmatrix} K & 1 \\ 1 & 1/K \end{bmatrix} \tag{3.2}$$

where, *K* is a non-zero constant and the Laurent polynomials, $s_i(\mathrm{z})$ and $t_i(z)$, make up the primal and dual lifting stages respectively. This polyphase matrix, being the product of elementary matrices, can be implemented using lifting steps which are shown in the following sections.

Using a successive division approach, the polyphase matrix for D4 wavelets is found to be:

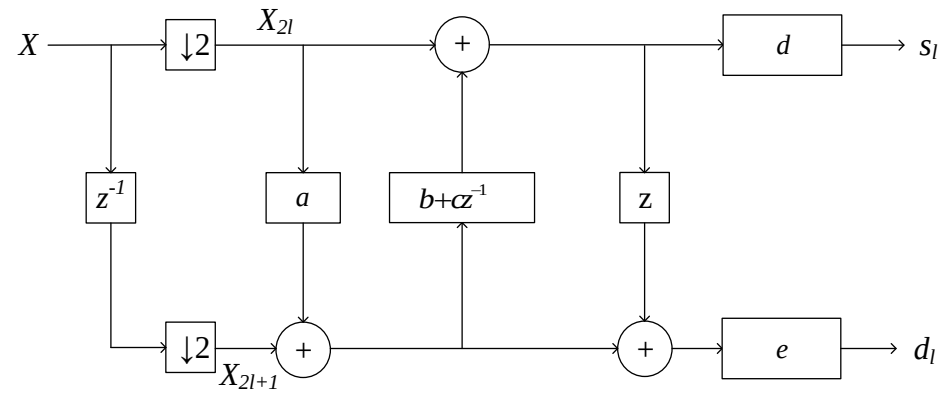
$$P'(z) = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ b + cz^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} d & 0 \\ 0 & e \end{bmatrix} \tag{3.3}$$

Where, $a = -\sqrt{3}$, $b = \sqrt{3}/4$, $c = (\sqrt{3}-2)/4$, $d = (\sqrt{3}+1)/\sqrt{2}$, and $e = (\sqrt{3}-1)/\sqrt{2}$.

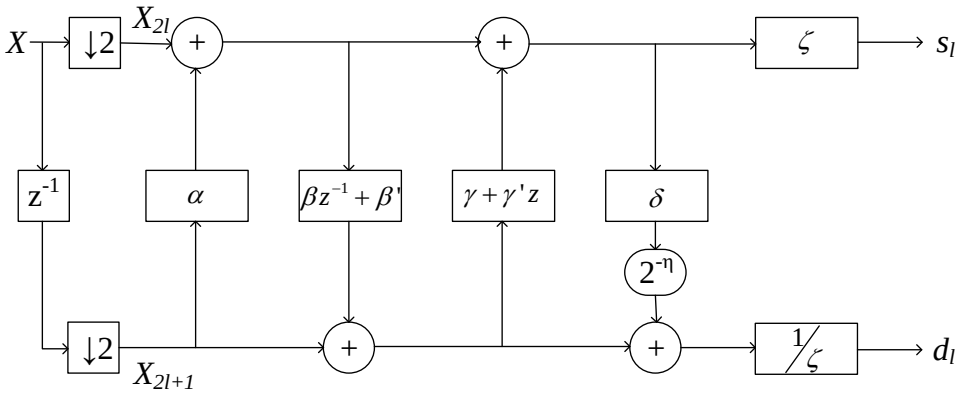
For D6 wavelets, similar division approach is used to find the factorized polyphase matrix:

$$P'(z) = \begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix} \begin{bmatrix} 1 & \beta z^{-1} + \beta' \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \gamma + \gamma' \mathrm{z} & 1 \end{bmatrix} \begin{bmatrix} 1 & \delta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \zeta & 0 \\ 0 & /\zeta \end{bmatrix} \tag{3.4}$$

where, $\alpha, \beta, \beta', \gamma, \gamma', \delta,$ and ζ are -0.4122865950, -1.5651362796, 0.3523876576, 0.0284590896, 0.4921518449, -0.3896203900, and 1.9182029462 respectively.



(a)



(b)

Fig. 3.1 Conventional lifting implementation of (a) D4 and (b) D6 wavelets

The conventional lifting schemes of D4 and D6 wavelets are shown in Fig. 3.1. Here, X is the serial input pixel which is subsampled by 2. It produces even component X_{2l} and odd component X_{2l+1}. Different factors of $P'(z)$ matrix are applied to compute the low-frequency component s_l and high-frequency component d_l. Now, it is seen that the entire coefficient set used in (3.3) and (3.4) are irrational in nature which not only requires a significant amount of hardware resources, but also introduces computation error during reconstruction. The aim of this work is to reduce the error and find efficient hardware architecture.

3.3 Proposed IPM-Based Lifting Algorithm

In this section, we present the IPM-based lifting algorithm of D4 wavelets followed by its extension to D6 wavelets.

3.3.1 IPM-based modified D4 wavelet

The proposed scheme is applied to D4 wavelets in two stages. At first, we carefully analyze the coefficients *b* and *c* along with the datapath structure. By using matrix decomposition and rearranging the datapath, a new polyphase matrix is constructed that eliminates one filter coefficient completely (i.e., *c* in Fig. 3.1(a)). The new matrix is below:

$$P'(z)=\begin{bmatrix}1 & a\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ b(1+z^{-1})-\frac{1}{2}z^{-1} & 1\end{bmatrix}\begin{bmatrix}1 & z\\ 0 & 1\end{bmatrix}\begin{bmatrix}d & 0\\ 0 & e\end{bmatrix} \tag{3.5}$$

The simplified diagram of the new filter structure is shown in Fig. 3.2. The elimination of one filter coefficient helps deal with one less multiplier that contributes heavily to accuracy improvement as well as complexity reduction.

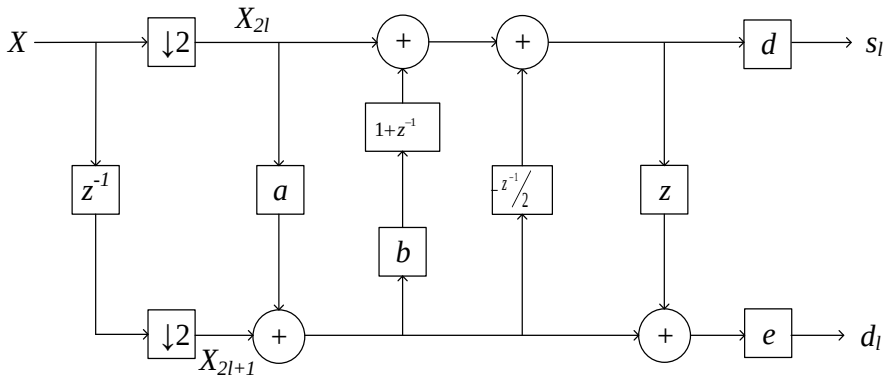


Fig. 3.2 Proposed lifting implementation of D4 (after stage 1)

In the second stage, we carefully apply a scaling function to both inputs and irrational coefficients. This scaling enables us to apply integer polynomial mapping (IPM) to encode the remaining coefficients; as a result, the irrational numbers are now mapped near-losslessly to integers which will effectively become the new filter basis.

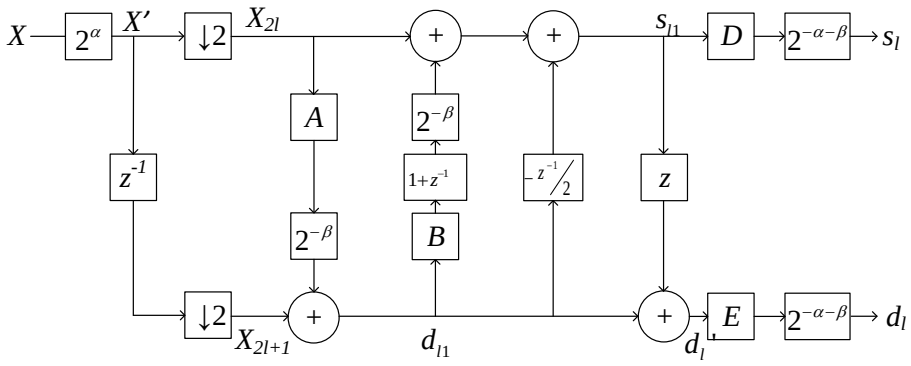


Fig. 3.3 Proposed lifting implementation of D4 (after stage 2)

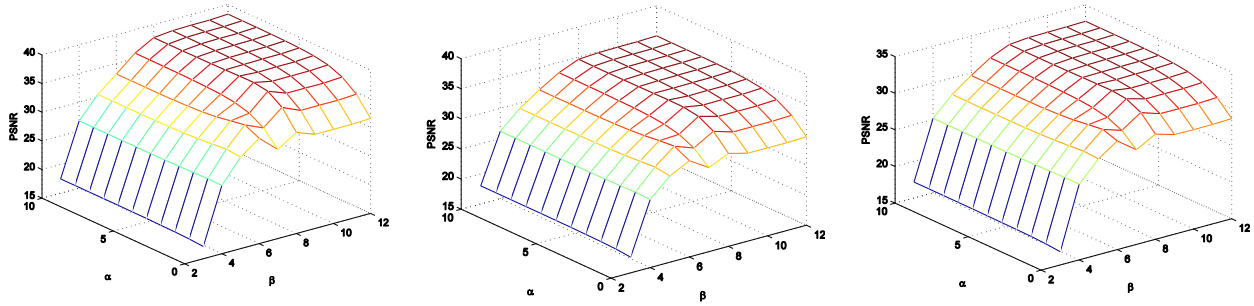
Let the scaling parameters for the input and the filter coefficients be 2^{α} and 2^{β} respectively. The value of α determines the minimum word length for the best representation of the input samples for optimum Peak Signal-to-Noise Ratio (PSNR) and β determines the best integer parameters of the irrational coefficients. Thus, the final polyphase matrix of D4 wavelets can be expressed as:

$$P'(z) = 2^{\alpha}\begin{bmatrix} 1 & A/2^{\beta} \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ B(1+z^{-1})/2^{\beta} - \frac{1}{2}z^{-1} & 1 \end{bmatrix}\begin{bmatrix} 1 & z \\ 0 & 1 \end{bmatrix}\begin{bmatrix} D/2^{\beta} & 0 \\ 0 & E/2^{\beta} \end{bmatrix}2^{-\alpha}$$

where, the new coefficients *A*, *B*, *D* and *E* are integers close to $-2^{\beta}\sqrt{3}$, $2^{\beta}\sqrt{3}/4$, $2^{\beta}(\sqrt{3}+1)/\sqrt{2}$, and $2^{\beta}(\sqrt{3}-1)/\sqrt{2}$ respectively. The filter structure after step 2 is given in Fig. 3.3.

The new parameters *A*, *B*, *D* and *E* are keys to efficient data reconstruction, and therefore computed after a careful observation and an exhaustive search of the scaling parameters α and β. In order to select the best values of these parameters, we simulated the entire algorithm using three benchmark images. The output image is evaluated using the PSNR index for different values of α and β for five scenarios with the compression ratio (CR) of 50%, 75%, 87.5%, 93.75% and 96.88%. The average PSNR indices are plotted in Fig. 3.4. It is noted that smaller values in α and β will ensure fewer hardware resources in implementation, and therefore will be desired.

As seen from Fig. 3.4, in all three images, the PSNR indices first increase as α and β increase; after a threshold, the indices hit the maximum and do not show any significant change. In order to limit the word length and hardware requirement, we consider the first maximum points as our desired point for implementation. Interestingly the first maximum point for all three benchmark images is the same, though the maximum PSNR indices vary for different images. From our analysis, the values chosen are: α = 3 and β = 8.



(a) Lena (b) Barbara (c) Mandrill

Fig. 3.4 Average PSNR indices at different bit rates are plotted against α and β values for three benchmark images

Table 3.1 New filter coefficients for D4 using IPM

Original Coeff.	Values	Integer coeff. after IPM	After applying CSD	Filter coefficients after simplification	
				Final values	Scaling
a	$-\sqrt{3}$	*A* = -444	$-2^9+2^6+2^2$	$-2^7+2^4+2^0$	$(\times 2^{-6})$
b	$\sqrt{3}/4$	*B* = 111	$2^7-2^4-2^0$	$2^7-2^4-2^0$	$\left(\times 2^{-8}\right)$
c	$(\sqrt{3}-2)/4$	Eliminated			
d	$(\sqrt{3}+1)/\sqrt{2}$	*D* = 495	2^9-2^4 -2^0	2^9-2^4 -2^0	$(\times 2^{-8})$
e	$(\sqrt{3}-1)/\sqrt{2}$	*E* = 132	2^7+2^2	2^5+2^0	$(\times 2^{-6})$

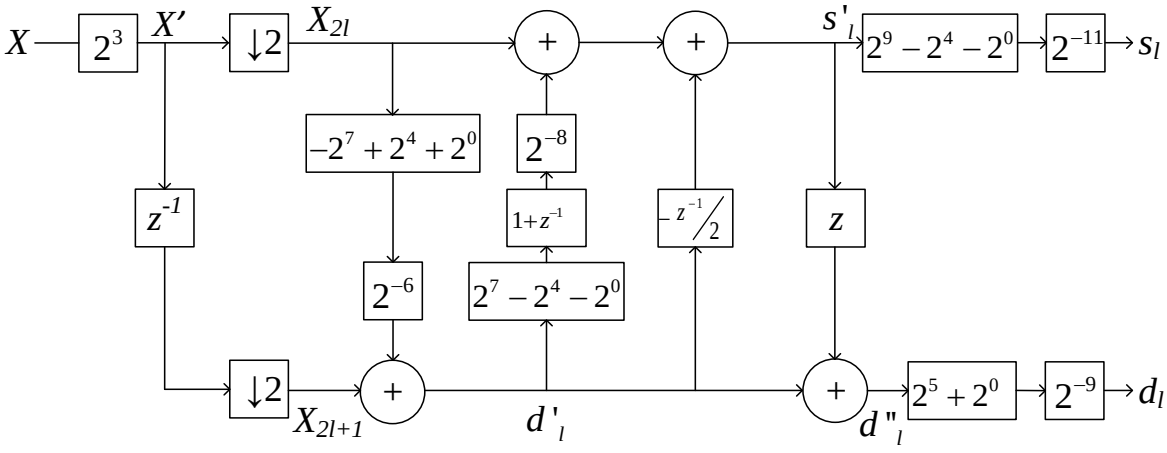


Fig. 3.5 Final IPM implementation of D4 wavelets

Table 3.1 shows the development of integer filter coefficients from the original irrational coefficients using the chosen β. It is seen that the use of IPM helps replace irrational numbers with integers which enables an error-reduced inverse operation. To improve the efficiency of hardware implementation, we further apply canonical signed digit (CSD) representation to eliminate all multipliers. Fig. 3.5 shows the final hardware architecture of D4 lifting wavelets.

3.3.2 IPM-based D6 wavelet

After successfully applying IPM to D4 wavelets, we extend it to D6 wavelets. This is also a 2-stage process. In the first stage, we directly apply IPM to encode the coefficients with optimum PSNR followed by CSD representation to eliminate multiplier. In stage 2, we analyze the bit pattern of the representation carefully and combine two coefficients to further save hardware cost. To do it, let the scaling parameters for input and filter coefficients be 2^{φ} and 2^{η} respectively. Then the new polyphase matrix can be expressed as:

$$P'(z) = 2^{\varphi}\begin{bmatrix} 1 & 0 \\ a/2^{\eta} & 1 \end{bmatrix}\begin{bmatrix} 1 & (bz^{-1}+b')/2^{\eta} \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ (c+c'z)/2^{\eta} & 1 \end{bmatrix}\begin{bmatrix} 1 & d/2^{\eta} \\ 0 & 1 \end{bmatrix}\begin{bmatrix} e/2^{\eta} & 0 \\ 0 & 2^{\eta}/\mathrm{e} \end{bmatrix}2^{-\varphi} \quad (3.6)$$

Here, the new coefficients are given as:

$a = \lfloor 2^{\eta}\alpha \rfloor$, $b = \lfloor 2^{\eta}\beta \rfloor$, $b' = \lfloor 2^{\eta}\beta' \rfloor$, $c = \lfloor 2^{\eta}\gamma \rfloor$, $c' = \lfloor 2^{\eta}\gamma' \rfloor$,
$d = \lfloor 2^{\eta}\delta \rfloor$, and $e = \lfloor 2^{\eta}\zeta \rfloor$

The polyphase matrix can now be implemented as shown in Fig. 3.6. The optimal values of parameters φ and η are chosen following similar exhaustive analysis (as described before for D4), i.e. maximization of PSNR index with minimization of φ and η values. The chosen values are φ = 0 and η = 6.

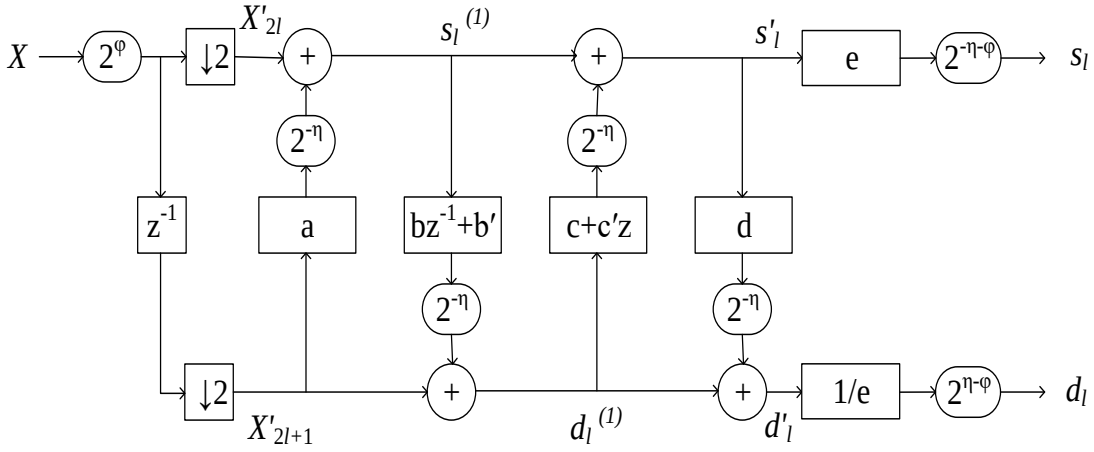


Fig. 3.6 Proposed lifting implementation of D6 (after stage 1)

Table 3.2 shows how integer coefficients are developed for D6 using the combination of IPM and CSD representation. The new coefficients are found to be: a = -26, b = -100, b' = 24, c = 2, c' = 31, d= -25, and e = 123. It is followed by a division by 2^6. Therefore, like D4, the application of integer mapping helps replace irrational coefficients with integers enabling an error-reduced forward and reverse operation.

In the 2nd stage, we analyze the bit pattern of each coefficient and then perform coefficient sharing as much as possible. For example, from Table 3.2, $b = -(2^2 b' + 2^2)$. So, the multiplier ($bz^{-1} + b'$) in Fig. 3.6 can now be expressed as $-(2^4 + 2^3)(2^2 z^{-1} - 1) - 2^2 z^{-1}$ which requires three adders to implement instead of four (if no sharing): $-(2^6 + 2^5 + 2^2)z^{-1} + (2^4 + 2^3)$. Finally, the new filter structure of D6 is formed and shown in Fig. 3.7.

Table 3.2 Development of new coefficients for D6 wavelets using IPM and CSD Representation

Original filter coefficients	Coefficient values	Integer coefficients after IPM	Integer values $\left(\times 2^{-\eta}\right)$	Integer values in CSD representation $\left(\times 2^{-\eta}\right)$	Filter coefficients after simplification	
					Final values	Scaling
α	–0.4122865950	a	-26	$-(2^4+2^3+2^1)$	$-(2^3+2^2+2^0)$	$(\times 2^{-5})$
β	–1.5651362796	b	-100	$-(2^6+2^5+2^2)$	$-(2^4+2^3+2^0)$	$(\times 2^{-4})$
β'	0.3523876576	b'	24	2^4+2^3	2^4+2^3	$(\times 2^{-6})$
γ	0.0284590896	c	2	2^1	2^1	$(\times 2^{-6})$
γ'	0.4921518449	c'	31	2^5-2^0	2^5-2^0	$(\times 2^{-6})$
δ	–0.3896203900	d	-25	$-(2^4+2^3+2^0)$	$-(2^4+2^3+2^0)$	$(\times 2^{-6})$
ζ	1.9182029462	e	123	$2^7-2^2-2^0$	$2^7-2^2-2^0$	$(\times 2^{-6})$

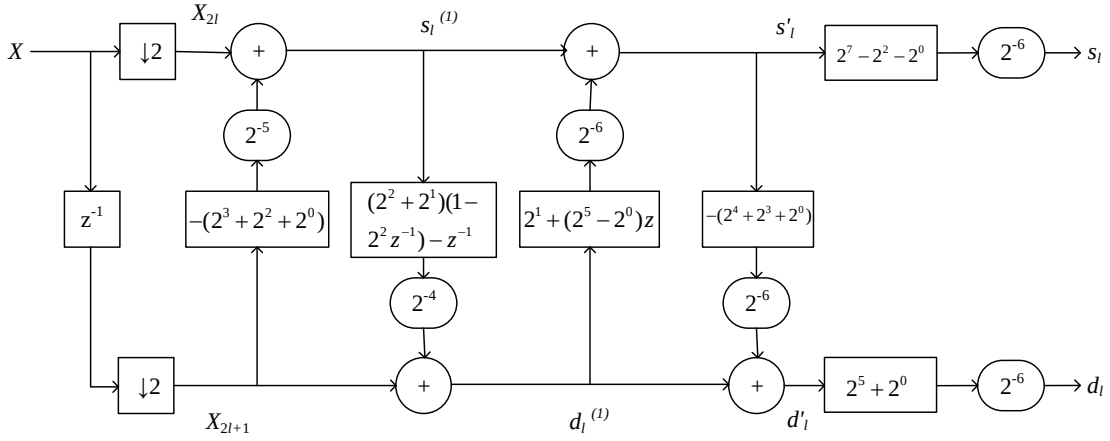


Fig. 3.7 Final IPM implementation of D6 wavelets

Overall, the new filter costs 12 adders and 11 shift registers for D4, and 16 adders and 18 shift registers for D6. No multipliers are required. Thus, the integer mapping of the irrational coefficients combined with coefficient sharing result in a decrease in hardware resources.

