Adaptive Wavelet Image Compression

Nasir Mahmood Rajpoot, M. Sc.

A thesis submitted to
The University of Warwick
for the degree of
Doctor of Philosophy

April 2001

Adaptive Wavelet Image Compression

Nasir Mahmood Rajpoot

A thesis submitted to
The University of Warwick
for the degree of
Doctor of Philosophy

April 2001

In recent years, there has been an explosive increase in the amount of digital image data. The requirements for its storage and communication can be reduced considerably by compressing the data while maintaining their visual quality. The work in this thesis is concerned with the compression of still images using fixed and adaptive wavelet transforms.

The wavelet transform is a suitable candidate for representing an image in a compression system, due to its being an efficient representation, having an inherent multiresolution nature, and possessing a self-similar structure which lends itself to efficient quantization strategies using zerotrees. The properties of wavelet transforms are studied from a compression viewpoint. A novel augmented zerotree wavelet image coding algorithm is presented whose compression performance is comparable to the best wavelet coding results published to date. It is demonstrated that a wavelet image coder performs much better on images consisting of smooth regions than on relatively complex images. The need thus arises to explore the wavelet bases whose time-frequency tiling is adapted to a given signal, in such a way that the resulting waveforms resemble closely those present in the signal and consequently result in a sparse representation, suitable for compression purposes.

Various issues related to a generalized wavelet basis adapted to the signal or image contents, the so-called *best wavelet packet basis*, and its selection are addressed. A new method for wavelet packet basis selection is presented, which aims to unite the basis selection process with quantization strategy to achieve better compression performance. A general zerotree structure for any arbitrary wavelet packet basis, termed the *compatible zerotree* structure, is presented. The new basis selection method is applied to compatible zerotree quantization to obtain a progressive wavelet packet coder, which shows significant coding gains over its wavelet counterpart on test images of diverse nature.

Keywords:
Lossless Data Compression, Image Compression, Wavelet Transform, Zerotree Quantization, Wavelet Packet Transform, Best Basis Selection

Acknowledgements

All gratitudes are due to the Almighty who bestowed upon me the great power of will, and by whose Grace I was able to conclude this work.

This work was supported mainly by the Quaid-e-Azam scholarship of the Ministry of Education, Government of Pakistan. The research was conducted within the Image and Signal Processing Research Group of the Department of Computer Science at the University of Warwick, UK, and the Computational Mathematics Research Group of the Departments of Mathematics and Computer Science at the Yale University, USA.

I would like to thank my laboratory fellows at Warwick for their friendship and providing a pleasant working environment, in particular: Guo-Huei Chen, Salim Gulam, Peter Meulemans, and Xiaoran Mo. Special thanks go to Doctor Ian Levy and Doctor Artur Sowa (Yale) for their friendship, help and extraordinary patience in response to my often stupid queries related to computing and applied mathematics.

I wish to appreciate the persistent encouragement and support of my supervisor, Professor Roland Wilson, and would like to thank him very much for his guidance throughout this project. I am greatly indebted to Professor Ronald Coifman of Yale University for his inspiring ideas and generous support, Doctor Francois Meyer of University of Colorado for inviting me to visit Yale and many motivating discussions, and Doctor Cenk Sahinalp of Case Western Reserve University for introducing me to the field of data compression and a continued collaboration.

Finally, I would like to thank my friends, especially Chien-Hui, and my family, especially my parents, for their continuous support.

Contents

List of Figures

List of Tables

Declaration

I declare that, except where acknowledged, the material contained in this thesis is my own work and that it has neither been previously published nor submitted elsewhere for the purpose of obtaining an academic degree.

Nasir M. Rajpoot

Chapter 1

Introduction and Scope of Thesis

The *digital information revolution* has brought with it an explosive increase in the amount of data, much of which is in the form of images. The increasing demand for such a volume of images, coming from various sources, such as medical imagery, synthetic aperture radar or SAR, fingerprints, natural scenes and so on, requires large storage and high communication bandwidth[1]. An 8-bit monochrome greyscale image of a natural scene, for example, with a 512×512 resolution (that is, having 512 rows and 512 columns of picture elements or pixels) may take nearly 256KB of space and almost 37 seconds to download via a 56kbps (kilobits per second) dial-up internet connection. If the visual quality of this image can be maintained after compressing it by a factor of 32 (which is quite possible, using the state-of-the-art compression methods and those presented in this thesis, for the majority of images), it will not only reduce the storage requirements by a factor of 32 but will also enable it to be downloaded to the user desktop in just over a second with the same 56kbps connection.

[1]Similarly, there is a growing need for efficiently storing other massive data/text (such as DNA sequences etc.) and sending them across the networks in a lossless manner (such that the original data/text is preserved and no loss, whatsoever, of the information being processed occurs). For the greater part of this thesis, emphasis will be laid on the lossy compression of images only.

1.1 Introduction

The current JPEG (Joint Picture Experts Group) image coding standard [62] employs the discrete cosine transform (DCT) on 8×8 segment blocks of the image in order to de-correlate the information contained in these blocks, avoiding at the same time any global mixing of the information contained in the whole image (which is the case when the discrete Fourier transform of the whole image is taken). At low bit rates, however, this coding scheme fails to produce visually pleasing results. The blocking artifacts, in particular, are clearly visible at even medium bit rates.

A spatial frequency subband decomposition appears to be a key feature of the human visual system (HVS) [15]. Early subband image coding methods such as [12, 100] were motivated by this observation. The development of wavelets by Strömberg [78] in 1981 and the work by Mallat and Meyer [45, 46] in 1987 led to a surge of research activity in wavelet transform based image coding during the last decade or so, primarily due to the fact that the wavelet transform matches the main feature of the HVS mentioned above. Image coding methods based on wavelets, therefore, have become the state-of-the-art[2]. Zerotree quantization [73] is an effective way of exploiting the self-similarities among high frequency subbands at various resolutions. Wavelet transform image coding methods based on the concept of zerotrees (groups of transform coefficients, belonging to different subbands, that are insignificant) such as [73, 71] have proved their superiority over other wavelet based methods in terms of both computational complexity and compression performance.

Wavelet transform based image coding methods, however, perform much better on images consisting of smooth regions with well-defined boundaries than on more com-

[2]The fact that the emerging JPEG 2000 [8] standard is wavelet based stands as firm evidence for this claim.

plex or textured images. Wavelet packets were invented [18] to pinpoint signal phenomena occurring locally in the frequency domain. The frequency adaptivity of wavelet packets makes them capable of adaptively decomposing a given image into finer spatial frequency subbands, depending on the image contents.

Various basis selection methods [22, 67, 83] have been proposed to select the best basis $\mathcal{B}^*$ among a library $\mathcal{D}$ of available wavelet packet bases by using some search method and a specific cost function. The use of different cost functions may result in different *best* bases which raises the issue of which is best overall. Moreover, these different bases may produce different coding results using the same quantization method. It is, therefore, important to take into account the quantization strategy at the time of basis selection, to ensure that the basis chosen by employing a given criterion will actually result in better performance in terms of coding gains.

Apart from being efficient in terms of both computational complexity and compression performance, zerotree quantization enables the embedded (progressive) transmission and reconstruction of images, which is required in some applications. These and the better frequency localization properties of the wavelet packet transform provide reason enough to study the possible unification of wavelet packets with zerotree quantization – one of the main subjects of the work in this thesis.

1.2 Related Work

To date, there have been many image coding methods based on the wavelet and the wavelet packet transforms. In this section, some well known methods from these two types of image coders only are reported.

1.2.1 Wavelet Image Coding

The research work on wavelet transform image coding can be divided into two main categories: schemes which employ the idea of zerotrees in order to exploit inter-band similarities (zerotree quantization methods) and schemes which employ some other method to exploit these and other similarities and the sparseness of wavelet representation.

Zerotree Quantization Methods

More precisely, a zerotree can be regarded as a quadtree consisting of a group of transform coefficients, belonging to different spatial frequency subbands, which are insignificant. The history of quadtrees (a tree whose each node, except the leaves, has four children) in image compression dates back to Wilson's work on quadtree predictive coding [95]. In wavelet coding, Lewis and Knowles [41] similarly hypothesized that in this tree-like structure, if the magnitude of a parent coefficient is below a given threshold, all of its children coefficients are most likely to follow this course too.

This idea of zerotrees was efficiently applied to the coefficient quadtrees, formed naturally in a wavelet subband decomposition, first by Shapiro [73] in his embedded zerotree wavelet (EZW) coder. In this coder, the transform coefficients belonging to a certain orientation subband are related to other coefficients in a higher spatial frequency subband of similar orientation, in such a way that each coefficient from the parent subband (at a lower resolution) is associated with four coefficients in the child subband (at a higher resolution) at a similar spatial location. The coding results of EZW were much better than those obtained by any other image coder at that time. In particular, they established the superiority of wavelet based image coding over the JPEG image coding standard.