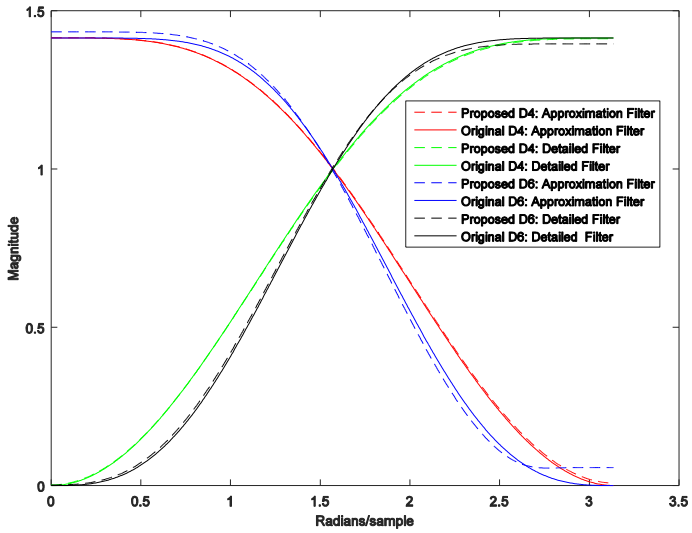


Fig. 3.8 Spectrum response of D4 and D6 (precision: 8-bits in proposed; 64-bits in original)

In our implementation, both D4 and D6 filters have been designed for 8-bit precision. To show the impact of precision loss on filter operation, we present the spectrum response of both approximation and detailed outputs in Fig. 3.8 and compare with 64-bit original filters. It is seen that, even with only 8-bit precision, the proposed filters follow the original filters very closely, and thus the influence of precision loss is minimal.

The proposed IPM technique is general enough to be applied to biorthogonal wavelets such as, 5/3 and 9/7 wavelets that have lifting steps. In that case, one needs to follow the decomposition steps as described above (similar to Figs 3.2 and 3.3), find optimal parameters (similar to Fig. 3.4), and then apply resource sharing or coefficient elimination depending on the newly found lifting coefficients (similar to Fig. 3.5).

3.4 Hardware Validation and Cost Assessment

For performance evaluation, we have coded our algorithm in Verilog HDL and implemented the filters physically into Cyclone II field-programmable gate array (FPGA) using a technique called *FPGA-in-the-loop*; this provides more accurate results than simulation-based implementation. Here, Simulink was used to send the input image pixels to the FPGA module.

Three standard images were taken as input and then compressed using the proposed schemes. The compressed output from the module is then read in Matlab and reconstructed back. Our design is later synthesized using 65nm CMOS technology using Synopsys Design Compiler.

Table 3.3 shows the summary of our implementation and compares it with other integer mapping architectures in the literature. Since, the works reported here used Xilinx FPGA, we too showed our results for Xilinx (Virtex-6 family) for a fair comparison. It is interesting to see that the application of hardware sharing in 2-D D6 actually results in a lower cost than that of 2-D D4. It is because of the proposed IPM technique that makes the sharing possible for D6. It is also seen from the table that the present work offers a significant advantage in hardware implementation compared with others (both matrix and lifting based schemes).

In [9], Wahid *et al.* presented a matrix based D4 and D6 hardware where the cost of logic cells is 248 for D4 and 680 for D6. In [10], a lifting-based 9/7 design was proposed that employs shared datapath but costs higher number of multipliers. In [12], Madishetty *et al.* presented a 2-D implementation of D4 and D6. Though the frequency of operation is high, hardware cost is high too for both wavelets. The authors later improved their work in [13] by applying approximation techniques to reduce the cost.

In [8], Balakrishnan *et al.* presented a lifting-based implementation of both D4 and D6 (1-D only) which showed considerable improvement over matrix based works. However, the proposed IPM scheme has lesser hardware cost and higher image quality than [8] since it eliminates one coefficient and also employs hardware sharing. As a result, the current design decreases the logic cells by 21.33% for 1-D D4 and 13.68% for 1-D D6.

In Table 3.3, we also report the image reconstruction quality in terms of PSNR. Here, we make best efforts to keep the settings (i.e., the number of decomposition level, type of coding, etc.) similar. Therefore, from the table, when the image quality is compared with other methods, along with the cost of implementation, the IPM scheme is found to be an efficient alternative in image coding application.

Table 3.3 Performance Comparisons with existing Integer based Daub Architectures

	Wahid [9]	Hongyu [10]	Gustafsson [11]	Madishetty [12]	Madishetty [13]	Balakrishnan [8]	IPM D4	IPM D6
Wavelet	D4 / D6	9/7	D6	D4 / D6	D6	D4 / D6	D4	D6
Architecture	1-D	1-D / 2-D	1-D	2-D	1-D / 2-D	1-D	1-D / 2-D	1-D / 2-D
Matrix/ Lifting	Matrix	Lifting	Matrix	Matrix	Matrix	Lifting (regular)	Lifting (modified)	Lifting (shared arch)
HW sharing	No	Yes	No	No	No	No	Yes	Yes
Multiplier	0	6 / 12	0	0	0	2 / 4	0	0
Adders	10 / 18	-	18	10 / 21	17 / 18	4 / 6	12	16
Logic cells (FPGA)	248 / 680	818 / 1758	-	426 / 1040	248 / 867	211 / 190	166 / 432	164 / 403
Cell count(VLSI)	3934 / 1005	-	-	-	- / 1040	-	384 / 907	327 / 795
Register bits	200 / 494	-	-	258 / 765	177 / 593	-	105 / 250	79 / 203
Frequency (MHz)	148 / 119	50 / 50	-	282 / 146	-	-	144 / 143	143 / 142
PSNR (dB)	38 / 39	74.85**	-	54.64/ 57.12	71.54*	32.55 / 33.19	50.83	64.77

*PSNR for 1-level 2-D approximation block only **Using 16-bit multiplier

Table 3.4 shows the resource consumption for different bit lengths in Xilinx Virtex-6 FPGA and CMOS 65nm technology. The 2-D results do not consider reuse of any logic cells of the 1-D filter. It is seen from Figs. 3.5 and 3.7 that 16 and 5 additional intermediate bits are required, due to the operations performed in the lifting stages, to retain the full precision for D4

and D6 respectively. As such, the memory requirement for 2-D transpose operation is $(b_i+16)N$ bits for D4 and $(b_i+5)N$ bits for D6, where b_i is the input word length and N is the size of the image or array to be transformed.

Table 3.4 Resource Consumption for IPM based Daub Architecture

Scheme and Technology		Resources	Input word lengths					
			8 bits		12 bits		16 bits	
			1-D	2-D	1-D	2-D	1-D	2-D
D4	FPGA	LUTs	166	432	214	488	262	575
		Freq (MHz)	144.3	143.7	143.3	143.8	143.3	143.3
		Mem (bits)	-	24N	-	28N	-	32N
	VLSI	Cells	384	907	496	1131	608	1357
		Power (μW)	14.46	34.72	19.10	43.99	23.74	53.29
D6	FPGA	LUTs	164	403	224	523	284	643
		Freq (MHz)	143.7	142.3	142.2	140.3	140.7	138.9
		Mem (bits)	-	13N	-	17N	-	21N
	VLSI	Cells	327	797	443	1029	559	1261
		Power (μW)	11.65	29.72	16.81	40.99	21.97	50.38

In order to show the performance for image reconstruction, the algorithms have been verified using several benchmark images. Here, we calculate the PSNR after 3-levels of 2-D decomposition followed by image reconstruction. No quantization was performed. As a result, the error produced on the output image is only caused by the truncation error during computation. Table 3.5 shows the results and compares them with that of the classical lifting algorithm with 8-b and 16-b fixed point coefficients. We also compare the results with that of Daubechies integer D4 scheme (shown as Int D4 [3]) for 8-b precision [3]. It is seen that, for all

three benchmark images, the proposed technique produces much higher PSNR index (or in other words, much lower reconstruction error) compared with classical floating point (FP) and Int D4 techniques.

In Table 3.6, we present the compression performance for two images at different compression ratio (CR). In this experiment, we applied 4-level decomposition followed by set partitioning in hierarchical trees (SPIHT) encoding [7] for all cases. The results show that the IPM scheme (using 8-bit) has comparable PSNR index with the classical D4 and D6 schemes (using 64-bit). In addition, the IPM schemes perform much better compared with other integer based lifting wavelets at different CRs, except for 5/3 wavelet that works better than IPM D6 and classical D6 at lower CR. The cost of our scheme in terms of adders is also lesser than Int D4 and (9,7) wavelets. Therefore, it is seen from Tables 3.5 and 3.6 that the proposed methods work well for both cases of 'compression' and 'without compression'.

Table 3.5 Assessment of Image Quality at Different Precision (PSNR in Decibels)

Schemes (precision)	Lena	Barbara	Mandrill	Average
IPM D4 (8 bits)	50.83	50.93	50.96	50.91
Classical D4 (8 bits)	40.51	41.23	40.55	40.76
Classical D4 (16 bits)	47.09	47.31	47.61	47.34
Int D4 (8 bits) [3]	36.39	36.14	36.13	36.32
IPM D6 (8 bits)	64.77	64.25	65.30	64.77
Classical D6 (8 bits)	48.61	48.76	48.77	48.71
Classical D6 (16 bits)	52.45	52.59	52.64	52.56

3.5 Conclusion

The paper presents a simplified decomposition algorithm of D4 and D6 wavelets and its efficient implementation. The use of integer polynomial mapping helps eliminate one core filter

coefficient in D4 and enable coefficient sharing in D6 that results in significant reduction in error accumulation during the computation process. The cost of implementation is also reduced, making it suitable candidate in image coding.

Table 3.6 Assessment of image quality at different CRs

Schemes	PSNR in dB at different CRs						Cost* (adders)
	Lena			Barbara			
	8:1	4:1	2:1	8:1	4:1	2:1	
IPM D4	37.9	42.1	47.0	34.0	40.5	45.9	12
IPM D6	32.4	34.2	35.0	27.0	32.2	32.9	15
Classical D4	37.9	42.2	47.5	34.1	40.6	47.1	--
Classical D6	32.4	34.4	35.2	28.7	32.2	33.0	--
Int D4 [3]	22.6	30.5	36.2	21.2	28.2	35.3	22
5/3 [4]	26.0	35.0	43.7	23.3	33.4	42.3	5
9/7 [3]	24.7	35.4	40.2	23.0	32.3	38.9	31

*considering 8 bit coefficient

3.6 References

[1] M. Vetterli and C. Herley, “Wavelets and filter banks: theory and design”, *IEEE Trans. Signal Pro.*, vol. 40, no. 9, pp. 2207-2232, 1992.

[2] CH Hsia, JM Guo, and JS Chiang, "Improved Low-Complexity Algorithm for 2-D Integer Lifting-Based Discrete Wavelet Transform Using Symmetric Mask-Based Scheme", *IEEE TCSVT*, 19, no.8, pp. 1202 – 1208, 2009.

[3] A. Calderbank, I. Daubechies, W Sweldens, and Boon-Lock Yeo, “Wavelet Transforms That Map Integers to Integers,” Appl. Comp. Harmon. Anal., 5(3), 332–369, 1998.

[4] M. Rabbani and R. Joshi, “An overview of the JPEG 2000 still image compression standard,” Signal Process. Image Commun., vol. 17, no. 1, pp. 3–48, 2002.

[5] B.K. Mohanty and A Mahajan, “Area and power efficient architecture for high-throughput implementation of lifting 2-d DWT,” IEEE Trans. Circuits. Syst. II, Exp. Briefs, vol. 59, no. 7, pp. 434–438, Jul. 2012.

[6] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps", *Journal of Fourier Analysis and Applications*", vol. 4, no. 3, pp. 245-267, 1998.

[7] A. Said and W. Pearlman, "A new, fast and efficient image codec based on set partitioning", *IEEE TCASVT*, vol.6, no.3, pp. 243-250, 1996.

[8] P. Balakrishnan, M. Hasan, and K. Wahid, “An Efficient Algorithm for Daubechies Lifting Wavelets Using Algebraic Integers,” *CJECE*, vol. 37, no. 3. pp. 127–134, 2014.

[9] K. Wahid, V. Dimitrov and G. Jullien, "VLSI architectures of Daubechies wavelet transforms using algebraic integers ", *J. Circuits Systems and Computer*, vol. 13, no. 6, pp.1251–1270, 2004.

[10] H. Liao, M.K. Mandal, and B.F. Cockburn, “Efficient architectures for 1-D and 2-D lifting-based wavelet transforms,” *IEEE Trans. Sig. Pro.*, 52(5), 1315–1326, 2004.

[11] S. Athar and O. Gustafsson, “Optimization of aiq representations for low complexity wavelet transforms,” in Proc. 20th ECCTD, 2011, 314–317.

[12] S. K. Madishetty, A. Madanayake, R. J. Cintra, V. S. Dimitrov, and D. H. Mugler, "VLSI Architectures for the 4-Tap and 6-Tap 2-D Daubechies Wavelet Filters Using Algebraic Integers," IEEE TCAS-I, vol.60, no.6, pp.1455,1468, June 2013.

[13] S. K. Madishetty, A. Madanayake, R. J. Cintra, and V. S. Dimitrov, "Precise VLSI Architecture for AI Based 1-D/ 2-D Daub-6 Wavelet Filter Banks With Low Adder-Count," IEEE TCAS-I, vol.61, no.7, pp.1984,1993, July 2014.

Chapter 4

Lossless Low-cost Implementation by Scaling Elimination

(case-study with Daubechies-8 wavelet)

Published as:

M. M. Hasan and K. A. Wahid, “Low-Cost Lifting Architecture and Lossless Implementation of Daubechies-8 Wavelets,” *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 65, no. 8, pp. 2515–2523, 2018.

Implementation of discrete wavelets transform in a low-cost and lossless/reversible manner is the primary objective of this thesis. As the first step to full-fill our objectives, in chapter 3, a low-cost and near-lossless wavelet implementation technique was proposed. In this chapter, a lossless and low-cost implementation is developed and applied on Daubechies-8 wavelet.

After outlining the primary reasons, for this approach, a proposed solution backed by a mathematical proof outlined in section 4.3.3 is presented. Application on usually costly Daubechies-8 wavelets transform results in lossless and low-cost implementation. Though in this chapter, only D8 is used, the technique can be utilized in other wavelets as well.

Low-cost Lifting Architecture and Lossless Implementation of Daubechies-8 Wavelets[2]

Abstract

This paper presents three lifting structures of Daubechies-8 (known as D8) wavelet transform based on the efficient factorizations of the polyphase matrix. All coefficients of the newly formed filters are mapped with integers that enable efficient hardware implementation. We first derive efficient polyphase matrices using the factorization algorithm, which form several lifting structures of D8. A theory is derived and experimentally proven to eliminate the scaling stage that incurs computation error in all integer-based wavelets. Because of the elimination of the scaling stage, the architectures become lossless and due to the optimum integer mapping, our results show that an 8-bit implementation of the proposed schemes perform very closely with the classical D8 filters with double-precision. Finally, we compare our schemes with classic D8 and other existing methods to demonstrate the advantage of our schemes in terms of lower cost, losslessness and higher performance.

Index Terms

Wavelet transform, Daubechies wavelets, lifting algorithm, lossless implementation.

[2]Manuscript received August 21, 2017; revised December 11, 2017; accepted January 17, 2018. Date of publication February 12, 2018; date of current version July 3, 2018. This work was supported in part by the Natural Science and Engineering Research Council of Canada, in part by the Canada Foundation for Innovation, and in part by the Western Economic Diversification Canada. This paper was recommended by Associate Editor G. Jovanovic Dolecek.

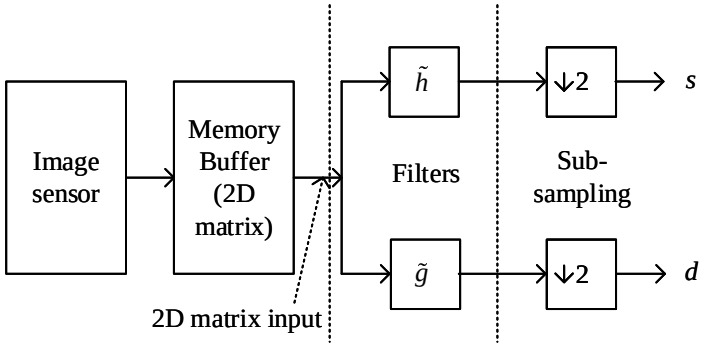
4.1 Introduction

When it comes to analyzing real-time signals and data which are aperiodic, irregular, noisy and transient, Discrete Wavelet Transform (DWT) is particularly useful because of its capability to explore signals concurrently in both frequency and time domain. Due to these powerful characteristics, DWT is used widely in science, mathematics, computer science, medicine, finance and engineering applications [1][2]. It is also adopted in JPEG2000 image compression standard[3]. A popular wavelet family is Daubechies wavelets which are based on a systematical architecture to construct the compact support orthogonal wavelet[4]. Orthogonal Daubechies wavelets are also computationally superior to biorthogonal wavelets such as, CDF 9/7 [5] [6]. Daubechies wavelet family includes members ranging from highly localized (e.g., 4-tap, 6-tap) to highly smooth (e.g., 20-tap) [7]. A higher tap or order wavelet filter offers comparatively better frequency localization and increased energy compaction, wavelet regularity, and transform coding gain (G_T) compared to low order ones[8]; this increment is important in many applications such as image compression since this results in better compression.

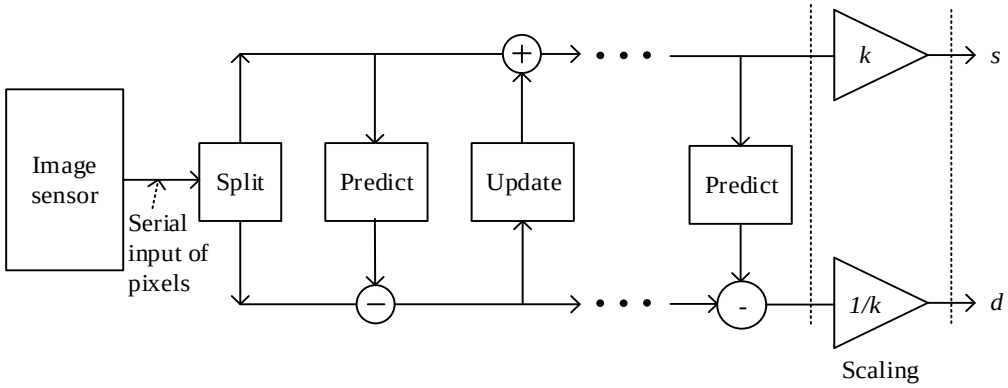
However, the advantage of orthogonal Daubechies wavelets comes with increased complexity, which increases the implementation cost and incurs more error in computation. This is why implementation of low-order filters (such as, Daubechies 4-tap or D4 and Daubechies 6-tap or D6 are more common in the literature [5], [7], [9]–[11], while the higher-order filters (such as, Daubechies 8-tap or D8 and higher) are very rare. It is accepted that D8 filters will have advantages over D4 and D6 in terms of computational performance and reconstruction accuracy; therefore its implementation was attempted in several works which are costly [12][13] simply because the higher order filter coefficients are highly complex in nature and less friendly for implementation.; they often suffer from low time localization and crucial edge information[8].

A survey in [14] shows that D8 gives the best performance in image compression among different wavelets in terms of statistical measures. D8 also provides the most correct results among Daubechies wavelets for classification in mammography [15][16]. Another study in [17] shows that D8 provides the best noise removal from the raw EEG signal of healthy patients.

Moreover, in multicarrier multipath wireless transmission, D8 decreases more inter-symbol interference power compared to D4 and D6 [18]. Also, D8 filter based steganographic algorithm achieves the best security compared to other filters[19]. Other use of D8 includes noise denoising in multivariate statistical modeling[20], data smoothing[21], image indexing and searching[22], analysis of engineering surface texture[23], fault detection[24], classification [25] and high-speed distance protection[26] in power systems, fingerprint recognition[27], shorting spike/action potentials in neurophysiology[28], compression of radar image [29], block compressed sensing of natural images[30]. Due to the inherent advantages and wide ranges of application, we aim to develop an efficient implementation of D8 which is free from the shortcomings of a higher order filter and consume comparable resources like D6, but yet is completely lossless in nature.



(a)



(b)

Fig. 4.1 Implementation of Daubechies wavelet transform. (a) General matrix based approach [2, 9, 10, 31] with memory buffer [7] and (b) lifting based approach [31, 32] that shows the advantages over matrix based approach in real-time image processing applications

There are two ways that a wavelet transform basis can be implemented: (a) conventional filter bank or matrix based and (b) lifting based as shown in Fig. 4.1. The former is a 2D block-based technique (generally, 4x4, 8x8 or 16x16 blocks are used) that requires additional buffer memory, and hence hardware cost is often high. Moreover, the need of buffer memory affects real-time processing of image sensor data adversely. The lifting based technique, on the other hand, is progressive scan in nature. It also has half of computational complexity than the other [31] which motivates us to use lifting-based structure.

The conventional methods of implementing a wavelet transform uses a well-known set of wavelet filters for that wavelet: *h* and *g*, where *h* is a low-pass filter and *g* is a high-pass filter; these filters are used to find the approximation coefficients (*s*) and detailed coefficients (*d*) respectively as shown in Fig. 4.1(a). In conventional way of implementing lifting-based wavelet [31] as shown in Fig. 4.1(b), this h and g filter is converted into lifting steps such as predict and update steps. In lifting method, the *h* and *g* filters are split into even and odd sequences which are subscripted by *e* and *o* respectively. Then the polyphase matrix is formed as:

$$P(\mathrm{z}) = \tilde{P}(z) = \begin{bmatrix} h_e(\mathrm{z}) & g_e(z) \\ h_o(\mathrm{z}) & g_o(z) \end{bmatrix} \tag{4.1}$$

Here $h_e(z)$ and $h_o(z)$ are Laurent polynomials. Now we need to find the greatest common divisor (GCD) of $h_e(z)$ and $h_o(z)$. Using the Euclidean algorithm for Laurent polynomials, successive division approach is employed that results in the following matrix decomposition (where *k* is the GCD):

$$\begin{bmatrix} h_e(z) \\ h_o(z) \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k \\ 0 \end{bmatrix} \tag{4.2}$$

Now, let another polyphase matrix $\hat{P}(z)$ be

$$\hat{P}(z) = \begin{bmatrix} h_e(\mathrm{z}) & \hat{g}_e(z) \\ h_o(\mathrm{z}) & \hat{g}_o(z) \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} \tag{4.3}$$

A polynomial $S(z)$ can be found such that

$$P(z) = \hat{P}(z) \begin{bmatrix} 1 & S(z) \\ 0 & 1 \end{bmatrix} \tag{4.4}$$

The final factorization matrix is given as:

$$P(z)=\prod_{i=1}^{n/2}\begin{bmatrix}1 & q_{2i-1}(z)\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ q_{2i}(z) & 1\end{bmatrix}\begin{bmatrix}1 & s(z)\\ 0 & 1\end{bmatrix}\begin{bmatrix}k & 0\\ 0 & 1/k\end{bmatrix} \tag{4.5}$$

where $s(z)=k^2S(z)$.

The lifting steps in this factorized polyphase matrix is used to implement the transform. Using this technique, the work in [31] derives the lifting steps for the Daubechies orthogonal wavelets such as D4 and D6.

Since most of the coefficients in the factorized polyphase matrix are irrational, the hardware implementation is expensive. The work in [32] proposes a technique to construct integer version of the lifting wavelet transform which makes many lifting wavelets (where the value of k is 1) lossless; this method proposes rounding of the multiplication operation in each lifting steps. However, the integer version of the orthogonal Daubechies wavelets, where the value of $k \neq 1$, does not become completely lossless, except for Haar wavelet. Because of the final scaling step ($\begin{bmatrix}k & 0\\ 0 & 1/k\end{bmatrix}$), the integer implementation of those wavelet schemes (with scaling) remain lossy [31]. Although they can be considered near-lossless in low-precision lower-order filters like D4 or double precision higher-order filters like D8, for the applications where completely lossless reconstruction is not essential; however, it is still a concern in low-cost low-precision implementation of higher-order filters like D8. Other works on the lifting based wavelet transform include [33]-[37], [7]; they use integer architecture based on the approaches proposed in [32] which results in lossy or near-lossless reconstructions for D4 and D6. The work in [7] proposes an integer mapping of the coefficients to generate a low-cost architecture of D4/D6 where the output still remains partly lossy in nature.

Although higher order filters like D8 have additional advantages in performance, as stated in our previous discussion, the hardware implementation of these filters is rare due to higher cost of implementation. Also in lifting implementation with finite precision, due to more lifting steps, the transform outputs deviate from the ideal output for the same precision (which can affect the quality of transform) compared to lower-order filter. Besides, in the integer version of a lifting implementation, the scaling steps incur additional computation error in reconstruction, since $k \neq 1$

in conventional lifting techniques of D8. To the best of our knowledge, none of the previous work attempted to implement lifting steps of D8 using integer mapping.

Therefore, the objectives of this work are to solve those issues of higher order filters and propose an implementation for a higher order yet useful filter (D8) which is lossless in reconstruction and maintains the quality of transform. Thus, we propose and prove that removal of scaling steps, which ensures lossless reconstruction and does not affect transform coding gain at all. Another important quality metric, entropy can be made close to the performance of the classic double-precision implementation by careful deduction of polyphase matrices. Therefore, in this work we attempt to: (a) find several polyphase matrices and corresponding lifting steps for D8 wavelets and identify the best basis matrix for high quality reconstruction; (b) make it lossless and reduce the cost by completely eliminating the scaling steps; this elimination is justified by both theoretical analysis and extensive experiments; (c) lastly, apply integer mapping on the basis coefficients to further reduce the hardware cost.

4.2 Daubechies 8-tap Filter (D8)

Let us first discuss the classic filter for Daubechies 8-tap wavelet (D8) and the implementation filters.

For Daubechies wavelets, the scaling function $\varphi(t)$ and wavelet function $\psi(t)$ are related by the following equations:

$$\varphi(t) = \sum_{k=0}^{N-1} h(k)\sqrt{2}\varphi(2t-k) = \sum_{k=0}^{N-1} c_k \varphi(2t-k) \tag{4.6}$$

$$\psi(t) = \sum_{k=0}^{N-1} g(k)\sqrt{2}\varphi(2t-k) = \sum_{k=0}^{N-1} c'_k \varphi(2t-k) \tag{4.7}$$

Here, $h(k)$ and $g(k)$ are normalized coefficients; $c_k = h(k)\sqrt{2}$ are the un-normalized coefficients. If we satisfy the conditions of the Daubechies wavelet such as normalization of the scaling function, orthogonality of both scaling function, wavelet function and integer translate of scaling and smoothness function, we get the coefficients $h(k)$ of the filter. For D8, the low-pass filter (h) and high-pass filter (g) are as follows:

$$h(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + h_3 z^{-3} + h_4 z^{-4} + h_5 z^{-5} + h_6 z^{-6} + h_7 z^{-7}$$
$$g(z) = -h_7 z^{6} + h_6 z^{5} - h_5 z^{4} + h_4 z^{3} - h_3 z^{2} + h_2 z^{1} - h_1 + h_0 z^{-1} \quad (4.8)$$

The coefficients are h_0 = 0.2303778133088960…, h_1 = 0.7148465705529090…, h_2 = 0.6308807679298580…, h_3 = -0.0279837694168599…, h_4 = -0.1870348117190930…, h_5 = 0.0308413818355607…, h_6 = 0.0328830116668851… and h_7 = -0.0105974017850690….

The above filter coefficients are irrational in nature, and usually implemented by taking a double precision (64 bits) floating point (FP) in MATLAB for lossless computation. However, implementing them in hardware will cost significantly large resources.

4.3 Proposed D8 Schemes

As seen in the previous section, the GCD of Laurent polynomials is not unique; lifting implementation is therefore not unique. Careful selection of quotient at the time of successive division results in different efficient lifting scheme. Now, we will employ this approach and present three new variants of Daubechies 8-tap wavelets that are low-cost and enables lossless hardware implementation.

4.3.1 Lifting schemes for D8

Using the steps described in Section 4.1, let us first convert the conventional D8 filter set (h, g), as given by eqn (4.1) into lifting steps. The polyphase matrix is assembled as:

$$P(z) = \begin{bmatrix} h_0 + h_2 z^{-1} + h_4 z^{-2} + h_6 z^{-3} & -h_7 z^{3} - h_5 z^{2} - h_3 z^{1} - h_1 \\ h_1 + h_3 z^{-1} + h_5 z^{-2} + h_7 z^{-3} & h_6 z^{3} + h_4 z^{2} + h_2 z^{1} + h_0 \end{bmatrix}. \quad (4.9)$$

So, the even and odd components of the $h(z)$ filter are:

$h_e(z) = h_0 + h_2 z^{-1} + h_4 z^{-2} + h_6 z^{-3}$ and $h_o(z) = h_1 + h_3 z^{-1} + h_5 z^{-2} + h_7 z^{-3}$.

Now we will find the greatest common divisor (GCD) of two Laurent polynomials $h_e(z)$ and $h_o(z)$. Since, the GCD of the Laurent polynomials is not unique, we carefully derive several lifting steps as presented below:

Let $x_0(z) = h_e(z)$ and $y_0(z) = h_o(z)$.

According to the Euclidean algorithm, to compute GCDs of $x_0(z)$ and $y_0(z)$ when $y_0(z) \neq 0$ and the degrees of the polynomials $|x_0(z)| \geq |y_0(z)|$, we need to iterate the following equation set in (4.10) from *i*=0:

$$\begin{aligned} x_{i+1}(z) &= y_i(z) \\ y_{i+1}(z) &= x_i(z) \% y_i(z) = x_i(z) - y_i(z) q_{i+1}(z) \end{aligned} \tag{4.10}$$

Here, $q_{i+1}(z) = x_i(z) / y_i(z)$ is the quotient with the degree $|q_{i+1}(z)| = |x_i(z)| - |y_i(z)|$. We need to choose a quotient with appropriate degree with the condition: $|y_{i+1}(z)| < |y_i(z)|$.

When $y_{n-1}(z)$ will be zero for the minimum value of *n*, $x_n(z)$ will be the GCD of $x_0(z)$ and $y_0(z)$. So we get,

$$\begin{bmatrix} x_n(z) \\ 0 \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} 0 & 1 \\ 1 & -q_i(z) \end{bmatrix} \begin{bmatrix} x_0(z) \\ y_0(z) \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x_0(z) \\ y_0(z) \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_n(z) \\ 0 \end{bmatrix} \tag{4.11}$$

So the first step using the equation set (4.10) will be

$x_1(z) = y_0(z) = h_1 + h_3 z^{-1} + h_5 z^{-2} + h_7 z^{-3}$ and

$y_1(z) = x_0(z) \% y_0(z) = x_0(z) - y_0(z) q_1(z)$.

Here, the degree of $|q_1(z)|$ should be $|x_0(z)| - |y_0(z)| = 3 - 3 = 0$ where $|y_1(z)| < |y_0(z)|$. Then,

$$\begin{aligned} y_1(z) &= h_0 + h_2 z^{-1} + h_4 z^{-2} + h_6 z^{-3} - (h_1 + h_3 z^{-1} + h_5 z^{-2} + h_7 z^{-3})(\frac{h_6}{h_7}) \\ &= (h_0 - h_1 \frac{h_6}{h_7}) + (h_2 - h_3 \frac{h_6}{h_7}) z^{-1} + (h_4 - h_5) \frac{h_6}{h_7} z^{-2} \end{aligned}$$

As we can see, $|q_1(z)| = \left| \frac{h_6}{h_7} \right| = 0$ and $|y_1(z)| = 2 < |y_0(z)|$. However, the selection of $q_i(z)$ is also not unique. For example, $q_1(z)$ can be h_0/h_1 as well while meeting the condition of degree requirement of $q_1(z)$ and $y_1(z)$. In convention, $x_n(z) = k$ is a constant. In our conversions, we do not impose this requirement, which provides us more flexibility to achieve different options of lifting steps for the same wavelet, but does not have any problem in implementation if the degree

is zero. So, *k* can be a function of *z* or a Laurent polynomial with the degree of zero (as shown later in Scheme 3 in Table 4.1).

From eqn (4.11), we find the following factorization:

$$\begin{bmatrix} h_e(z) \\ h_o(z) \end{bmatrix} = \begin{bmatrix} x_0(z) \\ y_0(z) \end{bmatrix} = \prod_{i=1}^{4} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k \\ 0 \end{bmatrix}$$

where,

$q_1(z) = -3.10293149$, $q_2(z) = 0.11602641z^{-1} + 0.35344919$,

$q_3(z) = 0.18109028z^{-1} - 1.1327374$,

$q_4(z) = -0.22141421z^{-1} - 0.066100553$,

$k = x_4(z) = 2.2779381$.

As shown in eqn (4.3), the matrix $\hat{P}(z)$ would be:

$$\hat{P}(z) = \begin{bmatrix} h_e(z) & \hat{g}_e(z) \\ h_o(z) & \hat{g}_o(z) \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}$$

$$= \prod_{i=1}^{2} \begin{bmatrix} 1 & q_{2i-1}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_{2i}(z) & 1 \end{bmatrix} \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}$$

We have chosen $\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}$ to ensure the determinant of $\hat{P}(z)$ is 1. $P(z)$ can now be recovered from $\hat{P}(z)$, i.e. g(z) from $\hat{g}(z)$, using the following relation:

$$\begin{aligned} P(z) &= \hat{P}(z) \begin{bmatrix} 1 & R(z) \\ 0 & 1 \end{bmatrix} \\ &= \prod_{i=1}^{2} \begin{bmatrix} 1 & q_{2i-1}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_{2i}(z) & 1 \end{bmatrix} \begin{bmatrix} 1 & r(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} \end{aligned} \tag{4.12}$$

where, $r(z) = R(z)k^2$. Combining eqns (4.9) and (4.12), we find:

$$r(z) = +0.23869460 * z^3 - 1.34832352 * z^2 + 4.51642196 * z$$

In another way, factorization of $P(z)$ can be expressed as:

$$P(z) = \tilde{P}(z) = \begin{bmatrix} 1 & s_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_{11} + t_{12}z^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & s_{21} + s_{22}z^{-1} \\ 0 & 1 \end{bmatrix} \cdots$$
$$\cdots \begin{bmatrix} 1 & 0 \\ t_{21} + t_{22}z^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & s_{31}z^3 + s_{32}z^2 + s_{33}z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} \quad (4.13)$$

Where,

$s_1 = -3.10293149$, $t_{11} = 0.35344919$, $t_{12} = 0.11602641$,
$s_{21} = -1.13273740$, $s_{22} = 0.18109028$, $t_{21} = -0.06610055$,
$t_{22} = -0.22141421$, $s_{31} = 0.23869460$, $s_{32} = -1.34832352$,
$s_{33} = 4.51642196$, $k = 2.27793811$.

So, the analysis polyphase matrix should be

$$\tilde{P}(1/z)^T = \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_{33}z^{-1} + s_{32}z^{-2} + s_{31}z^{-3} & 1 \end{bmatrix} \cdots$$
$$\cdots \begin{bmatrix} 1 & t_{22}z + t_{21} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_{22}z + s_{21} & 1 \end{bmatrix} \begin{bmatrix} 1 & t_{12}z + t_{11} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_1 & 1 \end{bmatrix} \quad (4.14)$$

If we split the input sequence x into even (x_e) and odd (x_o) components, the decomposed output will be: $\begin{bmatrix} s \\ d \end{bmatrix} = \tilde{P}(1/z)^T \begin{bmatrix} x_e \\ x_o \end{bmatrix}$

Finally, it brings the lifting steps of one of our three schemes (known as scheme 1) as given in (4.15):

$$\begin{aligned} x_{o1} &= s_1 x_e + x_o \\ x_{e1} &= x_e + (t_{11} + t_{12}z)x_{o1} \\ x_{o2} &= (s_{21} + s_{22}z)x_{e1} + x_{o1} \\ x_{e2} &= x_{e1} + (t_{21} + t_{22}z)x_{o2} \\ x_{o3} &= (s_{31}z^{-3} + s_{32}z^{-2} + s_{33}z^{-1})x_{e2} + x_{o2} \\ s &= kx_{e2} \\ d &= x_{o3}/k \end{aligned} \quad (4.15)$$

Using the similar techniques, we present two more variants of the orthogonal lifting schemes as summarized in Table 4.1. It should be noted here that, using the proposed technique, many other variants could be formed. One should choose the best one from these based on factors like image reconstruction quality such as, entropy, coding gain, and performance metrics

like Peak Signal to Noise Ratio (PSNR) and Structural Similarity index (SSIM) [44] and implementation cost (such as, cell count).

Table 4.1 Orthogonal lifting schemes for D8 – three variants

Lifting scheme	Polyphase matrix	Value of coefficients
1	$\tilde{P}_2(1/z)^T = \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_{33}z^{-1} + s_{32}z^{-2} + s_{31}z^{-3} & 1 \end{bmatrix} \cdots$ $\cdots \begin{bmatrix} 1 & t_{22}z + t_{21} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_{22}z + s_{21} & 1 \end{bmatrix} \begin{bmatrix} 1 & t_{12}z + t_{11} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_1 & 1 \end{bmatrix}$	$s_1 = -3.10293149$, $t_{11} = 0.35344919$, $t_{12} = 0.11602641$, $s_{21} = -1.13273740$, $s_{22} = 0.18109028$, $t_{21} = -0.06610055$, $t_{22} = -0.22141421$, $s_{31} = 0.23869460$, $s_{32} = -1.34832352$, $s_{33} = 4.51642196$, $k = 2.27793811$.
2	$\tilde{P}_4(1/z)^T = \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_3 z^{-3} & 1 \end{bmatrix} \begin{bmatrix} 1 & t_{22}z^3 + t_{21}z^2 \\ 0 & 1 \end{bmatrix} \cdots$ $\cdots \begin{bmatrix} 1 & 0 \\ s_{22}z^{-1} + s_{21}z^{-2} & 1 \end{bmatrix} \begin{bmatrix} 1 & t_{12}z + t_{11} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_1 & 1 \end{bmatrix}$	$s_1 = -3.10293149$, $t_{11} = 0.29195313$, $t_{12} = -0.07630009$, $s_{21} = -1.66252835$, $s_{22} = 5.19949157$, $t_{21} = 0.03789275$, $t_{22} = -0.00672237$, $s_3 = 0.31410649$, $k = 2.61311837$
3	$\tilde{P}_1(1/z)^T = \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_3 z^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & t_{22}z + t_{21} \\ 0 & 1 \end{bmatrix} \cdots$ $\cdots \begin{bmatrix} 1 & 0 \\ s_{22} + s_{21}z^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & t_{12}z + t_{11} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s_1 & 1 \end{bmatrix}$	$s_1 = -3.10293149$, $t_{11} = 0.29195313$, $t_{12} = 0.11602641$, $s_{21} = -5.1994916$, $s_{22} = 16.261204$, $t_{21} = -0.05762195$, $t_{22} = -0.00068729$, $s_3 = 3.072278281$, $k = 8.172424199$

4.3.2 Integer mapping of the formed D8 filters

After lifting conversion, we have irrational coefficients for each scheme, such as s_1, t_1, s_2, t_2, and so on. Implementation of these coefficients using floating-point (FP) or fixed-point hardware will take a large amount of resources. Also, the quantization or rounding step adds

more error into the reconstruction. To mitigate the problem, we now use the integer mapping [32] that enables low-cost implementation [7]. We first take different precisions of the coefficients by multiplying them by 2^b where b is the precision in the number of bits. The new coefficient for old coefficient s_1 is $S_1 = \|s_1 2^b\|$ where $\|*\|$ denotes the symbol for rounding. The new lifting steps, $x_{o1} = s_1 x_e + x_o$ and $x_{e1} = x_e + (t_{11} + t_{12}\mathrm{z})x_{o1}$ would be $x_{o1} = \|(S_1 x_e)/2^b\| + x_o$ and $x_{e1} = x_e + \|(T_{11}x_{o1} + T_{12}x_{o1}\mathrm{z})/2^b\|$ respectively while [7] utilizes the floor ($\lfloor * \rfloor$) instead of $\|*\|$.

4.3.3 Elimination of scaling stage

Now we have three variants of integer based D8 lifting filters, they still suffer from quantization error due to a scaling operation which is the last step of the lifting process (i.e., $\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}$). It should be noted that this scaling stage is common in all lifting wavelet transforms. Although, in integer lifting transform, all other lifting steps are perfectly invertible without any error, it is the scaling operation that incurs error, and thereby makes the entire process lossy [31][32]. It is also noted that the scaling stage can be factorized into lifting steps, as presented in [31] and shown below in eqn (4.16), which can make the implementation theoretically lossless.

$$\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix} = \begin{bmatrix} 1 & k-k^2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1/k & 1 \end{bmatrix} \begin{bmatrix} 1 & k-1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \tag{4.16}$$

However, in practice, these extra lifting steps in eqn (4.16) will not only increase the hardware resources, but also incur additional computational error due to the approximation of the coefficients in the integer based implementation. As a result, this approach (also known as “lifted scaling”) is not popular in literature [45]. For example, JPEG-2000 lossy algorithm uses CDF 9/7 wavelet transform with scaling without factorizing the scaling step to the lifting steps [3].

Therefore, our aim in this work is to eliminate the scaling stage to make the entire process lossless. In this section, we first determine both theoretically and experimentally if that is a choice. In Section 4.4 later, we also present comparative analysis between our “no/without scaling” method with the “lifted scaling” method.

Transform coding gain is a metric that determine the quality of a transform (while entropy is another metric). For the input sequence x of an orthogonal transform, that is transformed into $y_0, y_1, \ldots, y_{N-1}$, the coding gain ($G_T$) is given by [38]:

$$G_T = \frac{\frac{1}{N}\sum_{n=0}^{N-1}\sigma_{y_n}^2}{(\prod_{n=0}^{N-1}\sigma_{y_n}^2)^{\frac{1}{N}}} = \frac{\sigma_x^{\,2}}{(\prod_{n=0}^{N-1}\sigma_{y_n}^2)^{\frac{1}{N}}} \tag{4.17}$$

where $\sigma_x^{\,2}$ and $\sigma_{y_n}^2$ represents the variance of the input sequence x and the output sequence y_n. For Daubechies wavelet, there are only two outputs. So from (4.17),

$$G_T = \frac{\sigma_x^{\,2}}{(\sigma_{y_0}^2\sigma_{y_1}^2)^{\frac{1}{2}}} \tag{4.18}$$

For a lifting-based transform, as shown in Fig. 4.2, $y_0 = ky_{0p}$ and $y_1 = y_{1p}/k$.