Soon after Shapiro's paper on EZW was published, an improvement was presented by Said and Pearlman [71] which gave better results by eliminating to some extent the redundancies found in an EZW-generated bitstream. In their SPIHT (set partitioning in hierarchical trees) coder, the spatial orientation trees are similar to the zerotrees of the EZW. The image coding results of SPIHT are among the best known and it is still widely regarded as a benchmark for image coding.

Non-Zerotree Quantization Methods

The *stack-run* image coder of Tsai et al. [84] achieves comparable coding results by doing a uniform scalar quantization (with a deadzone) of the wavelet transform coefficients, followed by a clever run-length coding which uses only four symbols for encoding both zero-runs and the sign and the magnitude of significant coefficients. Their run-length coding utilizes the fact that the most significant bit (MSB) of a significant coefficient does not need to be explicitly encoded if some other means of terminating the binary word can be found, and the MSB can then be used to encode the sign of the coefficient.

The Estimation-Quantization (EQ) framework of LoPresto et al. [44] is based on mixture modelling of the wavelet coefficients belonging to all subbands and then finding and applying the optimal quantizers to each subband. They model the wavelet coefficients as being drawn from an independent Generalized Gaussian distribution, of fixed unknown shape for each subband, having a zero mean and unknown, slowly varying variances. Based on these variance estimates, they apply an off-line rate-distortion optimized quantization strategy for each subband.

The application of trellis coded quantization (TCQ) to efficiently encode the wavelet coefficients was investigated by Sriram and Marcellin [75]. Their coder uses the DCT

(with a 4×4 block size) to encode the lowest frequency subband, while the other subbands are encoded with the TCQ after distributing the bit budget optimally among all the subbands.

1.2.2 Wavelet Packet Image Coding

Although wavelet packets offer better frequency localization, there has not been as much research activity in wavelet packet image coding as in the area of wavelet image coding, perhaps due to it being computationally more expensive than the wavelet transform. The dynamic programming approach of Coifman and Wickerhauser [22] to select the best wavelet packet basis for a given cost function used a bottom-up search method and assumed that the cost functions were additive. Contrary to this bottom-up search for the optimal wavelet packet basis, Taswell [82, 83] advocated the use of top-down searches to obtain the so-called *near-best basis* in order to reduce the computational complexity of best basis selection, at the loss of some performance in compression.

The space-frequency quantization (SFQ) algorithm of Xiong et al. [103] employs a rate-distortion (R-D) optimization framework to select the best basis and assigns an optimal quantizer to each of the wavelet packet subbands at the same time. The best wavelet packet basis is selected using a dynamic programming approach [22] with a cost function made up of both rate and distortion.

Recent work by Meyer et al. [55] uses factorized non-separable 2-D filters to compute a fast wavelet packet transform for image coding. Their convolution-decimation algorithm runs 4 times faster than the standard convolution-decimation, thus neutralizing to a large extent the computational overhead in the selection of the best wavelet packet basis. They approximate the distribution of each wavelet packet subband as a Lapla-

cian distribution and employ near optimal scalar quantizers [79] designed specifically for a memoryless Laplacian source.

The work presented here is an attempt to combine wavelet packets with zerotree quantization to give an efficient and flexible image compression scheme.

1.3 Mathematical Preliminaries

In this section, a brief review of the preliminary concepts from the representation and information theories, and some complexity notations are presented.

1.3.1 Representation Theory

A *vector space* over $\mathcal{C}$ or $\mathcal{R}$, is a set E of vectors together with *addition* and *scalar multiplication* satisfying the following for $x, y \in E$ and $\alpha, \beta \in \mathcal{C}$ or $\mathcal{R}$:

1. $x + y = y + x$.
2. $(x + y) + z = x + (y + z)$, $(\alpha\beta)x = \alpha(\beta x)$.
3. $\alpha(x + y) = \alpha x + \alpha y$, $(\alpha + \beta)x = \alpha x + \beta x$.
4. $\exists\, 0 \in E$ such that $x + 0 = x$, $\forall x \in E$.
5. $\exists\, -x \in E$ such that $x + (-x) = 0$, $\forall x \in E$.
6. $\exists\, 1 \in E$ such that $1 \cdot x = x$, $\forall x \in E$.

A set M of vectors is called a *subspace* of E if $M \subset E$ and it satisfies the following conditions:

a). $x + y \in M$, $\forall x, y \in M$.

b). $\alpha x \in M$, $\forall x \in M$ and $\forall \alpha \in \mathcal{C}$.

An *inner product* of two complex-valued functions $f(t)$ and $g(t)$ over $\mathcal{R}$ is denoted by $\langle f, g \rangle$ and is given by

$$\langle f, g \rangle = \int_{-\infty}^{\infty} f^*(t) g(t) dt \tag{1.1}$$

whereas an inner product of two discrete sequences $x[n]$ and $y[n]$ is given by

$$\langle x, y \rangle = \sum_{-\infty}^{\infty} x^*[n] y[n]. \tag{1.2}$$

The *complex (real) space* $\mathcal{C}^n$ ($\mathcal{R}^n$) is the set of all n-tuples $\mathbf{x} = (x_1, \ldots, x_n)$, with finite $x_i \in \mathcal{C}$ ($\mathcal{R}$). A function $f(t)$ defined on $\mathcal{R}$ is said to be in the *Hilbert space* $L^2(\mathcal{R})$, if $|f(t)|^2$ is *Lebesgue integrable*. In other words,

$$\int_{t \in \mathcal{R}} |f(t)|^2 dt < \infty. \tag{1.3}$$

In discrete terms, a sequence $x[n]$, $n \in \mathcal{Z}$ is a vector in the *Hilbert space* $l^2(\mathcal{Z})$ if it has a finite energy (i.e. it has a finite square sum).

Two vectors $x, y \in E$ are called *orthogonal* if and only if

$$\langle x, y \rangle = 0. \tag{1.4}$$

An orthogonal set of vectors $\{x_1, x_2, \ldots\}$ ($x_i \perp x_j$, $\forall j \neq i$) is called an *orthonormal system* if all the vectors are normalized to have a unit norm. In other words,

$$\langle x_i, x_j \rangle = \delta[i - j]. \tag{1.5}$$

An orthonormal system in a vector space E is an *orthonormal basis* if it completely spans E. For an orthonormal system of vectors $\{x_k\}$ in E, *Bessel's inequality*

$$||y||^2 \geq \sum_k |\langle x_k, y \rangle|^2 \tag{1.6}$$

holds for all $y \in E$. It becomes an equality if this system is complete in E or in other words, if $\{x_k\}$ is an orthonormal basis.

1.3.2 Information Theory

Consider a discrete random variable X drawing values from an *alphabet* $\mathcal{A}$ which is generated by a *source*. The *probability mass function* $p(x) = Pr\{X = x\}$, where $x \in \mathcal{A}$, is a mapping from the symbols in $\mathcal{A}$ to real numbers such that the following *probability axioms* are satisfied:

1. $p(x) \geq 0$ for any symbol x.
2. $p(\mathcal{A}) = 1$.
3. $p(x \cup y) = p(x) + p(y)$ for any mutually exclusive symbols $x, y \in \mathcal{A}$.

The *entropy* H of this random variable X is defined as

$$H(X) = -\sum_{x \in \mathcal{A}} p(x) \log p(x). \tag{1.7}$$

The entropy $H(X)$ can be referred to as the measure of information contained in the random variable, or sometimes known as the *self-information* of a random variable. The unit of entropy is *nats* if the base of logarithm is e and it is *bits* if the base of logarithm is 2, which will be used in this thesis. The lossless source coding[3] theorem of Shannon [72] states that the average coding length of X is bounded below by $H(X)$. The *conditional probability*

$$p(y|x) = Pr(Y = y|X = x)$$

is defined as the probability of another random variable Y taking on the value y (possibly from another alphabet $\mathcal{B}$) given the random variable X takes the value x. Let $\{x_k\}_{k=1,\ldots,n}$ be a partition of $\mathcal{A}$. According to the *total probability theorem*,

$$p(y) = \sum_{k=1}^{n} p(y|x_k) \cdot p(x_k). \tag{1.8}$$

[3]Emphasis is laid upon only the source coding problem in this thesis, as fundamental theorem for a discrete channel with noise [72] allows us to separate source coding from channel coding.

The *conditional entropy* of another random variable Y given X is

$$\begin{aligned} H(Y|X) &= -\sum_{x\in\mathcal{A}} H(Y|X=x) && (1.9)\\ &= -\sum_{x\in\mathcal{A}} p(x)\sum_{y\in\mathcal{B}} p(y|x)\log p(y|x) && (1.10)\\ H(Y|X) &= -\sum_{x\in\mathcal{A}}\sum_{y\in\mathcal{B}} p(x,y)\log p(y|x). && (1.11) \end{aligned}$$

This defines the conditional entropy of Y given X as the expected value of the entropies of the conditional distributions. The *joint entropy* of two random variables X and Y with a joint distribution $p(x,y)$ is defined as

$$H(X,Y) = -\sum_{x\in\mathcal{A}}\sum_{y\in\mathcal{B}} p(x,y)\log p(x,y). \qquad (1.12)$$

The conditional and joint entropies defined above are related to each other by the *chain rule* as follows

$$H(X,Y) = H(X) + H(Y|X). \qquad (1.13)$$

This means essentially that the joint entropy of two random variables is the sum of entropy of one and the conditional entropy of the other. The *relative entropy* or Kullback-Leiber distance between two probability mass functions $p(x)$ and $q(x)$ is defined as

$$D(p||q) = \sum_{x\in\mathcal{A}} p(x)\log\frac{p(x)}{q(x)}. \qquad (1.14)$$

The *mutual information* between X and Y is defined as the relative entropy between the joint distribution $p(x,y)$ and the product distribution $p(x)p(y)$

$$I(X;Y) = -\sum_{x\in\mathcal{A}}\sum_{y\in\mathcal{B}} p(x,y)\log\frac{p(x,y)}{p(x)p(y)}. \qquad (1.15)$$

The relationships between mutual information and the conditional and joint entropies are as follows

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y|X) && (1.16)\\ I(X;Y) &= H(X) - H(X|Y) && (1.17)\\ I(X;Y) &= H(X) + H(Y) - H(X,Y) && (1.18) \end{aligned}$$

It can be easily proved that the relative entropy (1.14) is a non-negative measure,

$$I(X;Y)\geq 0. \tag{1.19}$$

Combining (1.16) and (1.19), we get

$$H(Y|X)\leq H(Y) \tag{1.20}$$

with equality if and only if X and Y are independent. Intuitively, this means that knowing another random variable Y can only reduce, on the average, the uncertainty in X. This important result will be used later in this thesis to efficiently encode the transform coefficients belonging to different subbands.

1.3.3 Computational Complexity Notation

Consider a function $g(n)$. The three most commonly used complexity notations work as a mapping from a function $g(n)$ to a set of functions. The Θ-*notation* defined as

$$\Theta\left(g(n)\right) = \{f(n) : \exists\, c_1, c_2, n_0 \text{ such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n),\ \forall\, n \geq n_0\} \tag{1.21}$$

bounds a function to within constant factors. The **O**-*notation* defined as

$$\mathbf{O}\left(g(n)\right) = \{f(n) : \exists\, c \text{ and } n_0 \text{ such that } 0 \leq f(n) \leq cg(n),\ \forall\, n \geq n_0\} \tag{1.22}$$

gives an upper bound for a function to within a constant factor. The Ω-*notation* defined as

$$\Omega\left(g(n)\right) = \{f(n) : \exists\, c \text{ and } n_0 \text{ such that } 0 \leq cg(n) \leq f(n),\ \forall\, n \geq n_0\} \tag{1.23}$$

gives a lower bound for a function to within a constant factor.

1.4 Thesis Organization

In this thesis, a detailed study of the theoretical and practical aspects of a wavelet approach – in its general adaptive form – to the image coding problem is carried out.

Issues related to the image representation in a transform coding framework are discussed in the next chapter. The quest for a suitable representation starts with a brief overview of applied harmonic analysis. The wavelet transform is selected as a suitable candidate for image representation, followed by an explanation of the wavelet subband decomposition of images using filter banks. Two other components of the transform coding framework, quantization and entropy coding, are also discussed. Entropy coding methods are classified into two main categories: dictionary based entropy coders and statistical entropy coders. A new improvement of the well known Ziv-Lempel dictionary compression scheme is presented.

Chapter 3 builds on the arguments from the previous chapter to study the properties of the wavelet transform for compression purposes, in order to develop efficient wavelet image coding algorithms. It is argued that the decay of wavelet coefficients is directly related to the compression performance at various bit rates. A simple wavelet-thresholding image coder exhibits the potential gains that can be obtained by employing a more sophisticated quantization strategy. The idea of zerotree quantization is explained and an improved coding algorithm is presented, whose compression performance is comparable to the state-of-the-art.

Wavelet image coders perform better on images consisting of smooth regions than on relatively complex images. Chapter 4 makes the case for exploring general wavelet bases (wavelet packets) whose time-frequency tiling is adapted to the contents of a given image. Issues related to the selection of the *best* basis among the huge library

of available wavelet packet bases are discussed. Necessary and sufficient conditions for a basis selected using one cost function to be better than another basis selected using a different cost function are presented. A new paradigm for wavelet packet basis selection which emphasizes the role of quantization strategy is presented.

Chapter 5 starts with the presentation of a general zerotree quantization framework for the wavelet packet transform. A set of rules is defined to construct the *compatible zerotrees* for an arbitrary wavelet packet basis. The wavelet packet subbands are modelled as a Markov chain in order to efficiently estimate the cost of zerotree quantization. This estimation is used to select the best wavelet packet basis, in the spirit of the new paradigm presented in the previous chapter. A progressive wavelet packet coder combining this cost estimator with the new compatible zerotree quantization gives significant coding gains, 0.6–1.5dB for the *Barbara* image, compared with its wavelet counterpart.

The thesis concludes with a summary and discussion of the results, limitations of the algorithms presented and future directions of research in the area of adaptive wavelet image coding.

Chapter 2

Image Representation for Coding

Many people find it useful to take notes of the seminars or meetings that they attend. We all know that musical scores are capable of describing a symphony, with reproduction of the symphony being dependent on the interpretation by musicians. Similarly, mathematical symbols and equations are helpful in explaining certain concepts and representing various natural phenomena. These are all familiar examples of a representation. What is representation and why do we need it? The *Concise Oxford Dictionary* defines 'representation' as *an image, likeness, or reproduction of a thing, e.g. a painting or drawing.* In his famous book *Vision* [51], Marr defines 'representation' in somewhat recursive terms as

> *"a formal system for making explicit certain entities or types of information, together with a specification of how the system does this. And I shall call the result of using a representation to describe a given entity a description of the entity in that representation."*

When asked why they take notes in a seminar, most of the people are likely to come up with an explanation similar to the following. By noting main points (or highlights) of

the event, we can reconstruct in our minds the whole event in a relatively easier way. These notes are, in a sense, a compressed version of the seminar: they highlight what was talked about. How easily they can recapitulate the whole event and how much of the information they are capable of reproducing clearly varies from one set of notes to another[1]. They are, nevertheless, a representation which briefly describes the seminar highlights.

The ease of reconstruction, the resources required to describe what is being represented, and the loss of information after reconstruction all depend largely on one thing: the language chosen for representation. As renowned harmonic analyst Yves Meyer says [35]:

> *"Different languages have different strengths and weaknesses. French is effective for analyzing things, for precision, but bad for poetry and conveying emotion – perhaps that's why the French like mathematics so much. I'm told by friends who speak Hebrew that it is much more expressive of poetic images. So if we have information, we need to think, is it best expressed in French? Hebrew? English? The Lapps have* 15 *different words for snow, so if you wanted to talk about snow, that would be a good choice."*

Natural images, in their raw form, are not suitably represented for their redundancies to be exploited for coding purposes. Thus arises the need for representing or *transcribing* the image into another domain (or language). In other words, a 'sparse' representation is required which can make the image redundancies more easily exploitable using structural and/or statistical approaches, thus reducing the number of bits required to

[1]The effect that the passage of time will have on reconstruction of the event is a separate issue and its discussion is beyond the scope of this thesis.

encode the image. An invertible representation incurs no loss of information, i.e. the image can be exactly reconstructed. The image representation can, however, be encoded with fewer bits, introducing some loss of information. The transcription and encoding should be carried out in such a way that the loss of information is minimal and the perceived quality of the decoded image is close to the original image.

The human visual system (HVS) responds more to the edge information (or the high bandwidth contents) in an image than the smooth information (or the low frequency contents) [51]. Human viewers notice distortion in smooth areas more easily than near the edges, the so-called *masking effect* [32]. Masking experiments show that we notice distortion in the same band as the local signal less than out-of-the-band noise [16, 77]. It seems that the HVS does a form of frequency decomposition using Gabor wavelets [94]. In order to reproduce a visually pleasing (or psychovisually tuned) image, a sparse representation that can perform a similar kind of channel or subband decomposition would be an ideal choice. The Fourier transform, introduced by Fourier in 1807, is an extremely useful analysis tool that reveals the frequency information contained in a signal. The continuous Fourier transform $F_f(\omega)$ of a function $f(t)$ is given by

$$F_f(\omega) = \int_{-\infty}^{+\infty} f(t)\, e^{-i\omega t}\, dt. \tag{2.1}$$

The original function $f(t)$ can be reconstructed from $F_f(\omega)$ by taking the inverse transform as follows

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F_f(\omega)\, e^{i\omega t}\, dt. \tag{2.2}$$

Although the discrete Fourier transform (DFT) of an image provides excellent frequency localization and is capable of capturing very well certain symmetries such as translation, it is not a good choice for image representation (especially in a compression framework) mainly due to its spatial support. Real-world signals (and images)

are non-stationary in nature and their representation in the Fourier domain makes it difficult to analyze and reproduce its local properties due to the *global mixing* of information by the Fourier transform [47]. Moreover, real-world signals are generally aperiodic and finite. This causes the induction of artificial oscillations (also known as the *Gibbs phenomenon* [9, 60]) near signal discontinuities due to truncation of the high frequency terms. The application of Heisenberg's uncertainty principle to image processing [97] suggests that when dealing with problems in image processing, analysis and coding, there is always a localization trade-off between space and spatial frequency. In the next section, we discuss alternatives to the Fourier transform which provide both space and frequency localization.