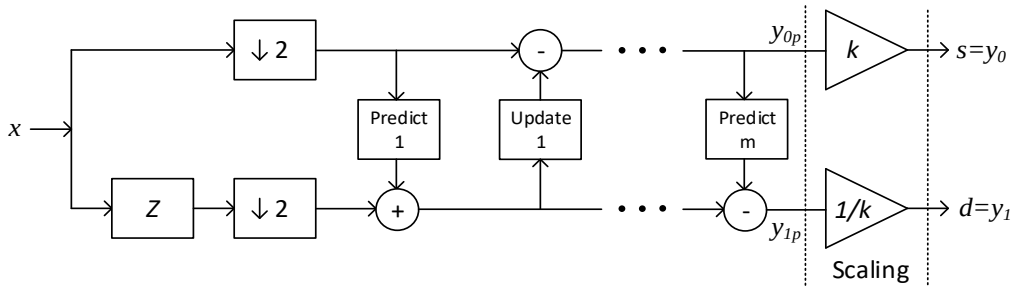


Fig. 4.2 A general lifting scheme with scaling steps shown. The scaling step results from the first factor of the polyphase matrix ($\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}$).

The variance of the y_0 sequence is:

$$\sigma_{y_0}^2 = \frac{1}{m}\sum_{i=1}^{m}(y_{0_i} - \mu_{y_0})^2 . \tag{4.19}$$

Here, the mean of y_0 is:

$$\mu_{y_0} = \frac{1}{m}\sum y_{0_i} = \frac{1}{m}\sum ky_{0p_i} = \frac{k}{m}\sum y_{0p_i} = k\mu_{y_{0p}} . \tag{4.20}$$

From (4.19), we find:

$$\sigma_{y_0}^2 = \frac{1}{m}\sum_{i=1}^{m}(ky_{0p_i} - k\mu_{y_{0p}})^2 = \frac{k^2}{m}\sum_{i=1}^{m}(y_{0p_i} - \mu_{y_{0p}})^2 = k^2\sigma_{y_0p}^2$$

Thus we find that $\sigma_{y0}^2 = k^2\sigma_{y_{op}}^2$, when $y_0 = ky_{0p}$. Similarly, $\sigma_{y1}^2 = \sigma_{y_{1p}}^2 / k^2$ as well since $y_1 = y_{1p} / k$. Using this relation, we find a new expression of transform coding gain:

$$G_T = \frac{\sigma_x^2}{(\sigma_{y_{0p}}^2 \sigma_{y_{1p}}^2)^{\frac{1}{2}}} \quad . \tag{4.21}$$

It is seen from eqn (4.21) that the coding gain is independent of scaling parameters and so the scaling stage can be eliminated without affecting the gain adversely. We will present experimental results later in Section 4.4 to support our claim.

4.4 Experimental Results and Analysis

In this section, we will conduct several experiments to assess the performance of the proposed D8 schemes and also make effort to find the optimum scheme (out of three presented in Table 4.1) that is suitable for efficient implementation.

Table 4.2 Effect of Discarding Scaling stage using Lena image

		Classic D8[1]	Proposed lifting schemes		
			1	2	3
Lifting with scaling	G_T	20.08	20.89	20.89	20.89
	Entropy (bpp)	5.95	5.94	5.94	5.99
	RMSE ($x10^{-14}$)	2.2	5.0	4.6	4.0
Lifting without scaling	G_T	-	20.89	20.89	20.89
	Entropy (bpp)	-	5.94	5.91	6.11
	RMSE ($x10^{-14}$)	-	4.7	4.3	4.0

[1]by taking MATLAB double-precision (64-bits)

In our first experiment, we demonstrate the impact of eliminating scaling stage (as discussed in Section 4.3.3) in floating-point (FP) implementation and compare the results with the classical D8 in Table 4.2. The metrics chosen are transform coding gain (G_T), entropy (in bpp), and error of reconstruction (in root mean square error – RMSE), which are essential in observing codec's performance [45]. The classical D8 implementation in MATLAB (using double precision) guarantees the highest accuracy of the lifting schemes.

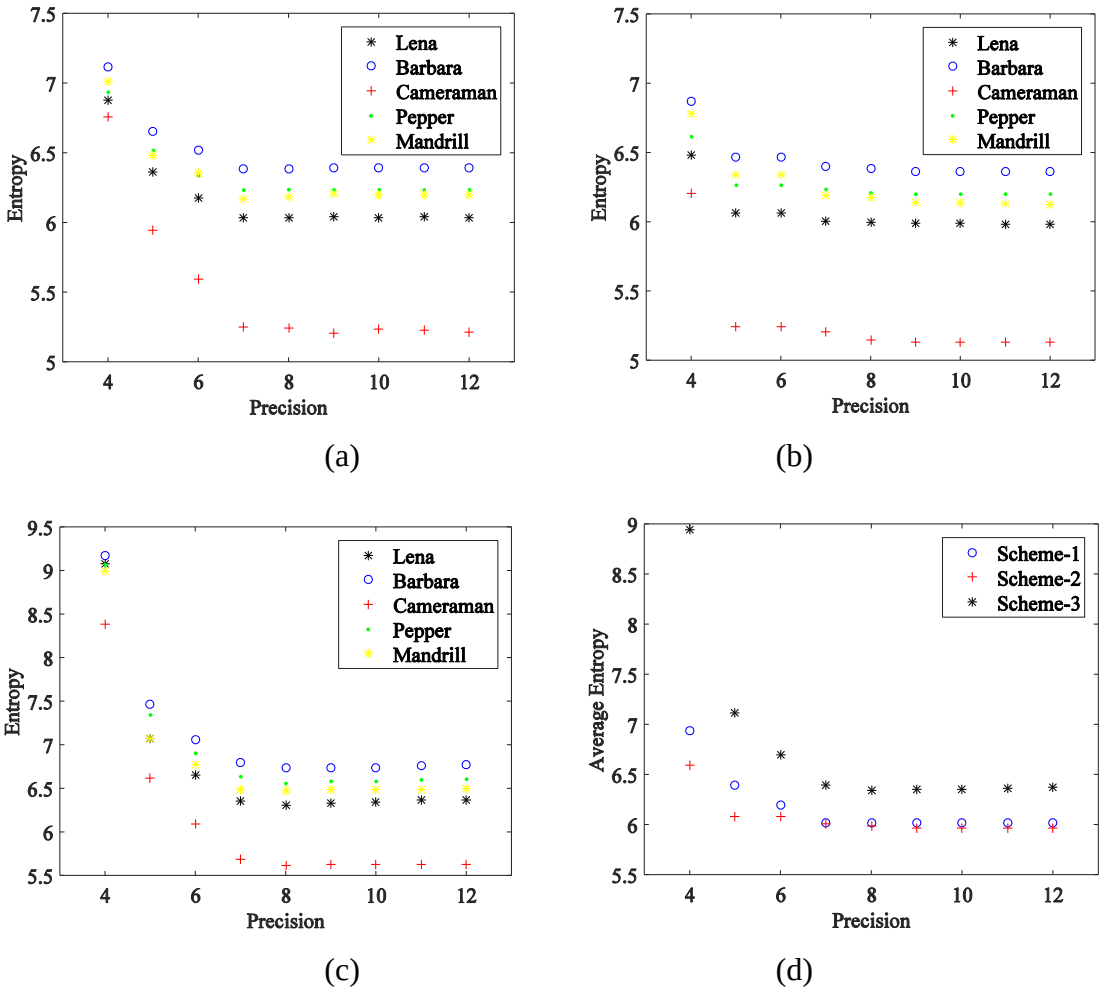


Fig. 4.3 Entropy for different precisions of coefficients without scaling: (a) scheme-1, (b) scheme-2, (c) scheme-3 and (d) comparison among proposed schemes (in each case, PSNR is infinite)

As we see in Table 4.2, G_T remains same for all schemes with and without the scaling process (as we expect in eqn (4.21)). The reconstruction errors (RMSE) are very small (in the range of 10^{-14}) for all schemes. Entropy remains same for all schemes when scaling is used, but varies when discarded. Therefore, we see that discarding the scaling stage does not affect the output of the wavelet filters.

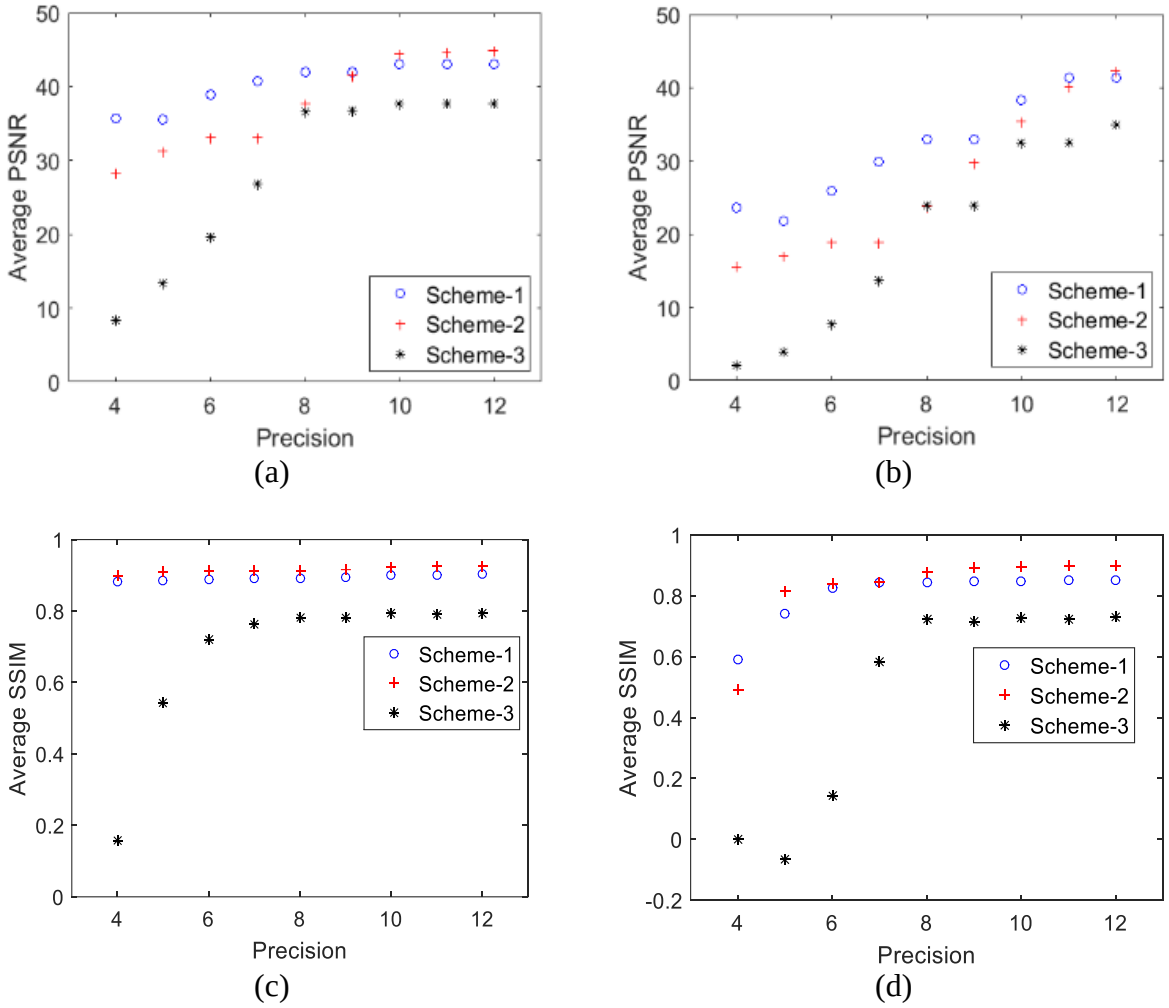


Fig. 4.4 Reconstruction quality (in PSNR in dB and in SSIM) for different precisions with scaling for (a) and (c) 1-level decomposition; (b) and (d) 6-level decomposition

Now we apply our schemes (without scaling) taking different bit precisions of the coefficients on several benchmark images. The results are shown in Fig 4.3. For each case, the entropy decreases with increasing bit-width, as expected, until we reach a threshold of 8-bits, after which the entropy does not change significantly. Fig. 4.3(d) compares the three schemes using average entropy values which indicates that scheme-1 and 2 perform the best. Note that, the integer mapped algorithms (without scaling) are completely reversible and the entire transform process is lossless. As a result, the PSNR of the reconstructed image is infinite.

In order to strength our approach of eliminating the scaling stage, we conduct more experiments where we compare the performance of the proposed schemes with scaling for multiple level decompositions as shown in Fig. 4.4. As indicated by the results, the transform process is not lossless due to the inclusion of scaling; the reconstruction quality also degrades when higher level of decomposition used.

Along the previous lines, we have conducted another experiment with our schemes combined with a lifted scaling (as given in eqn (4.16)), and present the results in Fig. 4.5. The average entropy values decrease for all cases of 'no scaling' showing slightly better energy compactness while decreasing the cost. In addition to five benchmark images, we also used 44 different images (16 colors and 28 monochromes) from SIPI Misc database [39] on the lifting schemes without scaling and found similar results as shown in Fig. 4.6.

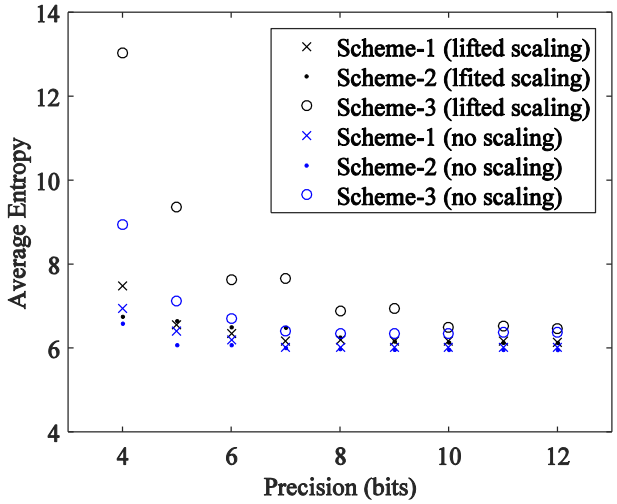
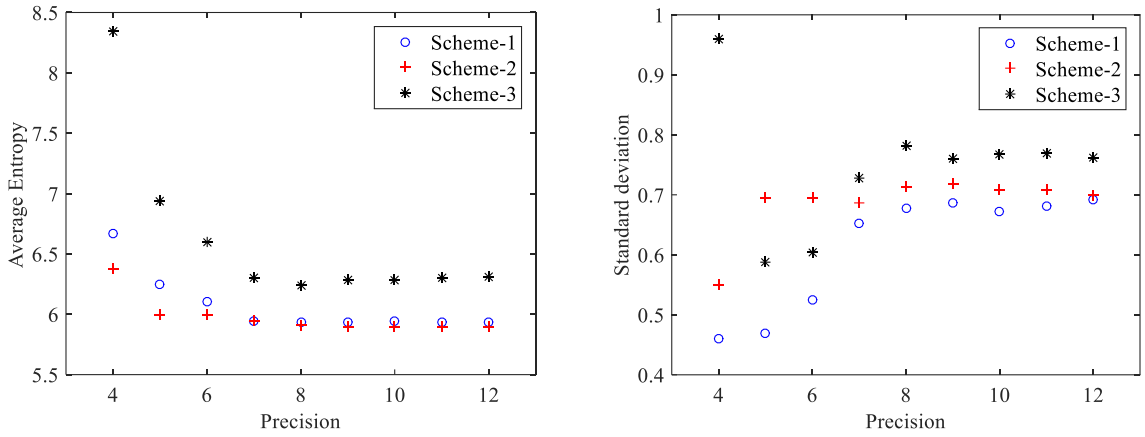


Fig. 4.5 Comparison of entropy for different precisions for all schemes using "lifted scaling [31]" and "no scaling (ours)" using five benchmark images



(a) (b)

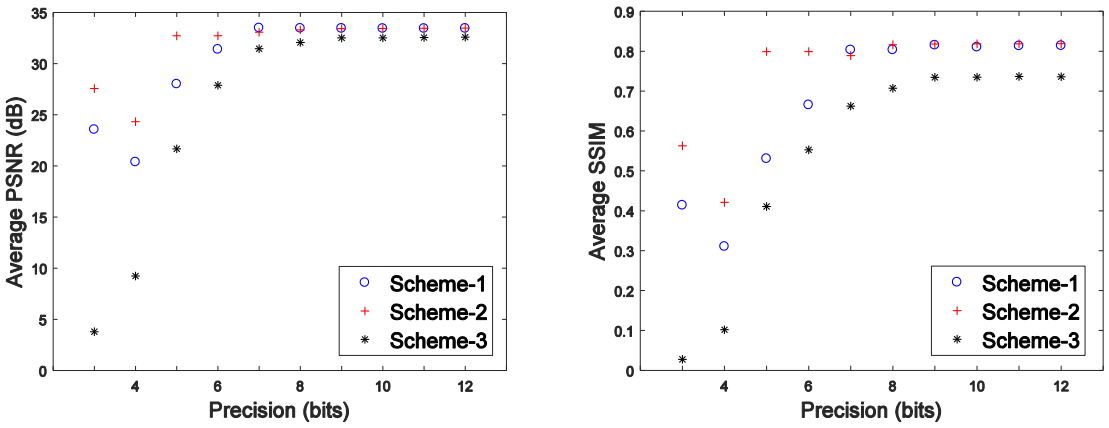
Fig. 4.6 Proposed schemes without scaling using SIPI Misc database [39]: (a) Average entropy and (b) standard deviation

At this point, we would like to study what effect our schemes have on the wavelet regularity, which prevents the energy of an image from "spilling into" higher energy zone, which is known as DC leakage [46]. Low DC leakage is desirable. In the next experiment, we applied 1-level decomposition of our schemes on five benchmark images, and then reconstructed them taking only the approximation or low frequency components (like, *s* in Fig. 4.1(b)). The average PSNR and SSIM values of these images are shown in Fig. 4.7. It can be seen that, like before, both indices become stable and yet acceptable (PSNR over 32dB and SSIM over 0.75) after 8-bits precision. It indicates that, our proposed schemes perform well in preventing DC leakage and thereby preserving energy components within the lower frequency band – a desired feature of an image transform tool.

In Table 4.2, we only presented the results for 'Lena' image; now we extend the experiment for three schemes using the images from the SIPI Misc database and present the results in Table 4.3. The results are consistent with that of Table 4.2, which indicates that scheme 1 without scaling performs very similar to the classic D8 approach (that has the highest accuracy). In addition, these experimental results show that the elimination of scaling step does not affect the coding gain of the transform, which is also consistent with the theoretical

experiment presented earlier in Section 4.3. Considering the above performance analysis, we move forward with schemes 1 and 2 (without scaling) for the rest of the study.

In Table 4.4, we compare the entropy of the processed image by our schemes (using 8-bits) with biorthogonal wavelets like 9/7 and 5/3 (using 64-bits) used in JPEG-2000 for five benchmark images. The results show that, even with lower bit-precision, our schemes produce similar performance as high-precision wavelets.



(a) (b)

Fig. 4.7 Reconstruction results using only approximation coefficients for five benchmark images: (a) average PSNR in dB, (b) average SSIM

Table 4.3 Effect of Discarding Scaling stage using image Database

		Classic D8[1]		Proposed lifting schemes					
				1		2		3	
		μ[5]	σ[6]	μ	σ	M	σ	M	σ
wS[3]	G_T	16.1	40.58	14.88	27.31	14.88	27.31	14.88	27.31
	Entropy	5.82	0.97	5.79	1.01	5.79	1.01	5.79	1.01
	RMSE[2]	2.34	0.76	5.25	1.69	4.93	1.48	4.18	1.25
woS[4]	G_T	-	-	14.88	27.31	14.88	27.31	14.88	27.31
	Entropy	-	-	5.73	0.98	5.73	0.98	5.99	1.13
	RMSE[2]	-	-	4.94	1.53	4.55	1.26	4.11	1.10

[1] Using MATLAB double-precision (64-bits), [2]root mean square error in 10^{-14}, [3]wS = lifting with scaling, [4]woS = lifting without scaling, [5]μ = mean, [6]σ = standard deviation

Table 4.4 Comparison of Entropy with other wavelet transforms

	Lena	Barbara	Camera	Pepper	Mandrill	Average
9/7[1]	6.07	6.55	4.99	6.22	6.02	5.97
5/3[1]	6.33	6.87	5.54	6.43	6.75	6.38
Scheme-1[2]	6.04	6.38	5.25	6.23	6.17	6.01
Scheme-2[2]	6.00	6.38	5.15	6.21	6.17	5.98

[1]Standard CDF 9/7 and 5/3 with 64-bit double precision using MATLAB built-in function;

[2]Proposed scheme-1 and 2 with 8-bit precision and without scaling

4.5 Hardware Implementation

Fig. 4.8 shows the hardware implementation of our low-cost lossless D8 filter structure. The filters are implemented in both software (MATLAB) and hardware (Verilog HDL). The integer-mapped coefficients are shown in Table 4.5. All multiplication operations are eliminated by shift and add operations. The number of adders is also minimized using Canonical Signed Digit (CSD) representation. Another alternative method of multiplication is described in [40] which can also be used to reduce the number of adders.