2.1 Finding a Suitable Representation

Harmonic analysis, *the study of harmonic components of a given signal*, has had an enormous impact on the evolution of present day image coding techniques (for a detailed exposition on the topic, see [27]). We attempt to present a very brief review of applied harmonic analysis only from the viewpoint of representation for coding purposes. Wavelet transforms are then selected as a starting point for efficient image representation in a compression framework.

2.1.1 Applied Harmonic Analysis: A Brief Overview

The decomposition of a signal into waveforms having a finite spread in the time-frequency plane sounds like a reasonable thing to do in order to extract useful frequency contents of the signal while avoiding the *global mixing* of information. Wigner, a theoretical physicist, was perhaps the first one who employed this idea in 1932 in the context of quantum mechanics. He proposed a time-frequency energy distribution [90]

as follows

$$W(t,\xi) = \int_{-\infty}^{+\infty} f(t+\frac{\tau}{2})\bar{f}(t-\frac{\tau}{2})\, e^{-i\xi t} d\tau. \tag{2.3}$$

Waveforms having compact support in both time and frequency were termed as *elementary time-frequency atoms* by Gabor in 1946 [29]. He argued that decomposing a signal over these elementary time-frequency atoms is closely related to our perception of sounds. The closely related windowed Fourier transform (WFT) $F_G(s,\xi)$ of the signal $f(t)$ is defined as

$$F_G(s,\xi) = \int_{-\infty}^{+\infty} f(t)\, \bar{g}_{s,\xi}(t) dt \tag{2.4}$$

where

$$g_{s,\xi}(t) = g(t-s)\, e^{i\xi t} \tag{2.5}$$

is a time window. In 1948, Ville [86] rediscovered and generalized the notion of time-frequency energy function similar to the Wigner distribution discussed above (to be termed later as *Wigner-Ville distribution*).

Although the WFT or short-term Fourier transform (STFT) provides a localized time-frequency representation, it brings with it the properties of the Fourier transform in each locality. Consider, for example, the time-frequency tiling of the STFT of an impulse, which encompasses all possible frequencies, however well localized in time it may be. Moreover, the global mix of information is now replaced by the *local mix* of information. This local mix and the discontinuity of basis functions at the block boundaries are largely responsible for the blocky artifacts caused by the JPEG image compression standard [62] at low bit rates. The JPEG standard employs block-based discrete cosine transform or DCT [2] for decorrelating the blocked image data[2].

[2]The Karhunen-Loeve transform (KLT) or the Hotelling transform does a very good job of transforming discrete variables into uncorrelated coefficients but has a computational complexity of $O(N^2)$ besides being signal dependent [85]. The DCT is an approximation to the KLT of a first-order Gauss-Markov process if the correlation coefficient (ρ) is nearly one [3].

As Daubechies proves in [25], there are no 'good' orthogonal bases in the case of the STFT due to its limited localization properties in both time and frequency. The *local cosine bases* [91, 49, 19] were designed to provide efficient time (space) localization for signal (image) analysis purposes. These bases (termed as the *Malvar bases* by Coifman and Meyer) are obtained by multiplying smooth windows of successive finite time intervals (which may be of different length) with local cosine functions of different frequencies given by

$$g_{pk}(t) = \sqrt{\frac{2}{l_p}}\, g\left(\frac{t-a_p}{l_p}\right) cos\left(\pi(k+\frac{1}{2})\frac{t-a_p}{l_p}\right) \tag{2.6}$$

where $l_p = a_{p+1} - a_p$ is the size of each interval $[a_p, a_{p+1}]$ and $g(t)$ is the indicator function

$$g(t) = \begin{cases} 1 & \text{if } 0 \leq t < 1 \\ 0 & \text{otherwise.} \end{cases} \tag{2.7}$$

The local cosine basis is well adapted for signals whose properties may vary in time but which do not include structures of very different time and frequency spread at any given time [47]. A *best* local cosine basis is an efficient representation if the image does not include very different frequency structures in the same region. In other words, it offers good localization in time at the cost of losing flexibility in frequency adaptation in each locality.

Wavelet Transform

The introduction of multiresolution image representations has been one of the most important developments in image analysis and coding over the last three decades or so [70, 81, 12, 98, 38, 94, 92]. The idea of analyzing a signal locally at different scales has resulted in a multidisciplinary boom in research on wavelets by attracting researchers from many different branches of science [31, 58, 24, 46]. The principle

behind wavelets is that shifts and dilations of a prototype function $\psi(t)$ are chosen instead of its shifts and modulations. The function $\psi(t)$ satisfies

$$\int_{-\infty}^{+\infty} \psi(t)\, dt = 0 \tag{2.8}$$

and is also called *mother wavelet.* When this function is dilated by a factor of a and translated by another scalar b, we get another wavelet denoted by $\psi_{ab}(t)$ and given by

$$\psi_{ab}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right) \tag{2.9}$$

The wavelet transform $F_w(a,b)$ of a function $f(t)$ at a scale a and position b is computed by correlating f with the wavelet $\psi_{ab}(t)$

$$F_w(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} \overline{\psi}\left(\frac{t-b}{a}\right) f(t)dt. \tag{2.10}$$

When a scale of 2^j is used, (2.9) can be written as,

$$\psi_{jk}(t) = 2^{-j/2}\psi\left(2^{-j}t - k\right) \tag{2.11}$$

where $a = 2^j$ and $b = 2^j k$. These wavelet bases dyadically decompose the function $f(t)$ into *pieces* belonging to different subspaces. The subspaces are of two types: the scaling space V_j and the wavelet space W_j. The subspace V_j is spanned by the basis $\{\phi_j(t - 2^j k)\}_{k\in\mathcal{Z}}$ and the subspace W_j is spanned by the basis $\{\psi_j(t - 2^j k)\}_{k\in\mathcal{Z}}$, where $\phi_j(t) = 2^{-j/2}\phi(2^{-j}t)$, $\psi_j(t) = 2^{-j/2}\psi(2^{-j}t)$. $\phi(t)$ is the scaling function, which is used to compute the approximation of $f(t)$ at different scales, and $\psi(t)$ is the mother wavelet function, used to determine the difference or detail information. This decomposition is such that,

$$V_j \oplus W_j = V_{j-1}. \tag{2.12}$$

The subspace W_j corresponds to the *detail* at level j. Together, the space $V_{+\infty}$ and the spaces W_j (for all $j \in \mathcal{Z}$) satisfy the completeness condition for $L^2(\mathcal{R})$ and thus

can represent an arbitrary function $f(t)$ from $L^2(\mathcal{R})$. This is illustrated in Figure 2.2, which shows the 4-level multiresolution decomposition of a 1-dimensional discrete signal (a vector made from the 325th column of the 512×512 *Lena* image shown in Figure 2.1, going through the hat, the left eye, and down the shoulder). The signal components shown here correspond to the subspaces V_{-6}, W_{-6}, W_{-7}, W_{-8}, and W_{-9} in order from top to bottom. Some commonly used scaling and wavelet functions are shown in Figures 2.3, 2.4 and 2.5.

The wavelet transform is particularly good at highlighting the high frequency signal contents at multiple scales. This becomes clear when one considers the wavelet transform of an impulse. The higher the bandwidth of a wavelet function, the more compact support it has in time and thus more capable it is of providing localized information about the impulse (unlike the STFT, where time support remains the same for all frequencies in a given time window). This helps one to pinpoint exactly where in time the impulse is occurring.

Wavelet Packet Transform

A more general form of the wavelet basis, known as the *wavelet packet basis* [18, 20] is constructed by adaptively segmenting the frequency axis (as opposed to the adaptive segmentation of time axis in case of local cosine basis). The frequency intervals of varying bandwidths are adaptively selected to extract specific frequency contents present in the given signal. This frequency segmentation is useful, for example, to analyze a local phenomenon occurring in the signal and belonging to a specific frequency band.

Consider the scaling function $\phi(t)$, the mother wavelet function $\psi(t)$ satisfying (2.8), and its scaled and translated versions $\psi_{jk}(t)$. Let W_j^p denote the subspace spanned by

Figure 2.1: The original 512×512 *Lena* image

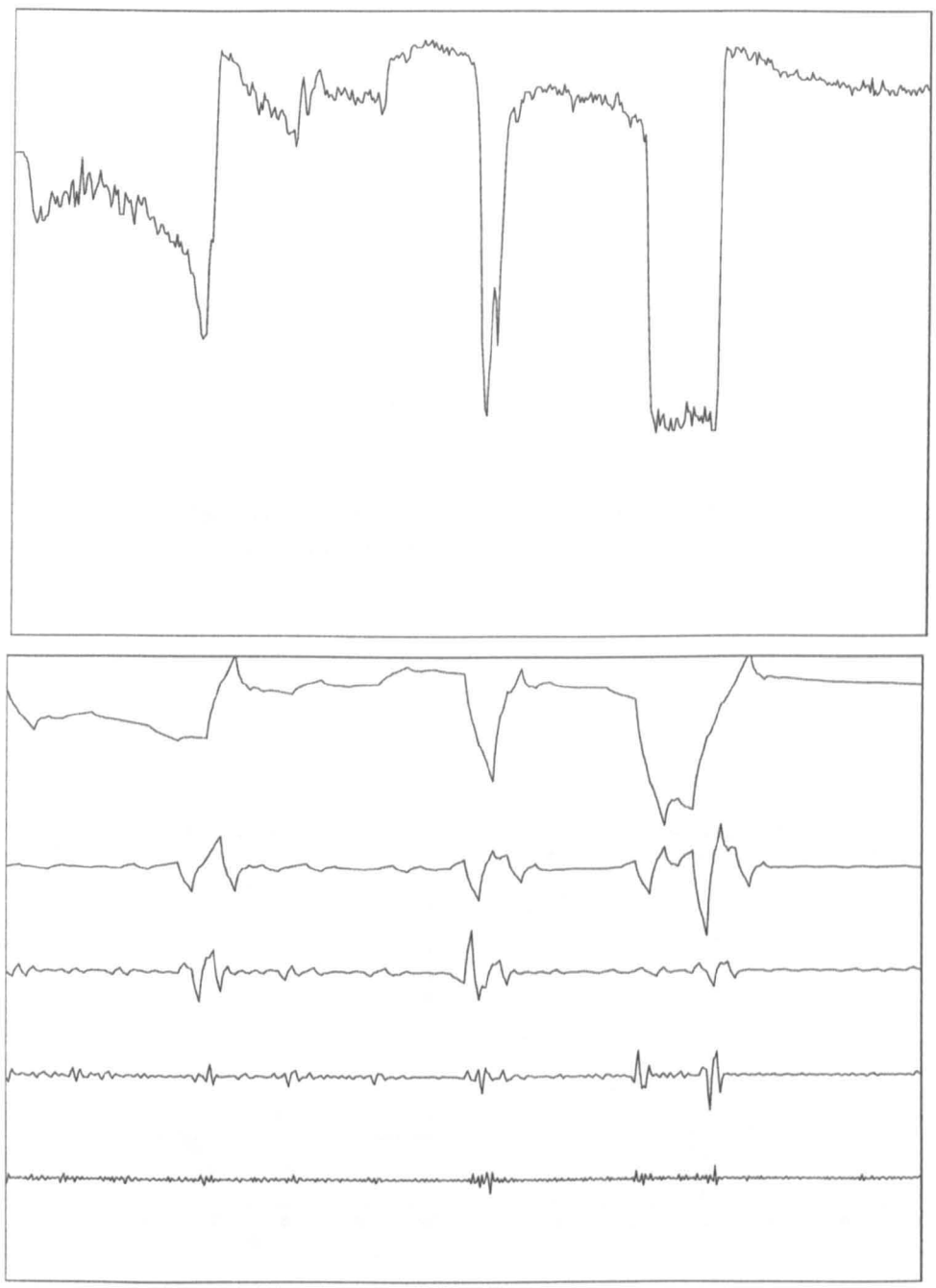


Figure 2.2: Multiresolution decomposition of a one-dimensional signal
Top: a 1-*d* vector, and *Bottom*: its components belonging to different subspaces.

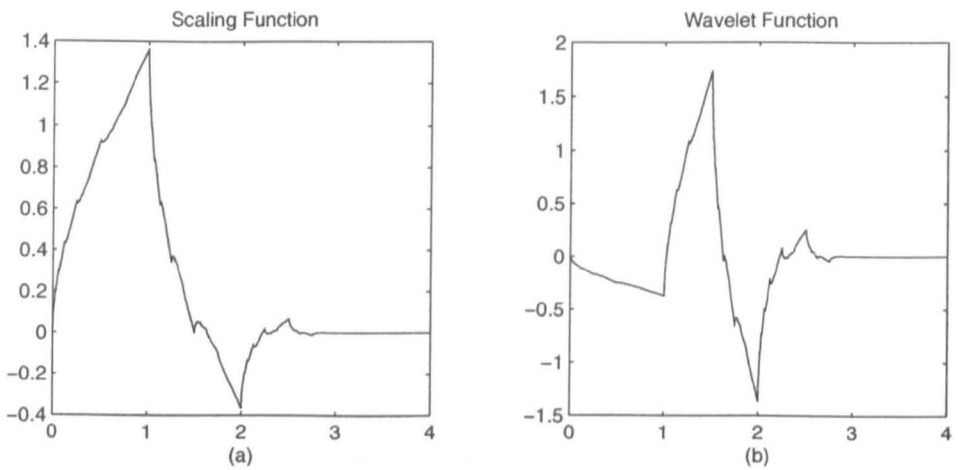


Figure 2.3: Daubechies-4 orthonormal functions
(*a*) scaling function and (*b*) wavelet function

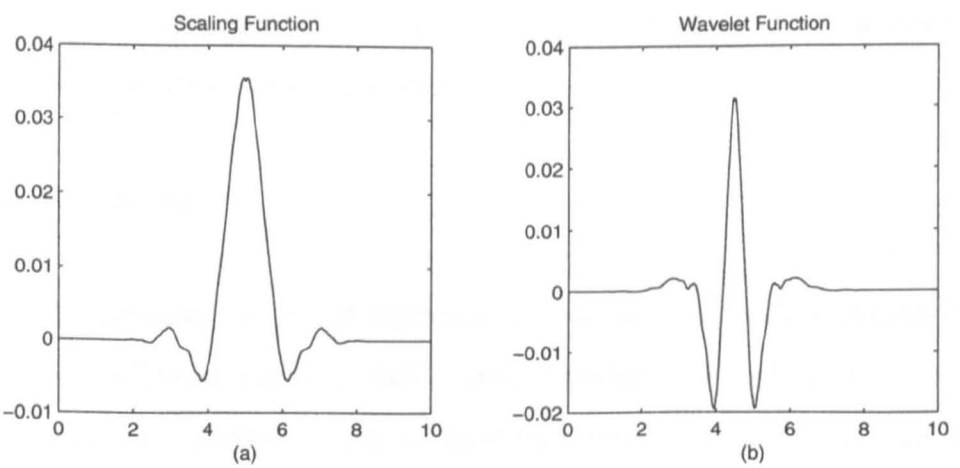


Figure 2.4: Adelson et al. near-orthogonal functions
(*a*) scaling function and (*b*) wavelet function

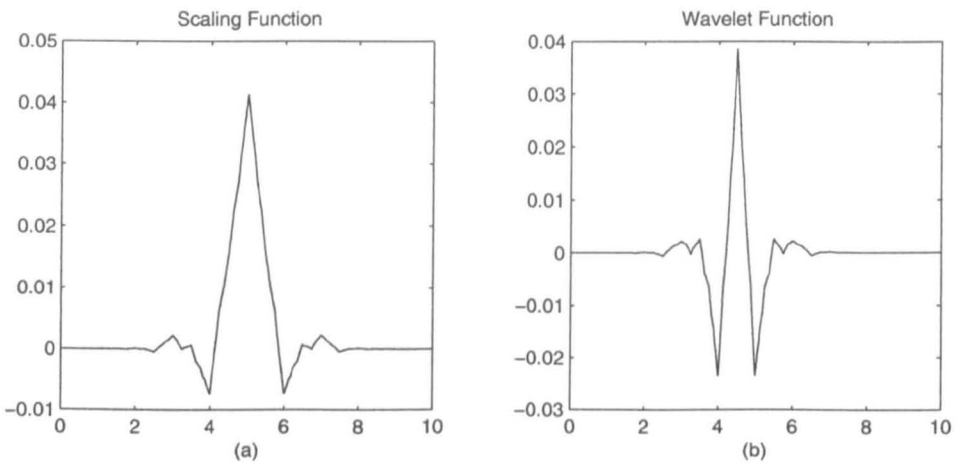


Figure 2.5: Daubechies et al. 9-7 biorthogonal functions
(a) scaling function and (b) wavelet function

$\psi_j^i(t - 2^j k)$ basis, where $\psi^0(t) = \phi(t)$ and $\psi^1(t) = \psi(t)$. As opposed to the wavelet transform, which decomposes the subspace V_j only into V_{j+1} and W_{j+1}, the wavelet packet transform decomposes each subspace W_j^p into W_{j+1}^{2p} and W_{j+1}^{2p+1} using scaling and mother wavelet functions respectively. Since this library of available bases provides an overcomplete description of the function $f(t)$, a fast optimization algorithm such as [22] is required to select a combination of bases from this library which is well suited to the function under consideration.