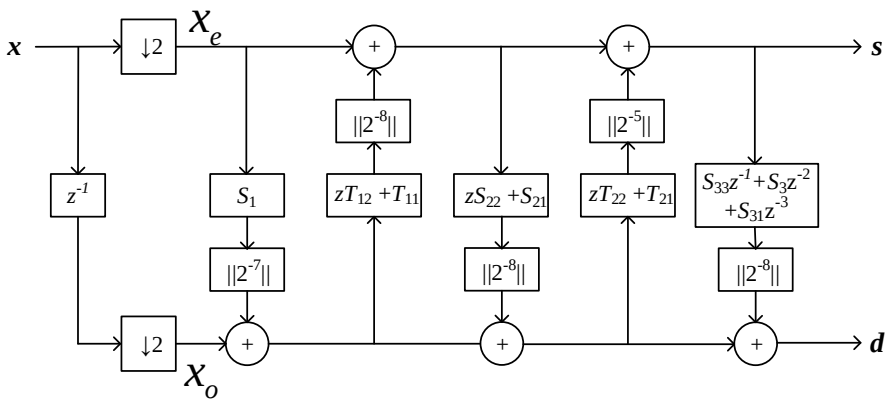
Table 4.5 Integer polynomial mapping of the D8 coefficients

	Scheme-1 coefficients		Scheme-2 coefficients	
	Actual value	Integer mapping	Actual value	Integer mapping
S_1	-397	$-(2^8+2^7+2^3+2^2+2^0)$	-397	$-(2^8+2^7+2^3+2^2+2^0)$
T_{11}	90	$2^6+2^4+2^3+2^1$	74	$2^6+2^3+2^1$
T_{12}	29	$2^5-2^1-2^0$	-19	$-2^4-2^2+2^0$
S_{21}	-289	$-(2^8+2^5+2^0)$	-425	$-(2^8+2^7+2^5+2^3+2^0)$
S_{22}	46	$2^5+2^4-2^1$	1331	$2^{10}+2^8+2^5+2^4+2^1+2^0$
T_{21}	-2	-2^1	9	2^3+2^0
T_{22}	-7	-2^3+2^0	-1	-2^0
S_3	-	-	5	2^2+2^0
S_{31}	61	$2^6-2^2+2^0$	-	-
S_{32}	-345	$-2^8-2^7+2^5+2^3-2^0$	-	-
S_{33}	1156	$2^{10}+2^7+2^2$	-	-

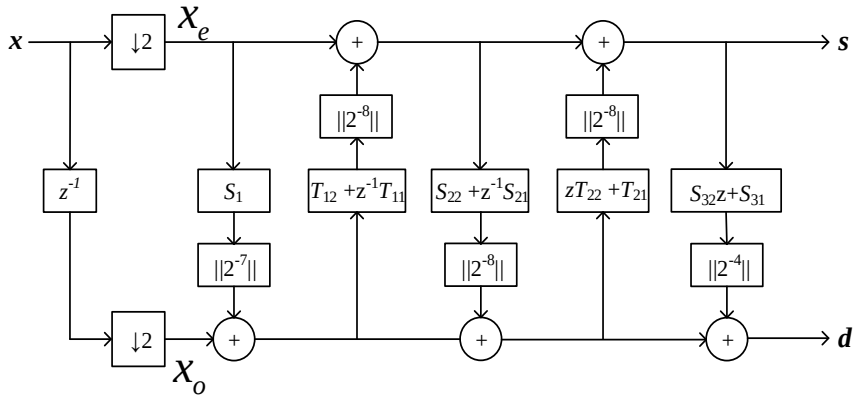
Table 4.6 Cost of implementation

Schemes		Implementation		
		LUTs	Regiter-bits	Frequency (MHz)
Scheme-1	Without scaling	**470**	**133**	**63**
	With scaling	478	133	53.2
	With lifted scaling	487	133	60
Scheme-2	Without scaling	**389**	**110**	**112**
	With scaling	391	110	112
	With lifted scaling	400	110	112
Classic D8 (double precision)		15,321*	-	-

*implemented using HDL coder in MATLAB with fixed-point and 16 inputs



(a)



(b)

Fig. 4.8 Hardware implementation of (a) scheme-1, (b) scheme-2

The cost of implementation in Xilinx Virtex6 FPGA (model: XC6VLX75T) is shown in Table 4.6. Our proposed schemes provide overall best results (bolded) in terms of resource cost and performance. Table 4.7 compares our schemes with other methods in literature. There is a large collection of works in literature concerning smaller wavelets (D2, D4, etc.) to bi-orthogonal wavelets, but for the sake of fair comparison, we limit our study to only D6 and D8 family. As seen from the table that our schemes (that has no scaling stage) decrease the cost significantly when compared to other D8 and many D6 architectures while offering a completely lossless implementation and closely maintaining the performance of the classical D8 wavelet.

4.6 Conclusion

This work proposes a low-cost algorithm and lossless implementation of Daubechies 8-tap wavelets (D8) filter. Three variants of lifting D8 structures have been presented and their performance is analyzed. The architecture eliminates the scaling stage and maintains a performance close to classical D8 implementation. The cost of hardware resources is also significantly reduced making D8 very suitable for fast and real-time image processing applications.

Table 4.7 Performance Comparisons with existing Integer based Daubechies Wavelet Architectures

	Wahid [10]	Madishetty [9]	Pranav [5]	Classic D8 [13]	Al-Haj [13]	Al-Haj [41]	Wahid [12]	Longa [42]	Huang [43]	Proposed D8 (Scheme 1)	Proposed D8 (Scheme 2)
Wavelet	D6	D6	D6	D8	D8	D8	D8	D8	D8	D8	D8
Architecture	Matrix	Matrix	Lifting (regular)	Matrix	DA[2]	DA	Matrix	DA	DA	Lifting (Integer)	Lifting (Integer)
Logic cells (FPGA)	680	248	190	1120	748	2334[3]	687	614	2619	470	389
Register bits	494	177	-	-	-	-	179	-	9907[1]	133	110
PSNR (dB)	39	57.12	71.54	-	-	-	90	41.3	-	∞	∞
RMSE	2.86	0.36	0.07	-	-	-	0.01	2.20	-	0	0
Device	Virtex-E	Virtex-6	Virtex-6	Virtex	Virtex	Virtex	Cyclone	Stratix II	Stratix	Virtex-6	Virtex-6

[1]Register cost is in Cyclone 2C35 FPGA; [2]DA = Distributed arithmetic; [3]1167 slices

4.7 References

[1] P. S. Addison, *The illustrated wavelet transform handbook: introductory theory and applications in science, engineering, medicine and finance*. CRC press, 2002.

[2] M. Vetterli and C. Herley, "Wavelets and filter banks: Theory and design," *IEEE Trans. signal Process.*, 40, 9, pp. 2207–2232, 1992.

[3] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *IEEE Signal Process. Mag.*, vol. 18, no. 5, pp. 36–58, 2001.

[4] L. Chun-Lin, "A tutorial of the wavelet transform," Dept. Elect. Eng., Nat. Taiwan Univ., Taipei, Taiwan, Tech. Rep., 2010. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6942141

[5] P. Balakrishnan, M. M. Hasan, and K. A. Wahid, "An Efficient Algorithm for Daubechies Lifting Wavelets Using Algebraic Integers," *Canadian J. of Electrical & Computer Engr.*, vol. 37, no. 3. pp. 127–134, 2014.

[6] B. K. Mohanty, A. Mahajan, and P. K. Meher, "Area-and power-efficient architecture for high-throughput implementation of lifting 2-D DWT," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 59, no. 7, pp. 434–438, 2012.

[7] M. M. Hasan and K. A. Wahid, "Low-cost Architecture of Modified Daubechies Lifting Wavelets using Integer Polynomial Mapping," *IEEE Trans. Circuits Syst. II*, vol. 64, no. 5, pp. 585–589, 2017.

[8] M. K. Mandal, S. Panchanathan, and T. Aboulnasr, "Choice of wavelets for image compression," in *Information Theory and Applications II*, Springer, 1996, pp. 239–249.

[9] S. K. Madishetty, A. Madanayake, R. J. Cintra, S. Member, and V. S. Dimitrov, "Precise VLSI Architecture for AI Based 1-D / 2-D Daub-6 Wavelet Filter Banks With Low Adder-Count," *IEEE Trans. Circuits Syst. I*, pp. 1–10, 2014.

[10] K. Wahid, V. Dimitrov, and G. Jullien, "VLSI architectures of Daubechies wavelet transforms using algebraic integers," *J. Circuits, Syst., Comput.*, vol. 13, no. 6, pp. 1251–1270, 2004.

[11] P. Sangeetha, M. Karthik, and T. KalavathiDevi, "VLSI architectures for the 4-tap and

6-tap 2-D Daubechies wavelet filters using pipelined direct mapping method," in *Proc. ICIIECS*, 2015, pp. 1–6.

[12] K. A. Wahid, M. A. Islam, and S.-B. Ko, "Lossless implementation of Daubechies 8-tap wavelet transform," *Proc. ISCAS*. pp. 2157–2160, 2011.

[13] A. M. Al-Haj, "Fast discrete wavelet transformation using FPGAs and distributed arithmetic," *Int. J. Appl. Sci. Eng.*, vol. 1, no. 2, pp. 160–171, 2003.

[14] S. Masud and J. V McCanny, "Finding a suitable wavelet for image compression applications," *Proc. IEEE ICASSP*, vol. 5. pp. 2581–2584, 1998.

[15] E. A. Rashed, I. A. Ismail, and S. I. Zaki, "Multiresolution mammogram analysis in multilevel decomposition," *Pattern Recognit. Lett.*, vol. 28, no. 2, pp. 286–292, Jan. 2007.

[16] M. Z. do Nascimento, A. S. Martins, L. A. Neves, R. P. Ramos, E. L. Flores, and G. A. Carrijo, "Classification of masses in mammographic image using wavelet domain features and polynomial classifier," *Expert Syst. Appl.*, 40, (15), pp. 6213–6221, Nov. 2013.

[17] K. Asaduzzaman, M. B. I. Reaz, F. Mohd-Yasin, K. S. Sim, and M. S. Hussain, "A study on discrete wavelet-based noise removal from EEG signals," in *Advances in Computational Biology*, Springer, 2010, pp. 593–599.

[18] B. G. Negash and H. Nikookar, "Wavelet-based multicarrier transmission over multipath wireless channels," *Electron. Lett.*, vol. 36, no. 21, pp. 1787–1788, 2000.

[19] V. Holub and J. Fridrich, "Designing steganographic distortion using directional filters," *Proc. IEEE WIFS*. pp. 234–239, 2012.

[20] D. Cho and T. D. Bui, "Multivariate statistical modeling for image denoising using wavelet transforms," *Signal Process. Image Commun.*, vol. 20, no. 1, pp. 77–89, Jan. 2005.

[21] G. W. Horgan, "Using wavelets for data smoothing: A simulation study," *J. Appl. Stat.*, vol. 26, no. 8, pp. 923–932, Dec. 1999.

[22] J. Z. Wang, G. Wiederhold, O. Firschein, and S. X. Wei, "Content-based image indexing and searching using Daubechies' wavelets," *Int. J. Digit. Libr.*, vol. 1, no. 4, pp. 311–328, 1998.

[23] X. Liu and J. Raja, "Analyzing engineering surface texture using wavelet filter," *Proc. SPIE*, vol. 2825, pp. 942–949, Oct. 1996. [Online]. Available: https://www.spiedigitallibrary.org/conference-proceedings-ofspie/2825/0000/Analyzing-engineering-surface-texture-using-waveletfilter/10.1117/12.255308.short?SSO=1

[24] C. Pang and M. Kezunovic, "Fast Distance Relay Scheme for Detecting Symmetrical Fault During Power Swing," *IEEE Transactions on Power Delivery*, vol. 25, no. 4. pp. 2205–2212, 2010.

[25] W. Zhao, Y. H. Song, and Y. Min, "Wavelet analysis based scheme for fault detection and classification in underground power cable systems," *Electr. Power Syst. Res.*, 53 (1), pp. 23–30, Jan. 2000.

[26] K. M. Silva, W. L. A. Neves, and B. A. Souza, "Distance protection using a wavelet-based filtering algorithm," *Electr. Power Syst. Res.*, vol. 80, no. 1, pp. 84–90, Jan. 2010.

[27] M. Tico, E. Immonen, P. Ramo, P. Kuosmanen, and J. Saarinen, "Fingerprint recognition using wavelet features," *Proc. ISCAS*, vol. 2. pp. 21–24, 2001.

[28] J. C. Letelier and P. P. Weber, "Spike sorting based on discrete wavelet transform coefficients," *J. Neurosci. Methods*, vol. 101, no. 2, pp. 93–106, Sep. 2000.

[29] M. Brandfass, W. Coster, U. Benz, and A. Moreira, "Wavelet based approaches for efficient compression of complex SAR image data," in *Proc. IEEE IGARSS*, 1997, vol. 4, pp. 2024–2027.

[30] L. Gan, "Block Compressed Sensing of Natural Images," *Int. Conf. Digital Signal Process.* pp. 403–406, 2007.

[31] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 247–269, 1998.

[32] A. R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, "Wavelet Transforms That Map Integers to Integers," *Appl. Comput. Harmon. Anal.*, vol. 5, no. 3, pp. 332–369, Jul. 1998.

[33] M. D. Adams and F. Kossentni, "Reversible integer-to-integer wavelet transforms for image compression: performance evaluation and analysis," *IEEE Trans on Image Processing*, 9 (6), pp. 1010–1024, 2000.

[34] G. Xuan, J. Zhu, J. Chen, Y. Q. Shi, Z. Ni, and W. Su, “Distortionless data hiding based on integer wavelet transform,” *Electron. Lett.*, vol. 38, no. 25, pp. 1646–1648, 2002.

[35] A. Bilgin, G. Zweig, and M. W. Marcellin, “Three-dimensional image compression with integer wavelet transforms,” *Appl. Opt.*, vol. 39, no. 11, pp. 1799–1814, 2000.

[36] M. F. Tolba, M. A. Ghonemy, I. A. Taha, and A. S. Khalifa, “Using integer wavelet transforms in colored image Steganography,” *Int. J. Intell. Coop. Inf. Syst.*, vol. 4, no. 2, pp. 230–235, 2004.

[37] A. R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, “Lossless Image Compresion Using Integer to Integer Wavelet Transforms.,” in *in Proc. ICIP*, 1997, pp. 596–599.

[38] N. S. Jayant and P. Noll, “Digital Coding of Waveforms--Principles and Applications to Speech and Video Englewood Cliffs.” New Jersey: Prentice-Hall, 1984.

[39] “SIPI Image Database-Misc,” *University of South Carolina*. [Online]. Available: http://sipi.usc.edu/database/database.php?volume=misc. [Accessed: 27-May-2017].

[40] Y. Voronenko and M. Püschel, “Multiplierless multiple constant multiplication,” *ACM Trans. Algorithms*, vol. 3, no. 2, p. 11, 2007.

[41] A. M. Al-Haj, “An FPGA-based parallel distributed arithmetic implementation of the 1-D discrete wavelet transform,” *Informatica*, vol. 29, no. 2, 2005.

[42] P. Longa, A. Miri, and M. Bolic, “A Flexible Design of Filterbank Architectures for Discrete Wavelet Transforms,” *Proc. IEEE ICASSP*, vol. 3. p. III-1441-III-1444, 2007.

[43] Q. Huang, Y. Wang, and S. Chang, “High-Performance FPGA Implementation of Discrete Wavelet Transform for Image Processing,” in *Proc. SOPO*, 2011, pp. 1–4.

[44] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans on Image Processing*, 13 (4), 4, pp. 600-612, Apr. 2004.

[45] P. S. R. Diniz, J. A. K. Suykens, R. Chellappa, and S. Theodoridis, Eds., *Academic Press Library in Signal Processing: Signal Processing Theory and Machine Learning*, vol. 1. Oxford, U.K.: Academic, 2014, chs. 8–9.

[46] A. Abbas and T. D. Tran, "Multiplierless Design of Biorthogonal Dual-Tree Complex

Wavelet Transform using Lifting Scheme," *Proc. Int. Conf. Image Processing*, Atlanta, GA, 2006, pp. 1605-1608.

Chapter 5

Lossless and Low-cost Implementation by Modified Factorization

(General)

Submitted:

Md. Mehedi Hasan and K. A. Wahid, "Factoring Wavelet Transform into Reversible Lifting Steps for Lossless Integer-to-Integer Mapped Implementation," *submitted in* a prestigious journal.

The objective of this thesis research is to propose a wavelet implementation technique which is lossless, low-cost, scaling-step free, integer-based, irrational coefficient-less, friendly to in-place implementation, and which can be generalized. Though chapter 4 achieves these objectives, that approach would require much manual work to find several non-unique factorizations of the polyphase matrix in order to apply that to the other wavelets.

This chapter presents a different strategy to solve the issue. The usual way of factorization of the polyphase matrix in a lifting-based wavelet transform is to factorize the polyphase matrix using Euclidian algorithm for Laurent polynomials which that generates lossy scaling steps. This work proposes a new factorization technique which does not generate lossy scaling steps, but instead incorporates the impact of the scaling steps in the previous lifting steps, and hence, becomes lossless in an integer-based implementation. Some formulas which can convert existing lifting factorization into the proposed factorization are also provided.

This technique is a general approach and is applied and tested on a number of orthogonal wavelets, such as D4, D6, D8, and D10, and biorthogonal wavelets, such as CDF-9/7, CDF 4.2, and CDF 5.1.

Factoring Wavelet Transforms into Reversible Lifting Steps for Lossless Integer-to-Integer Mapped Implementation

Abstract

Implementation of wavelet transform in the lifting method is attractive due to various reasons. That requires the standard wavelet filters to be expressed in a polyphase matrix and factorization of the matrix into lifting steps. This factorization is a well-known procedure that utilizes the Euclidean algorithm for the determination of greatest-common-divisor *k* of Laurent Polynomials. The issue in the conventional factorization is the last factor which is not a ladder like a lifting step, but that is a scaling step with *k.* In most case, the value of k is not 1, which makes integer-to-integer implementation lossy. This paper solves that issue by proposing a novel way of factorization. Moreover, the proposed factorization does not increase the number of factors but replace the last few factors with a lossless ladder like lifting steps. This provides a way to make integer-based implementation purely lossless without increasing the number of non-one factors.

Index Terms

Wavelet transform, Lifting steps, integer-to-integer mapping, factorization, Laurent polynomial.

5.1 Introduction

Discrete wavelet transform (DWT) is a powerful tool to analyze real-time signals or aperiodic, irregular, noisy and transient data because of its capability to explore signals in both

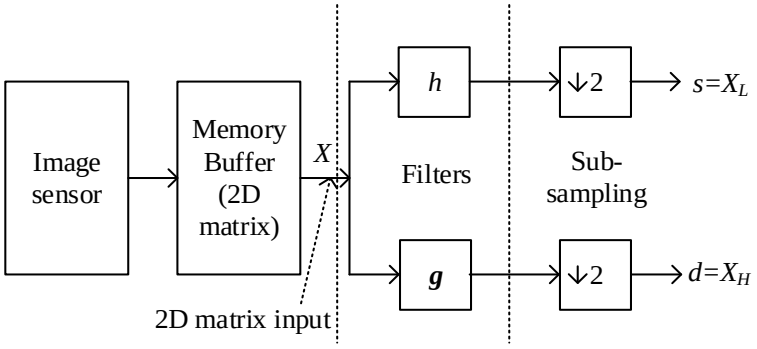
frequency and time domain [1]–[3]. Due to those powerful characteristics, it is extensively used in a wide number of fields in science, mathematics, computer science, medicine, finance, and engineering[3]–[7] and has already been adopted in JPEG2000 image compression standard[8].

The decomposition and reconstruction of a wavelet filter can usually be implemented in two ways: (a) standard convolution and (b) lifting method. There are few advantages of lifting method over the convolution-based method [9]–[11]: (a) in-place implementation of the wavelet transform where the transform can be calculated without allocating additional memory [see Fig. 5.1], (b) the cost of implementation decreases to half. In addition to that, the lifting method paves a way to build the wavelet transform that can map integer-to-integers [12] which is important for easier hardware implementation and sometimes helpful for lossless implementation. Moreover, the integer conversion of the lifting filter coefficients with limited precision [10] leads to lower cost implementation. So, the lifting implementation with overall integer mapping (integer-to-integer mapping and integer conversion of the coefficients) is a good choice for implementation low-cost and sometimes lossless implementation.

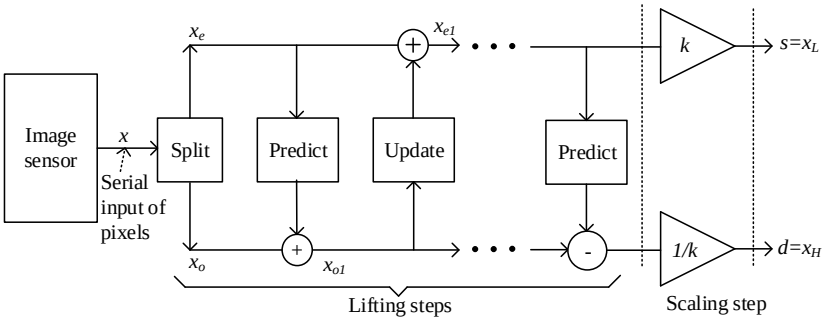
But the open issue with integer mapping is scaling steps with coefficient *k*. If the value of this coefficient is not 1, which is the case for many lifting wavelets including Daubechies orthogonal wavelets (except Haar wavelet), biorthogonal wavelet CDF-9/7 wavelets, the integer mapped wavelet remains lossy[9], [10], [12] because the rounder results of a multiplication by non-one value cannot be recovered by division by the same number. Mathematically, if the number is x and $\|xk\| = s$, $\|s/k\| \neq x$ when $k \neq 1$; therefore, the number x is not likely to be recovered.

There are few solutions proposed so far to overcome this issue. Firstly, Daubechies[9] proposed that scaling can be replaced with four lifting steps. The issue in this lifted scaling method is that the extra lifting steps increase the implementation cost and each extra lifting step adds few extra errors when the limited precision of the coefficient is used or integer-to-integer mapping is used. The second proposal was to remove the scaling step [10] since there is no effect of having scaling on the coding gain of the transform, one of the performance metrics. This method is less costly compared to the first one and performs better since it avoids adding error in the additional steps. The issue is the removal of scaling steps has an effect on the entropy of the

transform, another important performance metric for any transform. When the scaling step is removed, the entropy may remain the same or may increase depending on how the lifting steps are manually factorized. Since the non-unique factorization is manual work, this method is worth for higher order costly filter like Daubehies-8[10]. Since the removal of the scaling step does not automatically guarantee an equivalent transform and needs manual work to try different factorizations, we need to have a common solution which works for all orthogonal and biorthogonal wavelets.