Related Transforms

Other recent advances in applied harmonic analysis include the development of the multiresolution Fourier transform (MFT) [96], brushlets [57], and adaptive time-frequency tiling [34]. Although each of these transforms is quite useful to represent images for analysis purposes, we do not find any of them a particularly suitable candidate for image representation in compression systems: the MFT due to its lack of orthogonality and being an oversampled representation, brushlets due to their limit-

ed applicability, and the adaptive time-frequency tiling due to its high computational complexity.

2.1.2 Why Wavelets?

We select the wavelet transform, to start with, as a representation language in an image coding framework for the following reasons:

a) *Sparse Representation*: Although the overall variance is preserved by the wavelet decomposition of a given image into low and high frequency subbands [11], it is the distribution of coefficients that determines how efficient the decomposition is from a coding viewpoint. The wavelet transform provides a sparse representation, with transform coefficients corresponding to smooth regions being very small, as is illustrated in the next section. This makes the wavelet transform particularly suitable for compressing images with smooth regions.

b) *Multiresolution Decomposition*: A multiscale representation is ideal, given the fact that there is no natural scale for images. The multiresolution nature of a wavelet decomposition brings with it multiscale edge detection [48] which can be used to efficiently reconstruct an image (according to Marr's conjecture [51]).

c) *Self-Similarity*: Despite being sparse, the wavelet subbands exhibit structural redundancies and can be organized in a tree structure. Moreover, the wavelet transform has a self-similar kind of structure due to its multiscale nature and orientation selectivity, which can be exploited for coding purposes [93, 40].

d) *Regularity and Minimax Optimality*: The amplitude of wavelet coefficients is directly related to the regularity of the function (signal) f. Also, wavelet transform codes have been shown to be nearly minimax optimal for the class of images that have bound-

ed variation [47].

2.1.3 Wavelets and Subband Decomposition

Let $f(t) \in L^2(\mathcal{R})$ denote a continuous signal. It is desired to find a complete orthonormal basis which spans $L^2(\mathcal{R})$. Let $\psi(t)$ denote the mother wavelet function, so that the basis functions in $\{\psi_{jk}\}$ are the scaled and translated versions of $\psi(t)$ given by (2.11). The signal $f(t)$, when represented in the wavelet domain, can be written as,

$$f(t) = \sum_j \sum_k c_{jk} \psi_{jk}(t) \tag{2.13}$$

where

$$c_{jk} = \langle f(t), \psi_{jk}(t) \rangle. \tag{2.14}$$

The wavelet transform of a digital image can be computed with the help of filter banks, as described in the following pages.

Filter Banks

In order to compute the multiresolution wavelet decomposition of the discrete-time signals, filter banks are used. In a two-channel filter bank, as shown in Figure 2.6, there are two filters on each side. The lowpass filter $\mathbf{H}$ and the highpass filter $\mathbf{G}$ correspond to the scaling function $\phi(t)$ and the wavelet function $\psi(t)$ respectively in the continuous time. These filters separate the signal into frequency subbands. To avoid the redundancy of the outputs of $\mathbf{H}$ and $\mathbf{G}$, both the outputs $\mathbf{U_0}$ and $\mathbf{U_1}$ are downsampled by a factor of 2, the operation denoted by $(\downarrow 2)$. This constitutes the *analysis bank.* The reconstruction of signal $\mathbf{X}$, i.e. $\widehat{\mathbf{X}}$, is achieved by upsampling of the filtered outputs $\mathbf{Y_0}$ and $\mathbf{Y_1}$ followed by the addition of filtered outputs $\mathbf{X_0}$ and $\mathbf{X_1}$. The two filters $\widetilde{\mathbf{H}}$ and $\widetilde{\mathbf{G}}$ form the *synthesis bank.* In an orthogonal filter bank,

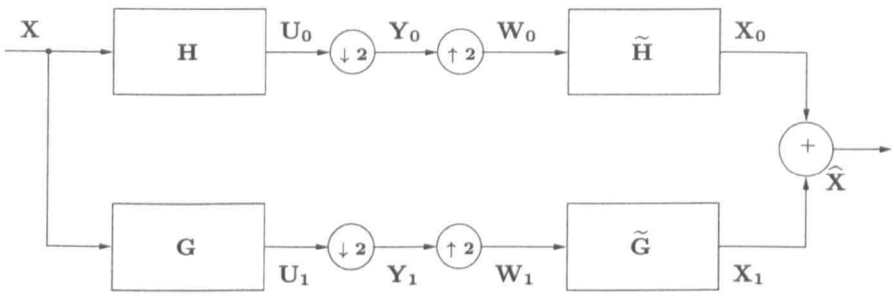


Figure 2.6: Discrete wavelet transform using a two-channel filter bank

$\widetilde{\mathbf{H}} = \mathbf{H}^{\mathbf{T}}$ and $\widetilde{\mathbf{G}} = \mathbf{G}^{\mathbf{T}}$, that is to say that the synthesis bank filters are the time reverse of those in the analysis bank.

Ideally, the approximated signal $\hat{x}(n)$ should be exactly the same as the original input signal $x(n)$: the reconstruction should be *perfect*. But there is a problem: the downsampling operation is a linear but time-variant operation and is responsible for the possible aliasing introduced in the frequency spectrum of the downsampled signal, due to the presence of a $X(\omega + \pi)$ term, where $X(\omega)$ denotes the Fourier transform of the signal $x(n)$. Croisier et al. [23] showed, in 1976, that the input signal $x(n)$ can be recovered from the filtered and subsampled signals by using a special class of filters called *quadrature mirror* (QM) filters to cancel the aliasing terms. This is possible if $X(\omega)$ is band-limited to either the upper half-band or the lower half-band [24], the so-called *spectral factorization*.

Subband Decomposition

The analysis side of a two-channel filter bank (discussed above) can be used to compute the multiresolution wavelet decomposition of an image. What is required now is a simple and fast method for the computation of wavelet transform coefficients of an image (a two-dimensional discrete-time signal). The one-dimensional (1-*d*) discrete

wavelet transform (DWT) can be computed using a conjugate quadrature filter (CQF) pair $h[n]$ and $g[n]$ as follows,

$$c^i[n] = \sum_k h[k]c^{i-1}[2n-k] \tag{2.15}$$

and

$$d^i[n] = \sum_k g[k]c^{i-1}[2n-k], \tag{2.16}$$

where $c^0[n] = x[n]$ represents the original signal and $c^i[n]$, $d^i[n]$, $i = 1, 2, \ldots$ are the scaling and wavelet coefficients respectively. In a CQF pair, $g[n] = (-1)^n h[1-n]$.

The simplest extension of the DWT to two-dimensions (2-*d*) is the separable DWT. The 1-*d* DWT is performed on the rows (treating each of them as a 1-*d* signal) followed by a 1-*d* DWT on the columns, as shown in Figure 2.7. The order does not matter here. This 2-*d* DWT splits an image of size $m \times n$ into four subimages, each of size $\frac{m}{2} \times \frac{n}{2}$. These four subimages represent the frequency information as in Figure 2.8, which shows a 1-level DWT of *Lena*, and are called *subbands*. Note that the three high frequency subbands were rescaled in order to improve the visibility. The subband LL represents the lowest frequency components of the original image at a coarse scale. The remaining three subbands HL, LH, and HH represent the high frequency components, especially the *edges*, of the image at a coarse scale. A closer look at these three subbands reveals that the HL, LH, and HH represent the high frequency components of the image in horizontal, vertical, and diagonal directions respectively. The original image is perfectly recovered by a 1-level inverse DWT as shown in Figure 2.9. Further decomposition of the image into more subbands can be achieved by operating another 1-level DWT on the LL band of the image.