(a)



(b)

Fig. 5.1 Implementation of wavelet transform for image processing [10]: (a) standard convolution or matrix based method, (b) lifting base method. Standard method requires additional buffer memory.

This research provides a common or general solution for the issues. This paper shows a factorization technique which leads to lifting wavelet transform where $k = 1$ is all cases without

increasing the number of lifting steps. Hence, we do not need to add three additional lifting steps or we do not need to do manual factorization to try different option to find where we can remove the lifting steps.

Since the last scaling step is not being used anymore, this method provides a completely lossless reconstruction for the integer-mapped wavelet transform which is already low-cost.

The rest of the paper is organized as follows. Section 5.2 provides a brief description of the conventional way of factorization and integer mapping so that we can compare and show a way to convert the traditional factorization into the proposed one. Section 5.3 describes the theory of the proposed factorization and shows how we can convert a polyphase matrix into lifting steps factors without having any scaling step. Section 5.4 applies the theory to generate factorization of some popular orthogonal and biorthogonal wavelet and shows both traditional and proposed factorization in a table. Section 5.5 provides some results which satisfy the theory. Finally, section 5.6 concludes the work.

5.2 Lifting Wavelet Transform

The discrete wavelet transforms (DWT) is based on small wavelets with limited duration to analyze both the frequency and time component of a signal. For the computation of the DWT, in addition to the traditional convolution or matrix based implementation [as shown in Fig. 5.1 (a)], Daubechies[9] and Sweldons[13] showed that an alternative structure of wavelet transform can be built from any orthogonal and biorthogonal filters by employing factorization of a polyphase matrix which is constructed from the traditional high and low-pass filter of the transform. This new method of implementation is known as the lifting scheme. As shown in Fig. 5.1. (b), the lifting scheme contains many predict and update lifting steps and at last scaling steps. As discussed in Section 5.1, our aim of this research is to build a new factorization so that we can have a scaling step where k=1.

5.2.1 The conventional way of factorization

However, here is the summary of how the factorization is done in the traditional way. It starts with a well-known set of filters (h,g) of any orthogonal or biorthogonal wavelets where h is

low-pass and *g* is a high-pass filter. These filters are split into even and odd sequences which are subscripted by *e* and *o* respectively. Then the polyphase matrix is assembled as:

$$P(\mathrm{z}) = \tilde{P}(\mathrm{z}) = \begin{bmatrix} h_e(\mathrm{z}) & g_e(z) \\ h_o(\mathrm{z}) & g_o(z) \end{bmatrix} \tag{5.1}$$

Here $h_e(z)$ and $h_o(z)$ are Laurent polynomials.

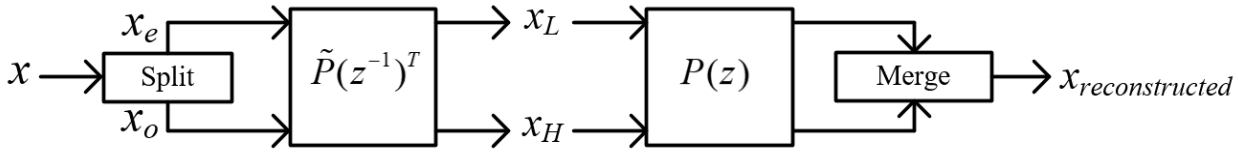


Fig. 5.2 Polyphase representation of wavelet transforms[9].

Conventionally, we start factorizing the first column of the polyphase matrix. Since both $h_e(z)$ and $h_o(z)$ are Laurent polynomials, we employ a successive division approach using the Euclidean algorithm for Laurent polynomials. This results in the following matrix decomposition (where *k* is the greatest common divisor-GCD):

$$\begin{bmatrix} h_e(z) \\ h_o(z) \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k \\ 0 \end{bmatrix} \tag{5.2}$$

The value of n is assumed even. Details of the factorization can be found in [9].

Now, we can find a complementary polyphase filter g^0 where another polyphase matrix $\hat{P}(z)$ is as follows:

$$\hat{P}(z) = \begin{bmatrix} h_e(\mathrm{z}) & \hat{g}_e(z) \\ h_o(\mathrm{z}) & \hat{g}_o(z) \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}$$

A polynomial $S(z)$ can be found such that

$$P(z) = \hat{P}(z) \begin{bmatrix} 1 & S(z) \\ 0 & 1 \end{bmatrix}$$

The final factorization matrix is given as:

$$P(z)=\prod_{i=1}^{n/2}\begin{bmatrix}1 & q_{2i-1}(z)\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ q_{2i}(z) & 1\end{bmatrix}\begin{bmatrix}1 & s(z)\\ 0 & 1\end{bmatrix}\begin{bmatrix}k & 0\\ 0 & 1/k\end{bmatrix} \tag{5.3}$$

where $s(z)=k^2S(z)$.

The total number of factors is n+2. The factorization matrix can be written as in the following format:

$$P(z)=\prod_{i=1}^{n/2+1}\begin{bmatrix}1 & p_i(z)\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ u_i(z) & 1\end{bmatrix}\begin{bmatrix}k & 0\\ 0 & 1/k\end{bmatrix}$$

Here $p_i(z)=q_{2i-1}(z)$ and $u_i(z)=q_{2i}(z)$ when $i\le\frac{n}{2}$

$p_{n/2+1}(z)=s(z)$ and $u_{n/2+1}(z)=0$

So, the forward transform matrix is

$$\tilde{P}(z^{-1})^T=\begin{bmatrix}k & 0\\ 0 & 1/k\end{bmatrix}\prod_{i=1}^{n/2+1}\begin{bmatrix}1 & u_i(z^{-1})\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ p_i(z^{-1}) & 1\end{bmatrix} \tag{5.4}$$

If the input sequence to the transform is x, and even and odd components are x_e= $x_{e.0}$ and x_o= $x_{o.0}$ respectively,

$$\begin{bmatrix}x_L\\ x_H\end{bmatrix}=\tilde{P}(1/z)^T\begin{bmatrix}x_e\\ x_o\end{bmatrix}=\tilde{P}(1/z)^T\begin{bmatrix}x_{e.0}\\ x_{o.0}\end{bmatrix} \tag{5.5}$$

Following forward transform can be found from (5.4) and (5.5):

$$x_{o.i}=p_i(z^{-1})x_{e.(i-1)}+x_{o.(i-1)}\ \text{for}\ 1\le i\le n/2+1 \tag{5.6}$$

$$x_{e.i}=x_{e.(i-1)}+u_i(z^{-1})x_{o.i}\ \text{for}\ 1\le i\le n/2+1 \tag{5.7}$$

$$x_L=s=kx_{e.(n/2+1)} \tag{5.8}$$

$$x_H=d=\frac{1}{k}x_{o.(n/2+1)} \tag{5.9}$$

The steps that are shown in the first two equations (5.6) and (5.7) are known as the lifting steps while the last multiplication steps, which are shown in (5.8) and (5.9), are known as scaling steps. If we implement this equation, that will result in a diagram like Fig. 5.1 (b) where predict step = $p_i(z^{-1})$ and update step = $u_i(z^{-1})$. The reverse transform can be found by the same equation if we calculate backward.

5.2.2 Integer mapping

When integer-to-integer mapping is performed on the lifting wavelets, the lifting steps presented at (5.6)-(5.9) become

$$x_{o.i} = \left\lfloor p_i(z^{-1})x_{e.(i-1)} + \frac{1}{2} \right\rfloor + x_{o.(i-1)} \text{ or } x_{o.i} = \left\| p_i(z^{-1})x_{e.(i-1)} \right\| + x_{o.(i-1)} \tag{5.10}$$

$$x_{e.i} = x_{e.(i-1)} + \left\| u_i(z^{-1})x_{o.i} \right\| \tag{5.11}$$

$$x_L = s = \left\| k x_{e.(n/2+1)} \right\| \tag{5.12}$$

$$x_H = d = \left\| \frac{1}{k} x_{o.(n/2+1)} \right\| \tag{5.13}$$

Here, $\lfloor * \rfloor$ and $\|*\|$ denotes the symbol for floor and rounding-to-integer operation respectfully; though $\lfloor *+1/2 \rfloor$ and $\|*\|$ are similar, we prefer $\|*\|$ since $\lfloor *+1/2 \rfloor$ (used in [12]) requires an additional addition with a non-integer number and $\|*\|$ allows to explore different techniques of rounding; that is why this author used $\|*\|$ in the previously published work[10] . The first two lifting steps using integer-to-integer are completely lossless and the values are recoverable with back-calculation while the last two steps, i.e. the scaling steps are not recoverable as explained in Section 5.1.

Now, integer conversion of the lifting filter coefficients with limited precision is performed by $A = \left\| 2^b a \right\|$ where a is one of the coefficients in the Laurent polynomial. For example, if $p_i(z^{-1}) = az^{-1} + b$ where *a*, *b*, *c* are non-integer irrational numbers, the new Laurent polynomial with integer coefficients with binary *b*-bit precisions, $P_i(z^{-1}) = Az^{-1} + B$ where $A = \left\| 2^b a \right\|$ and $B = \left\| 2^b b \right\|$. So, when overall integer mapping for the forward transform is as follows:

$$x_{o.i} = \left\| (P_i(z^{-1})x_{e.(i-1)}) / 2^b \right\| + x_{o.(i-1)} \tag{5.14}$$

$$x_{e.i} = x_{e.(i-1)} + \left\| (U_i(z^{-1})x_{o.i}) / 2^b \right\| \tag{5.15}$$

$$x_L = s = \left\| (K x_{e.(n/2+1)}) / 2^b \right\| \tag{5.16}$$

$$x_H = d = \left\| (K' x_{o.(n/2+1)}) / 2^b \right\| \text{ where } K' = \left\| 2^b / k \right\| \tag{5.17}$$

The advantage of integer conversion of the lifting coefficients is of two-folds: (a) multiplication with integer coefficient is easier and low-cost with Booth algorithm[14], [15] and (b) we found in the extensive experiments (shown in [10]) that it is possible to find a limited value of *b* for which the performance of a transform (entropy and transform coding gain) remain similar to the double-precision value. Limited value of *b* results in smaller coefficients and low-cost calculation.

However, the disadvantage of the above integer-to-integer mapping remains. As explained in Section 5.1, the scaling step remains lossy.

5.3 Proposed Factorization Theory

This section will be a tutorial in nature. This section describes how we can convert a standard discrete wavelet filter into the lifting so that there is no effective scaling step anymore, i.e. the value of *k* is 1 and integer-to-integer mapping or overall integer-mapping become completely lossless. We tried to use similar notation as much as possible as used in [9] so that the experienced reader can relate and may find it relevant.

5.3.1 Factorization of the first column of the polyphase matrix

Like the conventional factorization, it starts with the polyphase matrix $P(z)$ shown in (5.1). At the first step, we will find the factorization of the first column of the polyphase matrix $\begin{bmatrix} h_e(z) \\ h_o(z) \end{bmatrix}$. To achieve that, the conventional way finds the greatest common divisor (GCD) of $h_e(z)$ and $h_o(z)$ using the Euclidean algorithm for Laurent polynomials. We will modify the steps in the algorithm to make sure that the GCD is always 1.

Euclidean algorithm for polynomial[9], [16] states that if there are two polynomials $a(z)$ and $b(z)$ with $|a(z)| \geq |b(z)|$, let $a_0(z) = a(z)$ and $b_0(z) = b(z)$ and repeat the following recursive steps for i=0,1,2,..,n.

$$a_{i+1}(z) = b_i(z) \tag{5.18}$$

$$b_{i+1}(z) = \text{remainder of } \frac{a_i(z)}{b_i(z)} = a_i(z) - b_i(z)q_{i+1}(z) \tag{5.19}$$

In matrix form: $$\begin{bmatrix} a_{i+1}(z) \\ b_{i+1}(z) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -q_i(z) \end{bmatrix} \begin{bmatrix} a_i(z) \\ b_i(z) \end{bmatrix} \tag{5.20}$$

Here, the quotient, $q_{i+1} = \left\lfloor \frac{a_i(z)}{b_i(z)} \right\rfloor$

There will be a step $i=n-1$ for which $b_{i+1}(z) = b_n(z) = 0$ when $|b_{i+1}(z)| < |b_i(z)|$ in each step. In that case, $a_{i+1}(z) = a_n(z)$ is the greatest common divisor (GCD) of $a(z)$ and $b(z)$. Let this GCD be k. So using (5.18), we get

$$a_n(z) = b_{n-1}(z) = k \tag{5.21}$$

From the above matrix shown in (5.20),

$$\begin{bmatrix} k \\ 0 \end{bmatrix} = \prod_{i=n}^{1} \begin{bmatrix} 0 & 1 \\ 1 & -q_i(z) \end{bmatrix} \begin{bmatrix} a_o(z) \\ b_o(z) \end{bmatrix}$$

Or, $$\begin{bmatrix} a_0(z) \\ b_0(z) \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k \\ 0 \end{bmatrix} \tag{5.22}$$

Since, in the step $i=n-1$,

$$b_n(z) = a_{n-1}(z) - b_{n-1}(z)q_n(z) = 0$$

$$q_n(z) = \frac{a_{n-1}(z)}{b_{n-1}(z)} \tag{5.23}$$

If we use the above algorithm to find GCD of $h_e(z)$ and $h_o(z)$ i.e. if $a_0(z) = h_e(z)$ and $b_0(z) = h_o(z)$, (5.22) can be rewritten as

$$\begin{bmatrix} h_e(z) \\ h_o(z) \end{bmatrix} = \prod_{i=1}^{n} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k \\ 0 \end{bmatrix} \tag{5.24}$$

This is in details how the factorization of the first column of the polyphase matrix is done in the conventional factorization. However, the GCD, k is not necessarily 1 and the factorization is not unique.

In our proposed factorization, in step *i*=*n*-1=*m*, we make sure $b_n(z) = 1$ instead of $b_n(z) = 0$. From equation (5.19),

$$b_n(z) = a_{n-1}(z) - b_{n-1}(z)q_n(z) = 1 \tag{5.25}$$

$$\Rightarrow q_n(z) = \frac{a_{n-1}(z)}{b_{n-1}(z)} - \frac{1}{b_{n-1}(z)}$$

Since $b_{n-1}(z) = a_n(z) = k$ as shown in (5.21), from the above equation (5.25), we get the new value of the *n*-th quotient as

$$q_n(z) = \frac{a_{n-1}(z)}{b_{n-1}(z)} - \frac{1}{k} \tag{5.26}$$

Comparing the conventional value of $q_n(z)$ in (5.23) and proposed value in (5.26),

Proposed $q_n(z)$ = conventional $q_n(z) - \frac{1}{k}$

In short, the new n-th quotient,

$$q_{n.pro}(z) = q_n(z) - \frac{1}{k} \tag{5.27}$$

We should also mention that the value of $q_1(z), q_2(z), ..., q_{n-1}(z)$ will remain the same as the conventional factorization.

In the next step, where *i*=*n*, we will make sure that $b_{i+1}(z) = b_{n+1}(z) = 0$.

Using the equations (5.18) and (5.19),

$a_{n+1}(z) = b_n(z) = 1$ [as per (5.25)]

$b_{n+1}(z) = a_n(z) - b_n(z)q_{n+1}(z) = a_n(z) - 1 \times q_{n+1}(z)$

Since $b_{n+1}(z) = 0$, $q_{n+1}(z) = a_n(z) = k$.

So, proposed $q_{(n+1).pro}(z) = k$ (5.28)

Now we are intentionally making the remainder zero in the step *i*=*n* where, in the conventional Euclidian way, the remainder becomes zero in the step *i*=*n*-1 in Euclidian algorithm equations (5.18) and (5.19). So, now the greatest common divisor (GCD) would be $a_{i+2}(z) = a_{n+1}(z)$ which is clearly 1.

That is why, from this alternative way, common divisor would be

$$k_{pro} = a_{n+1}(z) = 1 \tag{5.29}$$

This is not the greatest common divisor, but common divisor of any two real numbers or Laurent polynomials.

Therefore, with the matrix form of the Euclidian algorithm (5.20), using the new quotient values from (5.27), (5.28) and (5.29), we can write,

$$\begin{bmatrix} a_{n+1}(z) \\ b_{n+1}(z) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -q_{(n+1).pro}(z) \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -q_{n.pro}(z) \end{bmatrix} \prod_{i=n-1}^{1} \begin{bmatrix} 0 & 1 \\ 1 & -q_i(z) \end{bmatrix} \begin{bmatrix} a_0(z) \\ b_0(z) \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -k \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -q_n(z) + \frac{1}{k} \end{bmatrix} \prod_{i=n-1}^{1} \begin{bmatrix} 0 & 1 \\ 1 & -q_i(z) \end{bmatrix} \begin{bmatrix} a_0(z) \\ b_0(z) \end{bmatrix}$$

Reorganizing this equation,

$$\begin{bmatrix} h_e(z) \\ h_o(z) \end{bmatrix} = \begin{bmatrix} a_0(z) \\ b_0(z) \end{bmatrix} = \prod_{i=1}^{n-1} \left(\begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \right) \begin{bmatrix} q_n(z) - \frac{1}{k} & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{5.30}$$

The equation (5.30) is our factorization of the first column of the polyphase matrix while (5.24) is the conventional factorization of the same.

5.3.2 Factorization of full polyphase matrix

We have factorized *h* filter in the equation (5.30). The next step would be factorizing the polyphase matrix *P*(*z*) shown in (5.1).

The determinant of a polyphase matrix should be 1 if the associated filter pair must be complementary. When we have a filter *h* as expressed in the form of $\begin{bmatrix} h_e(z) \\ h_o(z) \end{bmatrix}$, if we have to find a complementary filter $\hat{g}$ expressed in the form of $\begin{bmatrix} \hat{g}_e(z) \\ \hat{g}_o(z) \end{bmatrix}$, we have to make sure that the determinant of the associated polyphase matrix $\hat{P}(z)$ is 1. Therefore,

$$\hat{P}(z) = \begin{bmatrix} h_e(z) & \hat{g}_e(z) \\ h_o(z) & \hat{g}_o(z) \end{bmatrix}$$

$$= \prod_{i=1}^{n-1} \left(\begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \right) \begin{bmatrix} q_n(z) - \frac{1}{k} & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} k & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{5.31}$$

Since like the conventional factorization, *n* is assumed to be even, we added $\begin{bmatrix} 0 \\ -1 \end{bmatrix}$ in the last matrix of (5.30) to make the determinant 1. If n is odd, we can use $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ instead. From (5.31), we can find,

$$\hat{P}(z) = \prod_{i=1}^{(n-2)/2} \left(\begin{bmatrix} 1 & q_{2i-1}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_{2i}(z) & 1 \end{bmatrix} \right) \begin{bmatrix} 1 & q_{n-1}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_n(z) - \frac{1}{k} & 1 \end{bmatrix} \begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{5.32}$$

We can recover the filter *g* from $\hat{g}$ with one lifting stage as practiced in [9]:

$$P(z) = \hat{P}(z) \begin{bmatrix} 1 & s(z) \\ 0 & 1 \end{bmatrix} \tag{5.33}$$

From the equation (5.33), it is possible to calculate the value of *s*(*z*), since we already know the value of *P*(*z*) from the equation (5.1) and *P*(*z*) from (5.32).

Calculation using the equation (5.32) and (5.33) provides us

$$P(z) = \prod_{i=1}^{(n-2)/2} \left(\begin{bmatrix} 1 & q_{2i-1}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_{2i}(z) & 1 \end{bmatrix} \right) \begin{bmatrix} 1 & q_{n-1}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_n(z) - \frac{1}{k} & 1 \end{bmatrix} \begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -s(z) & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{5.34}$$

Equation (5.34) shows the proposed final factorization of the polyphase matrix. It should be mentioned that the value of $q_1, q_2, q_3 \ldots q_n$ and *k* can be kept similar to what we have in the conventional factorization. This makes the conversion of existing factorization to the proposed one easier.

Moreover, as we can see, the number of factors is effectively the same (n+2) as the conventional transform; since the last factor $\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$ does not cost any computation except a negation operation.

5.3.3 Implementation with integer mapping

Now, the factorization matrix can be written as in the following format:

$$P(z) = \prod_{i=1}^{n/2+1} \begin{bmatrix} 1 & p_i(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ u_i(z) & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{5.35}$$

$$\tilde{P}(z^{-1})^T = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \prod_{i=1}^{n/2+1} \begin{bmatrix} 1 & u_i(z^{-1}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ p_i(z^{-1}) & 1 \end{bmatrix} \tag{5.36}$$

As discussed in section 5.2, if an input sequence *x* is split into even and odd components, the proposed forward transform would be

$$x_{o.i} = p_i(z^{-1}) x_{e.(i-1)} + x_{o.(i-1)} \text{ for } 1 \le i \le n/2+1$$

$$x_{e.i} = x_{e.(i-1)} + u_i(z^{-1}) x_{o.i} \text{ for } 1 \le i \le n/2+1$$

$$x_L = s = -x_{o.(n/2+1)}$$

$$x_H = d = x_{e.(n/2+1)}$$

The reverse transform can be formed by the calculation in reverse direction:

$$x_{e.(n/2+1)} = x_H$$

$$x_{o.(n/2+1)} = -x_L$$

$$x_{e.(i-1)} = x_{ei} - u_i(z^{-1}) x_{o.i} \text{ from i = } n/2+1 \text{to } 1$$

$$x_{o.(i-1)} = x_{oi} - p_i(z^{-1}) x_{e.(i-1)} \text{ from} i = n/2+1 \text{to} 1$$

With the overall integer mapping, as shown in (5.14)-(5.17) for the conventional mapping, the forward transform is as follows:

$$x_{o.i} = \left\| (P_i(z^{-1}) x_{e.(i-1)}) / 2^b \right\| + x_{o.(i-1)}$$

$$x_{e.i} = x_{e.(i-1)} + \left\| (U_i(z^{-1}) x_{o.i}) / 2^b \right\|$$

$x_L = s = -x_{o.(n/2+1)}$

$x_H = d = x_{e.(n/2+1)}$

The reverse transform can also be found in a similar way shown above.

5.4 Application of Proposed Theory of Factorization

The previous section provides the theory of how a factorization of the wavelet can be achieved so that the lossless reconstruction is possible. This section will present a number of factorizations performed using the proposed theory.

Using the proposed theory, we can take an existing factorization and convert them into the proposed factorization. To do that, we can compare the factorization with the conventional method shown in (5.4) and the proposed factorization in (5.34). We can observe that first (*n*-1) factors are the same where n is the number of 2×2 matrix factors in the equation (5.2). The rest factors can be found from (5.34). Details are as follows:

a) For $\begin{bmatrix} 1 & 0 \\ q_n(z) - 1/k & 1 \end{bmatrix}$ and $\begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix}$, $q_n(z)$ and k can be found from conventional *n*-th factor and last factor.

b) For $\begin{bmatrix} 1 & 0 \\ -s(z) & 1 \end{bmatrix}$, $s(z)$ can be found using (5.33).

Table 5.1 lists several conversions as examples. As theory shows, the interested reader should be able to convert the conventional factorization of other wavelets using the above technique or factorize the wavelets from scratch using the method shown in Section 5.3. It should be mentioned that the integer implementation can be performed by the method shown in Section 5.3.3.

Table 5.1 Example Conversion from Conventional to Proposed Factorization

Wavelet name	Conventional $P(z)$ or $\tilde{P}(z)$	Proposed $P(z)$ or $\tilde{P}(z)$
D4	$\begin{bmatrix}1 & c_1\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_2 & 1\end{bmatrix}\begin{bmatrix}1 & c_3\\0 & 1\end{bmatrix}\begin{bmatrix}k & 0\\0 & 1/k\end{bmatrix}$ $c_1 = -1.7320508076$ $c_2 = 0.43301270189 - 0.066987298108z^{-1}$ $c_3 = 1$, $k = 1.9318516526$	$\begin{bmatrix}1 & c_1\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_2 & 1\end{bmatrix}\begin{bmatrix}1 & c_3\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_4 & 1\end{bmatrix}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix}$ $c_1 = -1.7320508076$ $c_2 = -0.084625388313 - 0.066987298108z^{-1}$ $c_3 = 1.9318516526$, $c_4 = -0.26794919243z - 0.51763809021$
D6	$\begin{bmatrix}1 & 0\\c_1 & 1\end{bmatrix}\begin{bmatrix}1 & c_2\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_3 & 1\end{bmatrix}\begin{bmatrix}1 & c_4\\0 & 1\end{bmatrix}\begin{bmatrix}k & 0\\0 & 1/k\end{bmatrix}$ $c_1 = 2.4254972439$ $c_2 = 0.079339456159\ z - 0.35238765768$ $c_3 = -2.8953474544z^{-1} + 0.56141490919z^{-2}$ $c_4 = -0.019750529237z^2$, $k = 0.43187999149$	$\begin{bmatrix}1 & 0\\c_1 & 1\end{bmatrix}\begin{bmatrix}1 & c_2\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_3 & 1\end{bmatrix}\begin{bmatrix}1 & c_4\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_4 & 1\end{bmatrix}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix}$ $c_1 = 2.4254972439$ $c_2 = 0.079339456159\ z - 0.35238765768$ $c_3 = -2.3154580432 - 2.8953474544z^{-1} + 0.56141490919z^{-2}$ $c_4 = 0.43187999149$, $c_5 = 0.10588941994z^2 - 2.3154580433$
D8	$\begin{bmatrix}1 & c_1\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_2 & 1\end{bmatrix}\begin{bmatrix}1 & c_3\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_4 & 1\end{bmatrix}\begin{bmatrix}1 & c_5\\0 & 1\end{bmatrix}\begin{bmatrix}k & 0\\0 & 1/k\end{bmatrix}$ $c_1 = -3.1029314859$ $c_2 = 0.291953126 - 0.076300086572z^{-1}$ $c_3 = -1.6625283534 + 5.199491573z^2$ $c_4 = 0.037892748126z^{-2} - 0.0067223726328z^{-3}$ $c_5 = 0.31410649341z^3$, $k = 2.6131183698$	$\begin{bmatrix}1 & c_1\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_2 & 1\end{bmatrix}\begin{bmatrix}1 & c_3\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_4 & 1\end{bmatrix}\begin{bmatrix}1 & c_5\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_6 & 1\end{bmatrix}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix}$ $c_1 = -3.1029314859$ $c_2 = 0.291953126 - 0.076300086572z^{-1}$ $c_3 = -1.6625283534 + 5.199491573z^2$ $c_4 = -0.38268453949 + 0.037892748126z^{-2} - 0.0067223726328z^{-3}$ $c_5 = 2.6131183698$, $c_6 = -0.046000097113 - 0.38268453949z^3$
D10	$\begin{bmatrix}1 & 0\\c_1 & 1\end{bmatrix}\begin{bmatrix}1 & c_2\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_3 & 1\end{bmatrix}\begin{bmatrix}1 & c_4\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_5 & 1\end{bmatrix}\begin{bmatrix}1 & c_6\\0 & 1\end{bmatrix}\begin{bmatrix}k & 0\\0 & 1/k\end{bmatrix}$ $c_1 = 3.7715192117$, $c_2 = 0.069884292321 - 0.24772929136z$ $c_3 = -7.5975797362z^{-1} + 3.0336897918z^{-2}$ $c_4 = 0.015799323692z^3 - 0.050396352639z^2$ $c_5 = -1.1031463286z^{-3} + 0.17257255569z^{-4}$ $c_6 = -0.0025143438267z^4$, $k = 0.34738904011$	$\begin{bmatrix}1 & 0\\c_1 & 1\end{bmatrix}\begin{bmatrix}1 & c_2\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_3 & 1\end{bmatrix}\begin{bmatrix}1 & c_4\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_5 & 1\end{bmatrix}\begin{bmatrix}1 & c_6\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_7 & 1\end{bmatrix}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix}$ $c_1 = 3.7715192117$, $c_2 = 0.069884292321 - 0.24772929136z$ $c_3 = -7.5975797362z^{-1} + 3.0336897918z^{-2}$ $c_4 = 0.015799323692z^3 - 0.050396352639z^2$ $c_5 = -2.8786170101 - 1.1031463286z^{-3} + 0.17257255569z^{-4}$ $c_6 = 0.34738904011$, $c_7 = 0.020834948928z^4 - 2.8786170115$
9/7 filter (bior4.4)	$\begin{bmatrix}1 & c_1\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_2 & 1\end{bmatrix}\begin{bmatrix}1 & c_3\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_4 & 1\end{bmatrix}\begin{bmatrix}k & 0\\0 & 1/k\end{bmatrix}$ $c_1 = -1.5861343421(1+z^{-1})$, $c_2 = -0.052980118573(1+z)$ $c_3 = 0.88291107553(1+z^{-1})$ $c_4 = 0.44350685204z + 0.44350685204$ $k = 1.1496043989$	$\begin{bmatrix}1 & c_1\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_2 & 1\end{bmatrix}\begin{bmatrix}1 & c_3\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_4 & 1\end{bmatrix}\begin{bmatrix}1 & c_5\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_6 & 1\end{bmatrix}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix}$ $c_1 = -1.5861343421(1+z^{-1})$, $c_2 = -0.052980118573(1+z)$ $c_3 = 0.88291107553(1+z^{-1})$ $c_4 = 0.44350685204z - 0.42635759958$ $c_5 = 1.1496043989, c_6 = -0.86986445162$
CDF4.2	$\begin{bmatrix}1 & 0\\c_1 & 1\end{bmatrix}\begin{bmatrix}1 & c_2\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_3 & 1\end{bmatrix}\begin{bmatrix}1 & c_4\\0 & 1\end{bmatrix}\begin{bmatrix}k & 0\\0 & 1/k\end{bmatrix}$ $c_1 = -0.25z - 1.25$, $c_2 = -0.23076923077z + 1$ $c_3 = -3.5208333333 + 0.8125z^{-1}$ $c_4 = 0.28402366864, k = -0.65271395186$	$\begin{bmatrix}1 & 0\\c_1 & 1\end{bmatrix}\begin{bmatrix}1 & c_2\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_3 & 1\end{bmatrix}\begin{bmatrix}1 & c_4\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_5 & 1\end{bmatrix}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix}$ $c_1 = -0.25z - 1.25$, $c_2 = -0.23076923077z + 1$ $c_3 = -1.9887686408 + 0.8125z^{-1}$ $c_4 = -0.65271395186, c_5 = 0.8653980259$
CDF5.1	$\begin{bmatrix}1 & 0\\c_1 & 1\end{bmatrix}\begin{bmatrix}1 & c_2\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_3 & 1\end{bmatrix}\begin{bmatrix}1 & c_4\\0 & 1\end{bmatrix}\begin{bmatrix}k & 0\\0 & 1/k\end{bmatrix}$ $c_1 = -5$, $c_2 = 0.025z + 0.20833333333$ $c_3 = -37.5 + 22.5z^{-1}$ $c_4 = 0.013333333333, k = -0.28284271247$	$\begin{bmatrix}1 & 0\\c_1 & 1\end{bmatrix}\begin{bmatrix}1 & c_2\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_3 & 1\end{bmatrix}\begin{bmatrix}1 & c_4\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\c_5 & 1\end{bmatrix}\begin{bmatrix}0 & -1\\1 & 0\end{bmatrix}$ $c_1 = -5$, $c_2 = 0.025z + 0.20833333333$ $c_3 = -33.964466094 + 22.5z^{-1}$ $c_4 = -0.28284271247$, $c_5 = 3.3688672393$

5.5 Performance Assessment

Now we know how to convert the existing factorizations in our proposed format or factorize from scratch. The benefits are already mentioned partly in Section 5.1 and 5.3.2. The overall advantages of the proposed factorizations are:

1) This factorization does not require any scaling steps. As already discussed, the scaling step is the main cause of the reconstruction error in the integer-to-integer mapping of the most wavelet transforms. So, the proposed factorization is completely lossless if implemented with either integer-to-integer mapping or overall integer-mapping.

2) The factorization does not usually require any additional number of lifting steps as well; the total number of scaling steps or stages is the same as the conventional factorization. Generally, integer-to-integer mapping of each lifting stage adds some errors, and the result deviates from the result obtained without any mapping. Each additional stage adds new errors. So, in the proposed factorization, while the scaling steps are not used, the effect of the scaling steps is incorporated in the previous stages and the overall number of scaling stages remains the same.

This is unlike the previous solution where the scaling step is suggested to replace with the additional four lifting steps[9] or removal of scaling steps completely[10] which requires manual work to try many factorization options to select the best one.

3) Another objective of this factorization is to have low-cost implementation as well in addition to being lossless. Integer conversion of the coefficients with limited precision allows implementing with lower cost. We need to ensure whether the limited precision of coefficients in the proposed factorization does not harm much performance of the transform.

Table 5.2 shows the reconstruction error in different wavelets shown in Table 5.1 when implemented with integer-to-integer mapping shown in the equation (5.11)-(5.14); a benchmark ‘Lena’ image is used as the input and this same image will be used for all other results. The table also shows results with the integer-conversion of the coefficients as shown in the equation

(5.15)-(5.18). As we can see, the wavelet transform algorithm found using the proposed factorization technique is completely lossless.

Now we would like to compare our approach with the approach where scaling steps is replaced with the following four lifting steps[9]:

$$P(z)=\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}=\begin{bmatrix} 1 & k-k^2 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ -1/k & 1 \end{bmatrix}\begin{bmatrix} 1 & k-1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$
$$P(z^{-1})^T=\begin{bmatrix} k & 0 \\ 0 & 1/k \end{bmatrix}=\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ k-1 & 1 \end{bmatrix}\begin{bmatrix} 1 & -1/k \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ k-k^2 & 1 \end{bmatrix} \quad (5.37)$$

Table 5.3 shows that comparison. This table lists the entropy of the transformed results after single-level decomposition of the image 'Lena' for the wavelets found from the proposed factorization and conventional replacement of the scaling steps. We can see that while the quality of the transform (i.e. entropy) with integer-mapping remains similar, the no of lifting stages is comparatively lower in the proposed approach. This translates into the decrease of the cost and lower number of flip-flops (in most case, the number of required flops decreases by 3) which are needed in each stage; which in turn decrease the latency by the same number of clock cycles.

Fig. 5.3 shows the entropy with different precisions (*b*) of the coefficients for the different wavelets with the proposed factorization. As we see, while the entropies decrease with higher precisions initially, it does not decrease anymore after 7 or 8-bit precision and remains almost similar. This information verifies that lifting wavelets found from the proposed factorization allows the usage of limited precisions of the coefficients and exhibits similar performance with a certain limited precision. This is important for low-cost implementation because the usage of the coefficients with double precision would make the implementation quite expensive.

An important metric to determine the quality of any transform is the transform coding gain. If the input sequence of the transform is x and the transformed output sequences are $y_0, y_1, \ldots y_N$, the transform coding gain is given by[17], [18]:

$$G_T=\frac{\sigma_x^{\,2}}{\left(\prod_{n=0}^{N-1}\sigma_{y_n}^2\right)^{\frac{1}{N}}}=\frac{\sigma_x^{\,2}}{\left(\sigma_{y_0}^2\sigma_{y_1}^2\right)^{\frac{1}{2}}}$$

where $\sigma_x^{\,2}$ and $\sigma_{y_n}^2$ represent the variance of the input sequence x and the output sequence y_n.

Table 5.2 Wavelet reconstruction error with integer-to-integer mapping

Wavelet	RMSE in the conventional factorization		RMSE with proposed ones
Integer conversion of coefficients	Without	with (8 bits precision)	Without/with (any precisions)
D4	0.9908	0.9837	0
D6	1.2166	1.2200	0
D8	1.5483	1.5533	0
D10	1.8458	1.8443	0
CDF9/7	1.1616	1.1561	0
CDF4.2	1.9930	1.9728	0
CDF5.1	3.3337	3.5445	0

Table 5.3 Comparison with existing lossless factorization

Wavelet	Scaling step replacement with eqn. (5.37) [9]		This approach	
	Entropy*	No of lifting stages	Entropy*	No of lifting stages
D4	6.0687	7	6.0624	4
D6	6.0678	8	6.0744	5
D8	6.0456	9	6.0210	6
D10	6.1285	10	6.3218	7
CDF9/7	5.9843	8	6.0030	6
CDF4.2	5.9960	8	5.9114	5
CDF5.1	6.8863	8	6.4000	5

*Entropy with integer-to-integer mapping and 8-bit precision in the integer conversion of the coefficients

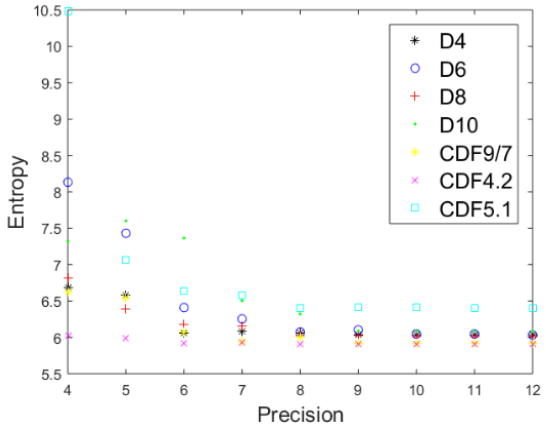


Fig. 5.3 Entropy for the wavelets found from the proposed factorization for the different precisions of the coefficients in integer conversion

Table 5.4 presents the transform coding gain of the different wavelet transform with 'Lena' image, after the single level of decomposition. It shows both traditional as well as the proposed implementation which ensures that the proposed implementation does not harm the quality of the transform. We should mention that, for Table 5.4, we do not use any integer-to-integer mapping and we use the double precision values of the coefficients for the computation of the wavelet transforms of the image.

Table 5.4 Transform Coding Gain

Wavelet	Transform coding gain		
	Standard convolution based*	Lifting based methods	
		Conventional	Proposed
D4	16.27	18.79	18.79
D6	14.13	20.13	20.13
D8	20.07	20.89	20.89
D10	20.03	21.84	21.84
CDF9/7	19.68	24.28	24.28
CDF4.2	33.20	34.84	34.84
CDF5.1	26.79	27.39	27.39

*Using MATLAB dwt2 function

In addition to the theoretical analysis and experimental results of the performance metrics, it is important to apply in an application and compare it. Among many applications achievable using wavelet transform, the most common one is image compression. We would like to compare the lossless compression rate of the wavelet transform achieved through our proposed algorithm with the costly alternative (shown in Table 5.3[9]) which is scaling step replacement with the equation (5.37). As we already indicated in Table 5.3 that the proposed technique maintains the performance of the transforms with the advantage of using a lower number of stages, Table 5.5 and 5.6 show the similar trends for the lossless image compression rate. In order to make a fair comparison, for both cases, we applied integer-to-integer mapping and used 8-binary bit precision in the integer conversion of the coefficients. To compute the lossless compression rate, we used Huffman lossless encoding for every case. We included the compression rate with a single level of decomposition and three levels of decompositions; both

of them show similar trends. Table 5.5 shows the compression rate of the benchmark 'Lena' image only; since the results obtained from the only single image may not be enough for a conclusion, we have also applied all the transforms on each of 44 images in SIPI Miscellaneous database [19] and presented the average value. The compression rates with both techniques are similar for SIPI Miscellaneous images as it was for 'Lena' image though the technique where we replace scaling steps with the equation (5.37) needs additional 3 lifting steps in most cases, additional flip-flops, and latency in hardware implementations.

So, in the lossless image compression application, our proposed technique provides similar performance in compression rate but uses less number lifting stages which are lower-cost in implementation. Therefore, the proposed mechanism provides a complete and general technique for the lossless wavelet implementation for both orthogonal and biorthogonal wavelets and that provides lower-cost implementation option while maintaining the performance.

Table 5.5 Lossless compression percentage with 'Lena' image

Wavelet	Lossless compression percentage (%)			
	Single level of decomposition		3 levels of decomposition	
	Replace the scaling step with eqn. (5.37)	This work	Replace the scaling step with (5.37)	This work
D4	23.74	23.75	38.84	38.86
D6	23.60	23.55	38.69	38.23
D8	23.93	24.29	39.02	39.35
D10	23.02	20.54	37.66	33.83
CDF9/7	25.57	25.61	41.31	41.37
CDF4.2	24.66	25.81	31.75	32.83
CDF5.1	13.57	19.51	13.74	19.40

*All results are with integer-to-integer mapping and 8-bit precision in the integer conversion of the coefficients

Table 5.6 Lossless compression percentage with SIPI images

Wavelet	Lossless compression percentage (%)			
	Single level of decomposition		3 levels of decomposition	
	Replace the scaling step with eqn. (5.37)	This work	Replace the scaling step with (5.37)	This work
D4	25.40	25.60	36.62	36.57
D6	24.58	24.37	35.72	36.57
D8	24.38	24.91	35.69	36.20
D10	23.16	21.08	34.16	30.72
CDF9/7	26.93	26.85	38.85	38.78
CDF4.2	25.34	27.47	29.33	31.63
CDF5.1	15.15	20.91	13.02	19.17

5.6 Conclusion

In this paper, we have shown an alternative general way of factorization of the polyphase matrix for wavelet transforms so that implementation with integer-to-integer mapping becomes purely lossless. The contributing factor for lossy integer-based implementation is the scaling step in the polyphase matrix. This paper shows the way to factorize polyphase matrix in a manner which does not return any scaling step; it incorporates the effect of the scaling step into the prior lossless lifting steps; everything is done in this factorization while making sure that it does not increase the number of lifting step factors. This paper provides a novel way of implementing purely lossless wavelet transform.

5.7 References

[1] P. S. Addison, *The illustrated wavelet transform handbook: introductory theory and applications in science, engineering, medicine and finance*. CRC press, 2002.

[2] C.-H. Hsia, J.-M. Guo, and J.-S. Chiang, "Improved low-complexity algorithm for 2-D integer lifting-based discrete wavelet transform using symmetric mask-based scheme," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 8, pp. 1202–1208, 2009.

[3] M. Vetterli and C. Herley, "Wavelets and filter banks: Theory and design," *IEEE Trans.*

signal Process., vol. 40, no. 9, pp. 2207–2232, 1992.

[4] M. L. Hilton, B. D. Jawerth, and A. Sengupta, "Compressing still and moving images with wavelets," *Multimed. Syst.*, vol. 2, no. 5, pp. 218–227, 1994.

[5] S. Mallat, *A wavelet tour of signal processing*. Elsevier, 1999.

[6] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE Trans. Inf. theory*, vol. 36, no. 5, pp. 961–1005, 1990.

[7] "Applications of Wavelet Transforms," 2015. [Online]. Available: http://www.ni.com/white-paper/3887/en/.

[8] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *IEEE Signal Process. Mag.*, vol. 18, no. 5, pp. 36–58, 2001.

[9] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 247–269, 1998.

[10] M. M. Hasan and K. A. Wahid, "Low-Cost Lifting Architecture and Lossless Implementation of Daubechies-8 Wavelets," *IEEE Trans. Circuits Syst. I Regul. Pap.*, pp. 1–9, 2018.

[11] M. M. Hasan and K. A. Wahid, "Low-cost Architecture of Modified Daubechies Lifting Wavelets using Integer Polynomial Mapping," *IEEE Trans. Circuits Syst. II*, vol. PP, no. 89, pp. 1–2, 2016.

[12] A. R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, "Wavelet Transforms That Map Integers to Integers," *Appl. Comput. Harmon. Anal.*, vol. 5, no. 3, pp. 332–369, Jul. 1998.