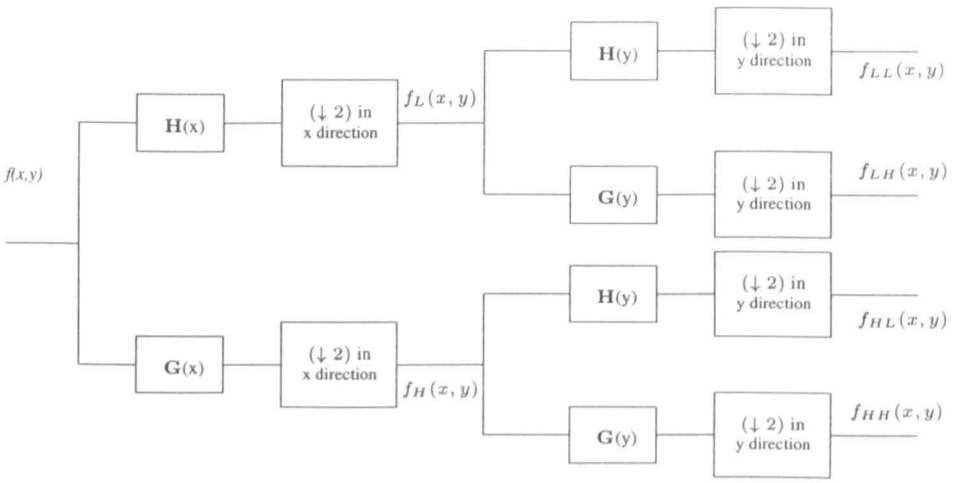


Figure 2.7: 2-*d* forward DWT of $f(x, y)$ using the analysis filter bank

Figure 2.8: Subbands in the 1-level DWT of 256×256 *Lena*

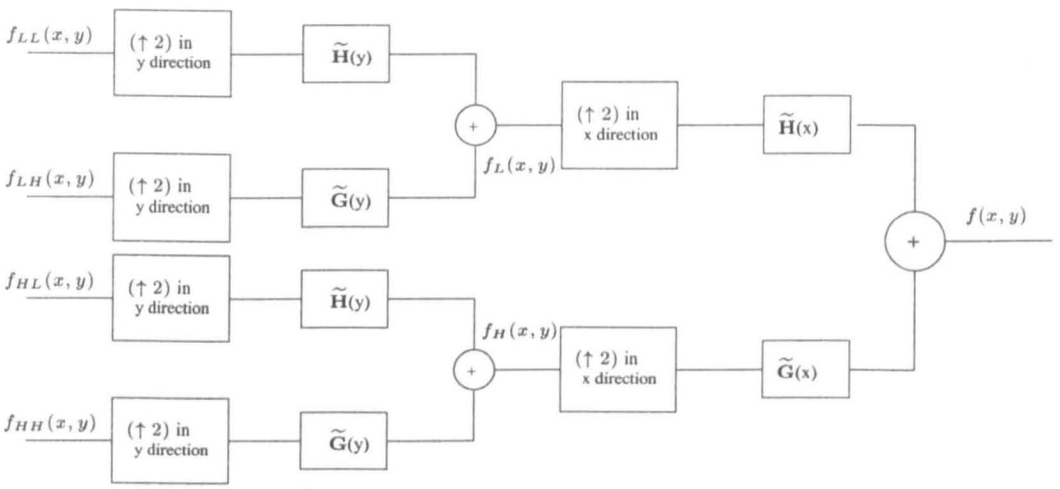


Figure 2.9: 2-*d* inverse DWT to recover $f(x, y)$ using the synthesis filter bank

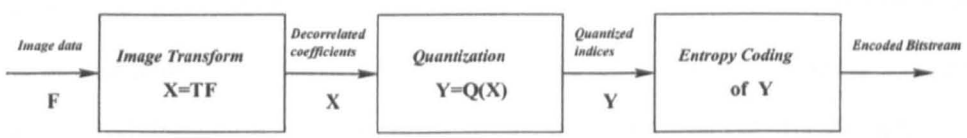


Figure 2.10: Block diagram of a simple transform coding system

2.2 Transform Coding

Once an efficient representation language has been found to transcribe a given image into the transform domain, it is now time to introduce other components of the coding system. The block diagram of a compression system based on the transform coding paradigm is shown in Figure 2.10. Consider an image $\mathbf{F}$ which is to be encoded. In this model, $\mathbf{F}$ is first transcribed into the transform coefficients $\mathbf{X}$ by applying the transform $\mathbf{T}$ so that $\mathbf{X} = \mathbf{TF}$. The transform coefficients $\mathbf{X}$ are then *quantized* into $\mathbf{Y}$ as follows

$$\mathbf{Y} = Q(\mathbf{X}) = Q(\mathbf{TF}) \tag{2.17}$$

where $Q(\cdot)$ denotes the quantizer function, designed so as to reduce the number of bits required to encode the coefficients. This is an irreversible process, which is mainly

responsible for the loss of information or *distortion*. The quantization indices $\mathbf{Y}$ are then encoded (losslessly) using an entropy coding scheme, such as Huffman coding [36], arithmetic coding [99], or dictionary coding [53]. No information is lost during entropy coding and the decoder should be able to reconstruct exactly the same quantization indices. These indices are then dequantized to approximate the transform coefficients $\hat{\mathbf{X}}$, so that

$$\hat{\mathbf{X}} = Q^{-1}(\mathbf{Y}) \tag{2.18}$$

where Q^{-1} denotes the dequantizer function. The decoded image $\hat{\mathbf{F}}$ is now reconstructed by taking the inverse transform $\hat{\mathbf{F}} = \mathbf{T}^{-1}\hat{\mathbf{X}}$ where $\mathbf{T}^{-1}$ denotes the inverse transform. The decoded image $\hat{\mathbf{F}}$ is a distorted version of the original image $\mathbf{F}$, the amount of distortion depending on the number of bits used to encode $\mathbf{F}$. The most commonly used distortion measure is the mean-squared-error (MSE) given by

$$MSE = E\{(\mathbf{F} - \hat{\mathbf{F}})^2\} \tag{2.19}$$

where $E\{\}$ denotes the expected value function and is approximated by the following sample average

$$MSE = \frac{1}{MN}\sum_{i,j}\left(F_{ij} - \hat{F}_{ij}\right)^2 \tag{2.20}$$

for an image of size $M \times N$. A related fidelity criterion, which is often used to compare the performance of image coders, is the peak-signal-to-rms-noise-ratio (PSNR) given by

$$PSNR = 20\log_{10}\left(\frac{F_p}{\sqrt{MSE}}\right) \tag{2.21}$$

where F_p is the maximum signal value, which is 255 for 8-bit greyscale images, for example.

2.3 Quantization Strategies

Quantization can be defined as a mapping Q from an input set X to an output set C such that the *ordinality* (number of elements) of C is smaller than that of X. By its very nature, quantization is a non-invertible function. It maps the input $\mathbf{x} = \{x_i\}$, $x_i \in X$, to an output $q \in C$, where q is a quantization index taken from the set of quantization indices C. If $\mathbf{x}$ contains only one element, then this operation is called *scalar quantization* and it is called *vector quantization* if $\mathbf{x}$ contains more than one elements. *Dequantization* is an 'inverse' operation which maps each element of C to one or more elements belonging to Y, a set consisting of the reconstruction values for the elements of X. Let the ordinality of both X and Y be N and the ordinality of C be M. Since $M \leq N$ and each disjoint subset consisting of one or more elements of X is mapped by Q to exactly one element in C, it is impossible to reverse the quantization operation by applying Q^{-1}. Some authors define quantization as $Q^{-1}(Q)$. We find the separation of Q and Q^{-1} to be more convenient for a general description of quantization strategies.

The operations of rounding, truncation, and thresholding are perhaps the simplest commonly used scalar quantization techniques. These functions are defined as follows.

Rounding function $\gamma(x)$ returns the nearest integer to its argument x.

$$\gamma(x) = \underset{y \in Z}{\operatorname{argmin}} |x - y| \tag{2.22}$$

Truncation function $\tau(x)$ returns only the integer part of its argument x.

$$\tau(x) = \lfloor x \rfloor \tag{2.23}$$

Thresholding function $\Theta(x)$ returns x if $x \geq \theta$, where $\theta \in \mathcal{R}$ is a threshold value, and it

returns 0 otherwise.

$$\Theta(x) = x\ 1_{x \geq \theta} \tag{2.24}$$

If x has a zero mean, an absolute thresholding function is normally used defined as,

$$\Theta_a(x) = x\ 1_{|x| \geq \theta}. \tag{2.25}$$

The output of Θ or Θ_a may usually be rounded to the nearest integer in order to reduce the ordinality of C.

If any of these four functions is employed as a quantizer, the dequantizer Q^{-1} is simply a straight-line function $Q^{-1}(y) = y$. The graphs of these functions used as quantizers are shown in Figure 2.11. The truncation function $\tau(x)$ behaves as a *deadzone quantizer*, since the length of interval for which $\tau(x) = 0$ is twice the length of all other intervals for which $\tau(x) \neq 0$. Such quantizers are of great importance for both speech and image coding because they produce a zero output for small input values. A coupling of Θ_a with a scaled version of γ will produce a similar kind of result.

In general, a scalar quantizer Q_s divides X into disjoint intervals $R_i = (x_{i-1}, x_i]$, $1 \leq i \leq M$, such that

$$Q_s(x) = i, \text{ if } x \in (x_{i-1}, x_i].$$

The difference $\Delta_i = x_i - x_{i-1}$ is the length of the ith interval $(x_{i-1}, x_i]$ and is also known as the *step size* or the *bin size* of the ith quantization bin.

Distortion or loss of information in the dequantized values is an obvious consequence of the quantization-dequantization operations. The two most commonly used objective distortion measures are the MSE and mean absolute error (MAE). It should be noted, however, that these distortion measures do not bear a simple relation with the subjective quality of dequantization values. For a uniform scalar quantizer with step size Δ at high resolution r (where $r = \log_2 M$), the MSE distortion D is given by $D \approx \Delta^2/12$.

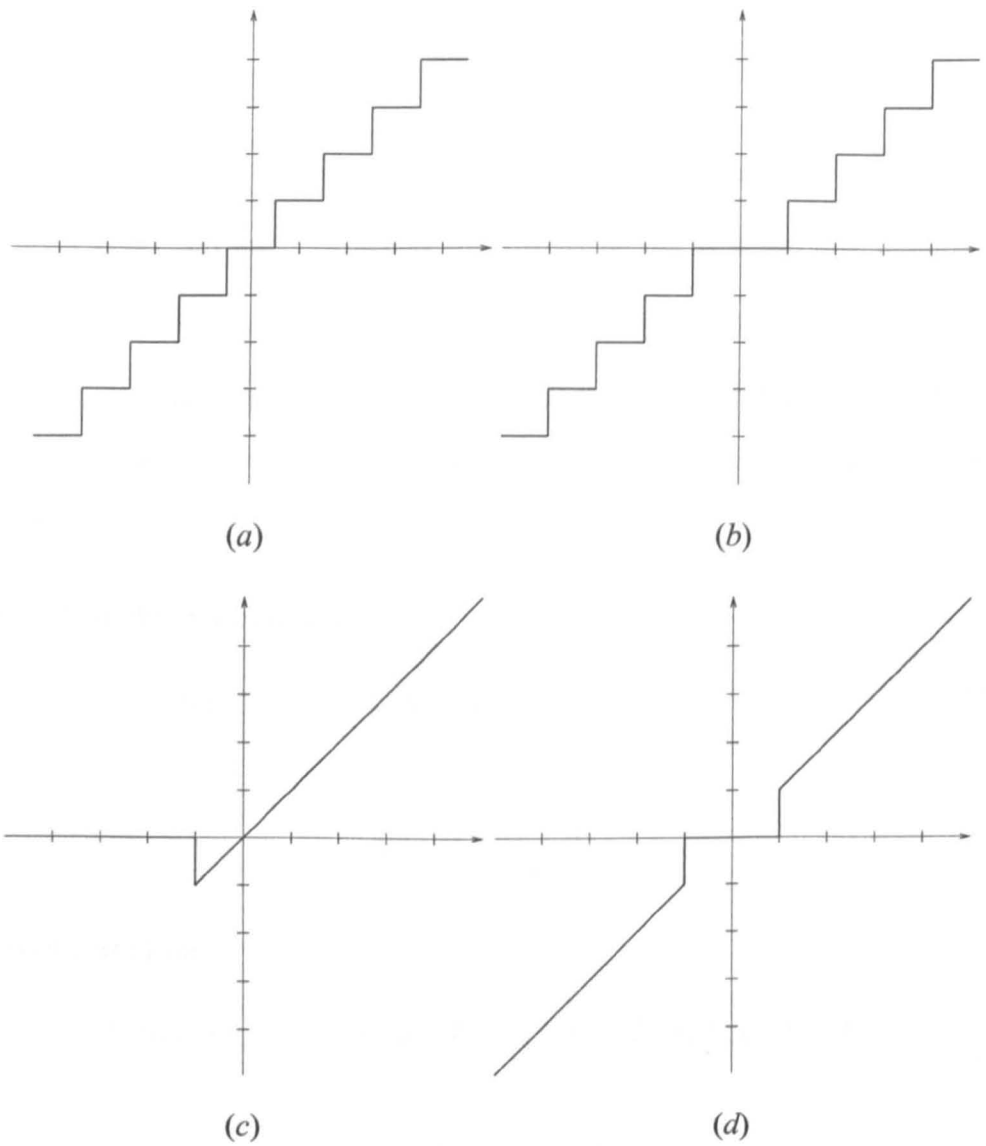


Figure 2.11: Graphs of simple scalar quantization functions
(a) Rounding function, (b) Truncation function, (c) Thresholding function ($\theta = -1$), and (d) Absolute thresholding function ($\theta = 1$)

In a non-uniform scalar quantizer, the step sizes Δ_i are not the same for all granular regions. In other words, the *boundary points* x_i are not equally-spaced. The distortion D for a non-uniform scalar quantizer at a high resolution r is given by [30],

$$D \approx \frac{1}{12} \sum_{i=1}^{M} \left(P_i \Delta_i^2 \right) \tag{2.26}$$

where

$$P_i = Pr\left(X \in R_i\right) = \int_{x_{i-1}}^{x_i} f_X(x) dx = f_i \Delta_i$$

and $f_X(x)$ is the pdf for X which is equal to f_i for $x \in R_i$ if the resolution is high enough.

The two optimality conditions for a scalar quantizer Q_s and dequantizer Q_s^{-1}, also known as the *nearest-neighbour condition* and the *centroid condition* respectively, are as follows.

Nearest-Neighbour Condition:

$$Q(x) = i \text{ if and only if } d(x, y_i) \leq d(x, y_j),\ \forall j \neq i \tag{2.27}$$

or

$$x_{i-1} = \frac{y_{i-1} + y_i}{2}.$$

Centroid Condition:

$$Q^{-1}(i) = y_i = \texttt{centroid}(R_i) = \underset{y \in \mathcal{Z}}{\operatorname{argmin}}\ E\{d(X, y) | X \in R_i\}. \tag{2.28}$$

If the distortion function $d(\cdot, \cdot)$ in (2.28) is MSE, then y_i is the center of mass of the interval R_i and if it is MAE, then y_i is the median of R_i.

A quantizer satisfying (2.27) and (2.28) is called a *Lloyd-Max quantizer*. A minimax-style iterative algorithm, known as *Lloyd's algorithm* [43], can be employed to design the interval R_i's and the codebook C.

2.3.1 Vector Quantization

A quantization function Q_v that maps each vector $\mathbf{x}_i \in X$ to an index $i \in C$ is called a vector quantizer. The quantization function Q_v^{-1} maps each $i \in C$ to a decoded vector $\mathbf{y}_i \in Y$

$$\mathbf{x}_i \in X \xrightarrow{Q_v} i \in C \xrightarrow{Q_v^{-1}} \mathbf{y}_i \in Y.$$

The idea here is to represent a group of input values, in the form of a vector, by a quantization index which can later be used to estimate these input values. This is a generalization of the scalar quantization and seems to offer more compression than a scalar quantizer Q_s due to the representation of a group of inputs by a single index. The nearest-neighbour and centroid optimality conditions can easily be generalized [42] so as to use the Lloyd's algorithm for finding the R_i's and the codebook C whose each index is now mapped by the dequantizer Q_v^{-1} to a vector $\mathbf{y}_i$.

The idea of *zerotree quantization* has recently been shown to be quite successful in the image coding domain. It can be regarded as a variable-size vector quantization scheme. A *zerotree* quantization index is generated when a group of input values belonging to different wavelet subbands follows a certain criterion. We shall study this quantization method in detail in later chapters.

2.4 Entropy Coding

The output of a quantizer Q (i.e. the quantization indices) needs to be encoded efficiently so that redundancies in the quantizer output can be exploited. This encoding has to be *reversible* or *lossless*, so that exactly the same sequence of quantizer indices can be recovered by the decoder and corresponding reconstruction values obtained. The inverse wavelet transform of these quantized coefficients then provides an approx-

imation of the original image. This *uniquely decodable* lossless encoding of quantizer indices is also known as *entropy coding*, because lossless encoding methods are now available which can encode these indices to a bit rate close to the entropy. Let Σ denote a constant size alphabet. An entropy coding algorithm C reads input array T, consisting of n symbols chosen from Σ, and computes an output T' whose representation is smaller than that of T, such that a corresponding decompression algorithm $C^{\leftarrow}$ can take T' as input and reconstruct T. Let $T[i]$ denote the ith element of T $(1 \leq i \leq n)$, and $T[i:j]$ be the subarray which begins at $T[i]$ and ends at $T[j]$ and so $T = T[1:n]$.

There are two main types of redundancies found in data to be encoded in a reversible manner: *structural* and *statistical*. Coders based on dictionaries of common phrases are good at dealing with the former type of redundancies, while statistical coders take advantage of redundancies of the latter type. All dictionary schemes have an equivalent statistical scheme that achieves exactly the same compression [6], but this comes with an added computational complexity, in order to accommodate all possible combinations for the context.

2.4.1 Dictionary Based Coding

The most common compression algorithms in practice are the dictionary schemes, also called parsing [6] or textual substitution [76] schemes. Such algorithms are based on maintaining a *dictionary* of strings of input symbols that are called *phrases*, and replacing substrings of an input with pointers to identical phrases in the dictionary. The term *string* here refers to an array of input symbols chosen from Σ and arranged in a particular order, and the term *substring* refers to a subarray of the string, usually referring to the text input. The task of partitioning the input into phrases is called *parsing* and the pointers replacing the phrases are called *codewords*.

A dictionary can be constructed in static or dynamic fashion. In *static* schemes, the whole dictionary is constructed before the input is compressed. Most practical compression algorithms, however, use *dynamic* schemes, introduced by Ziv and Lempel [105, 106