[13] W. Sweldens, "The Lifting Scheme: A Construction of Second Generation Wavelets," *SIAM J. Math. Anal.*, vol. 29, no. 2, pp. 511–546, Mar. 1998.

[14] C. Chen, *Signal processing handbook*, vol. 51. CRC Press, 1988.

[15] A. D. Booth, "A signed binary multiplication technique," *Q. J. Mech. Appl. Math.*, vol. 4, no. 2, pp. 236–240, 1951.

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

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

[18] K. Sayood, "Transform Coding." [Online]. Available:

http://www.ws.binghamton.edu/fowler/fowler personal page/EE523_files/Ch_13_3 Transform Coding - Coding Gain & Classic Transforms (PPT).pdf. [Accessed: 16-Jul-2018].

[19] A. Weber, “SIPI Image Database Misc.” [Online]. Available: http://sipi.usc.edu/database/database.php?volume=misc. [Accessed: 22-Sep-2018].

Chapter 6

Summary and Conclusion

6.1 Summary

The wavelet transform has become a new tool for information processing in many different fields, such as science, geophysics, astrophysics, biology, mathematics, computer science, medicine, finance, and engineering. Some of the key applications and usefulness are discussed briefly in chapter 1. Lossless implementation of the wavelet transform has been crucial in many applications. However, due to a few issues, the conventional integer-based implementations are not lossless in most wavelets. This thesis opts to solve this issue.

The discrete wavelet transform is based on a strong and complex mathematical foundation. Therefore, it is necessary to give an overview of the context of this new transform developed in the past few decades. Starting from the Fourier series/transform, chapter 2 discusses the issues of each transform and proceeds to the wavelet transform for an appropriate solution. The issues of existing architectures or implementations also are identified.

As the first step in this research, in chapter 3, a low-cost technique for DWT was proposed and applied to the widely used Daubechies-4 and -6 wavelets. The proposed architecture uses integer polynomial mapping (IPM) as well as the resource sharing/filter-coefficient elimination technique. IPM converts the irrational coefficients into integers and reduces the computational errors due to the truncation of coefficients. The proposed technique ensures an efficient low-cost and near-lossless implementation.

In chapter 4, a purely lossless technique with the elimination of lifting steps was proposed. The technique is based on a mathematical theory which was described along with its related

experimental proof. The technique was applied to another well-performing but costly wavelet, called the Daubechies-8, to provide a low-cost and lossless implementation. Along with lifting-step elimination to ensure losslessness, integer-mapping or integer-polynomial mapping, shown in chapter 3, was used on low-cost lifting schemes for lower cost architecture. Since lifting conversion of wavelet filters are non-unique, several lifting schemes were derived to choose the best ones. This ensures well-performing wavelets as determined by the entropy or capability to compress information.

In chapter 5, instead of the removal of the scaling step, the effect of scaling steps was incorporated in the previous lifting steps. The theory was constructed with a mathematical derivation and the performance predicted by the theory was accessed. Since this is a general theory, it was tested on a number of orthogonal and biorthogonal wavelets and the results were presented.

In all three chapters (chapters 3, 4 and 5), the integer-based techniques are employed. The integer-to-integer wavelet building technique are utilized where the lifting step $x_{o1} = s_1(z^{-1})x_e + x_0$ is approximated by $x_{o1} = \lfloor s_1(z^{-1})x_e + 1/2 \rfloor + x_0$ and the non-integer filter polynomials $s_i(z)$ are also mapped into an integer $S_i(z)$ by multiplying by 2^b (where b is the precision of the coefficients) followed by a rounding or floor operation. It is true that after integer-mapping, the transformed output is not exactly the same as we get in the classical wavelets and there is difference between the transformed outputs of the classical wavelet and that integer-based wavelet. Due to the difference, there is impact on the performance when precisions are lower. As observed in this research work, the impact has been minimal with higher bit precisions. Usually at 8 bit-precision, the performance (measured in terms of entropy) becomes saturated and does not improve much after that (i.e. at higher precisions). So, the impact at a reasonable precision is slim while the advantage is huge in terms of implementation cost since it is very expensive to implement double-precision floating-point calculation. Another advantage of building wavelet which maps integer-to-integer is that the lifting steps become reversible.

The evidence of the minimal impact of the integer-based wavelets compared to classical wavelets are shown for all three algorithms. In chapter 3, Fig. 3.8 compares the spectrum response of the proposed integer-based algorithms and original classical algorithm in double

precision and shows that they are quite close for both D4 and D6. Table 3.6 shows that the proposed integer based D4/D6 maintains a similar PSNR compared to classical D4/D6 (in double precision) and provides a low-cost implementation. In chapter 4, Fig. 4.3 shows that 8-bit precision of scheme-1 and 2 maintains the similar entropy in processing 'Lena' image as the entropy found in classic D8 shown in Table 4.2. The advantage of integer-implementation is evident in Table 4.6 which shows a drastic decrease in cost. In chapter 5, Table 5.2 shows almost similar root-mean-square-error (RMSE) with or without 8-bit precision (without precision implies that no integer-mapping) for different wavelets. Therefore, integer-mapping of the non-integer or irrational coefficients maintains the transform performance (measured by the entropy and PSNR) compared to the classical wavelets in double precision while decreasing the cost significantly.

Table 6.1 The proposed schemes (single level decomposition) with 'Lena' image

	Chapter 3		Chapter 4		Chapter 5		
	Lossy or near-lossless algorithm		Lossless algorithm		Generalized Lossless algorithm		
	D4	D6	D8 scheme 1	D8 scheme 2	D4	D6	D8
PSNR	50.80	65.18	∞	∞	∞	∞	∞
RMSE	0.74	0.14	0	0	0	0	0
Entropy	5.63	5.86	6.04	6.00	6.06	5.98	6.02
Adders	12	16	32	28	18	24	38
Multipliers	0	0	0	0	0	0	0
LUTs	166	164	470	389	-	-	-

Finally, all the proposed algorithms in chapter 3, 4, and 5 are summarized and shown in Table 6.1 which presents the performance with 'Lena' image. PSNRs, RMSEs, and entropies are calculated after a single level of 2D decomposition and reconstruction without any encoding such as SPIHT or Huffman encoding. The costs shown in adders or multipliers are for a single level of single-dimension decomposition with 8-bit precisions of the filter coefficients. Since the conventional polyphase matrix for D6 in Table 5.1 in chapter 5 and chapter 3 are different (which is usual because polyphase conversion is not unique), in order to have fair comparison in

the same table, the method of chapter 5 is applied on the conventional polyphase shown in the chapter 3 and following factorization is achieved which has been used for the data in Table 6.1.

$$P_{D6}(z)=\begin{bmatrix}1 & 0\\ c_1 & 1\end{bmatrix}\begin{bmatrix}1 & c_2\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ c_3 & 1\end{bmatrix}\begin{bmatrix}1 & c_4\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ c_5 & 1\end{bmatrix}\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}.$$

Here,

$c_1 = -0.4122865950$, $c_2 = -1.5651362796z^{-1} + 0.3523876576$,

$c_3 = -0.4928621825 + 0.4921518449z$, $c_4 = 1.9182029462$, and $c_5 = -0.4154318521$.

As shown in Table 6.1, chapter 3 provides the lowest cost algorithm while chapter 4 and 5 proves purely lossless implementations. Though chapter 3 is lossy, PSNR is high enough to consider it near-lossless in many practical applications. Therefore, this thesis provides three different approaches of implementation: lowest cost near-lossless algorithm, lossless and generalized lossless algorithms.

6.2 Achievement of Research Objectives

As explained in the above summary, this thesis proposes a number of algorithms which will ensure low-cost and lossless implementations and achieves the objectives of this research as outlined below.

1. The goal of completely lossless and reversible implementation of wavelet transforms has been achieved (see chapters 4 and 5). The scaling steps which are responsible for becoming lossy have been analyzed both mathematically and experimentally.
2. The method of conversion of the polyphase matrix into the lifting steps has been modified in order to find a new method so that the algorithms become free of the scaling steps without any change to the main algorithm or performance. This has been achieved in chapter 5 with the new polyphase matrix factorization technique. This has also been achieved in chapter 4 with the theory of the removal of scaling steps.
3. In each part of the research work, the cost has been compared with recent similar implementations (if they exist) and it has been found that the proposed algorithms are of comparatively lower cost. To make them low-cost, several efficient techniques have been combined. Firstly, lifting-based techniques have been used which decreases the cost by a

half. Secondly, for conversion scaling steps into lifting steps, a lower cost alternative has been used by avoiding the comparatively costly conversion shown in equation (2.36).

4. In all parts of the research work, the proposed wavelets are integer-based, and a lifting-wavelets technique is utilized that maps integer-to-integer. In this technique, the processed result in each lifting step becomes an integer which makes the lifting step reversible, and everything except the scaling steps become lossless. As mentioned earlier, in the last two parts of the work, usually lossy scaling steps become lossless.
5. In all parts of the research work, the irrational coefficients of filter polynomials have been mapped into integers. Since the floating-point calculation is inherently costly, any floating-point calculation has been avoided.
6. All parts of the work have used lifting based algorithms which allow a series of data from a source like a camera to be processed in place, making them easy to utilize in such applications.
7. The proposed techniques have been applied and tested on the popular orthogonal and biorthogonal wavelets. They have been applied on the Daubechies-4 and -6 in chapter 3, the Daubechies-8 in chapter 4, and a number of wavelets including the CDF-9/7 in chapter 5.
8. The goal of the work was to develop a generalized solution, which is achieved in chapter 5. A general algorithm has been proposed for a lossless integer-based lifting wavelet transform implementation. The proposed general algorithm has been applied and tested on a number of orthogonal and biorthogonal wavelets. In addition, other algorithms can also be generalized to apply to the other widely used wavelets.

6.3 Suggestion for Further Studies

The research done in this thesis may constitute a significant milestone towards lossless and low-cost wavelet transform implementation. It can be used for many applications and the development of newer algorithms or techniques. Following are several suggestions for further studies based on this research.

1. Since the proposed techniques are low-cost and hardware-friendly, they can be utilized to make close-to-real-time systems for many applications. While the algorithm in chapter 3 can be used for applications where near-lossless or lossy compression or processing are acceptable and lower cost is desirable, the algorithms in chapter 4 and 5 can be used for lossless implementation.
2. Though the proposed algorithms and techniques have been tested on image compression applications, there are many other applications, as outlined in chapter 1, where the proposed algorithm can be applied to get lossless and well-performing transforms at the hardware or software level. Some such areas are compression of the data from electrocardiograms (ECGs) or electroencephalograms (EEGs), computed tomography (CT), magnetic resonance (MR), and positron emission tomography (PET).
3. The proposed algorithms can be used for reversible data hiding and digital watermarking, compression of satellite or remote-sensing data and text compression.
4. The techniques in chapter 3 and 4 were applied only on the Daubechies-4, -6, and -8 wavelets. They can be used on other wavelets, as well. Techniques in chapter 3 can be applicable for near-lossless or lossy but low-cost implementation, while those in chapter 4 and 5 are more suitable for lossless and low-cost applications.
5. Proposed algorithms can be utilized for analysis and classification of time series, also in the fields of machine-learning, deep-learning, and data science. They can also be used in the area of finance to predict the stock market, especially for high-frequency trading.
6. The proposed techniques can be used to denoise or process any time-series signals from wearable devices and can be utilized for in-place implementations. Two examples are heart-rate monitoring and blood-oxygen monitoring signals.

6.4 Conclusion

Discrete wavelet transforms have significant potential for signal and data processing as a new statistical tool. Algorithm development and efficient implementation of wavelet transforms have been an active subject of research in the last few decades in many areas, such as mathematics, physical science, bioscience, and engineering. From an engineering point of view,

it is vital to contribute to efficient hardware and software implementation in order to unleash the mathematical potential for real-world applications.

Having been motivated by that purpose, the existing basic techniques have been explored. This work stands on the shoulder of the giants, like Jean Morlet, the geophysicist who pioneered wavelet analysis; Ingrid Daubechies, the physicist and mathematician significantly contributed to the formulation of orthogonal wavelets and lifting scheme, Stéphane Mallat known for developing multiresolution analysis construction for wavelets which made the wavelet implementation practical for engineering applications. Using the work of those earlier researchers, this work develops a modified version of lifting schemes developed by Daubechies. This present scheme allows for the efficient building of lossless wavelets that map integer-to-integers, addressing the limitation of the Daubechies's lifting scheme, as she herself explained, in building integer-to-integer wavelet transforms, the scaling step becomes non-invertible.

In addition, this work also focuses on low-cost, hardware-friendly implementation using overall integer-mapping, both the integer-polynomial mapping and the technique of building wavelets that map integer-to-integer. All the techniques were tested on one or more wavelets.

To conclude, the work presented in this thesis should prove to be very useful for many different applications in the fields of medical imaging, satellites, remote-sensing and even standard data processing.

Appendix

A.1 List of Publications

Related published peer-reviewed Journal papers:

1. Md. Mehedi Hasan and K. A. Wahid, "Low-Cost Architecture of Modified Daubechies Lifting Wavelets Using Integer Polynomial Mapping," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 64, no. 5, pp. 585–589, 2017.
2. Md. Mehedi Hasan and K. A. Wahid, "Low-Cost Lifting Architecture and Lossless Implementation of Daubechies-8 Wavelets," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 65, no. 8, pp. 2515–2523, 2018.
3. P. Balakrishnan, M. M. Hasan, and K. A. Wahid, "An Efficient Algorithm for Daubechies Lifting Wavelets Using Algebraic Integers," *Can. J. Electr. Comput. Eng.*, vol. 37, no. 3, pp. 127–134, 2014.

Submitted related papers in the peer-reviewed journal:

4. Md. Mehedi Hasan and K. A. Wahid, "Factoring Wavelet Transforms into Reversible Lifting Steps for Lossless Integer-to-Integer Mapped Implementation," submitted in a prestigious journal.

Other peer-reviewed journal papers published during Ph.D. work:

5. M. M. Hasan, M. W. Alam, K. A. Wahid, S. Miah, and K. E. Lukong, "A Low-Cost Digital Microscope with Real-Time Fluorescent Imaging Capability," *PLoS One*, vol. 11, no. 12, pp. e0167863–e0167863, Dec. 2016.
6. M. W. Alam, M. M. Hasan*, S. K. Mohammed, F. Deeba, and K. A. Wahid, "Are Current Advances of Compression Algorithms for Capsule Endoscopy Enough? A Technical Review," *IEEE Rev. Biomed. Eng.*, vol. 10, pp. 26–43, 2017.

   *Mohammad Wajih Alam and Md. Mehedi Hasan contributed equally to this work and therefore share the first authorship.

7. R. Shrestha, S. K. Mohammed, M. M. Hasan, X. Zhang, and K. A. Wahid, “Automated Adaptive Brightness in Wireless Capsule Endoscopy Using Image Segmentation and Sigmoid Function,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 4, pp. 884–892, 2016.

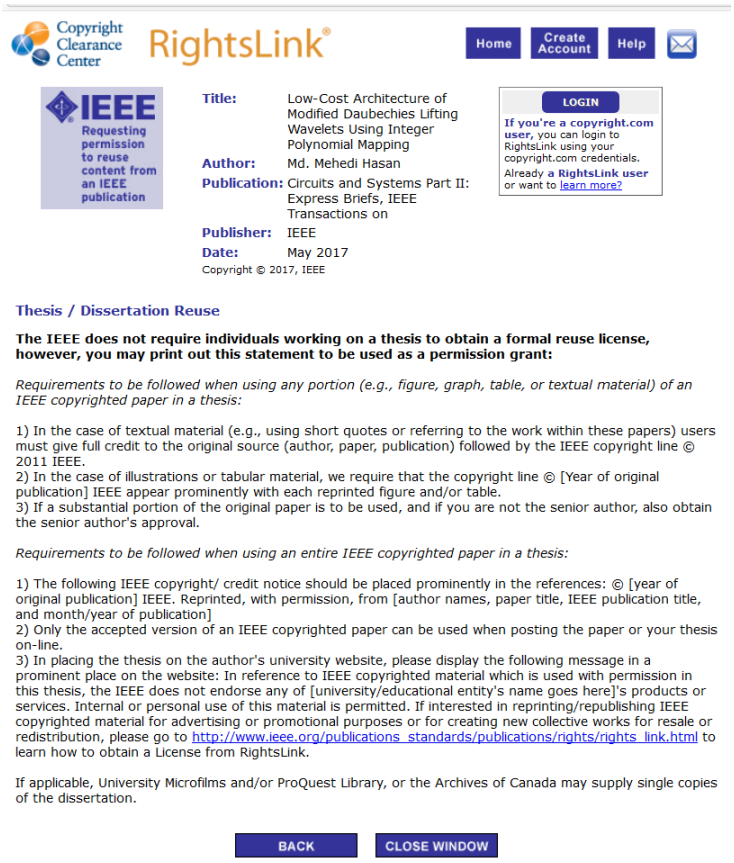
Other Conference paper published during Ph.D. work:

8. Z. Gias, M. Hasan, K. Wahid, and others, “Multi-beamforming with uniform linear array and algebraic integer quantization based DCT,” in *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*, 2015, pp. 2616–2619.

A.2 Copyright Permission

This appendix includes the copyright permission for reusing the IEEE transaction papers in this thesis. As per IEEE's guideline which states "The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant.", I am attaching the statements.

Chapter 3 reuses the manuscript titled "Low-Cost Architecture of Modified Daubechies Lifting Wavelets Using Integer Polynomial Mapping". The screenshot for the statements is as shown in Fig. A.1.


Copyright Clearance Center RightsLink®

Home | Create Account | Help

IEEE Requesting permission to reuse content from an IEEE publication

Title: Low-Cost Architecture of Modified Daubechies Lifting Wavelets Using Integer Polynomial Mapping
Author: Md. Mehedi Hasan
Publication: Circuits and Systems Part II: Express Briefs, IEEE Transactions on
Publisher: IEEE
Date: May 2017
Copyright © 2017, IEEE

LOGIN
If you're a copyright.com user, you can login to RightsLink using your copyright.com credentials.
Already a RightsLink user or want to learn more?

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

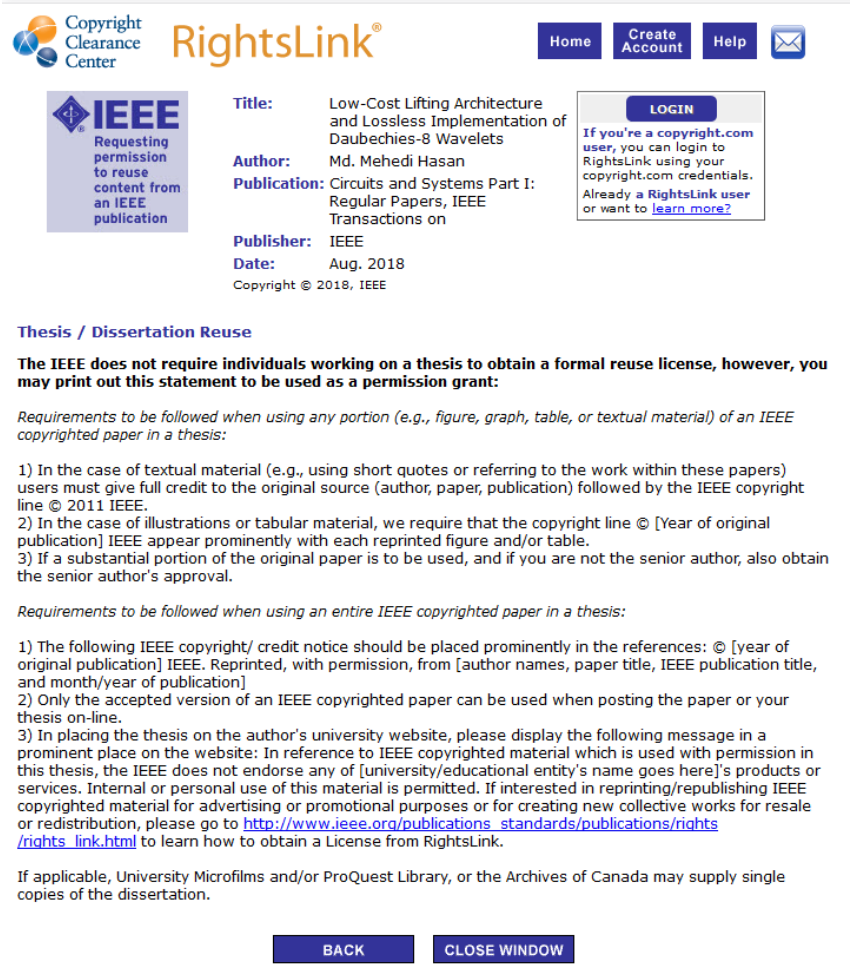
1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK | CLOSE WINDOW

Fig. A.1 IEEE statement for the manuscript reused in chapter 3.

Chapter 4 reuses the manuscript titled "Low-Cost Architecture of Modified Daubechies Lifting Wavelets Using Integer Polynomial Mapping". The screenshot for the statements is as shown in Fig. A.2.



Copyright Clearance Center RightsLink®

Home | Create Account | Help

IEEE Requesting permission to reuse content from an IEEE publication

Title: Low-Cost Lifting Architecture and Lossless Implementation of Daubechies-8 Wavelets
Author: Md. Mehedi Hasan
Publication: Circuits and Systems Part I: Regular Papers, IEEE Transactions on
Publisher: IEEE
Date: Aug. 2018
Copyright © 2018, IEEE

LOGIN
If you're a copyright.com user, you can login to RightsLink using your copyright.com credentials.
Already a RightsLink user or want to learn more?

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK | CLOSE WINDOW

Fig. A.2 IEEE statement for the manuscript reused in chapter 4.

Since this thesis would be placed in the university website, as per the guideline, I am putting the following messages for both manuscripts included in chapter 3 and 4.

"In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of University of Saskatchewan's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink."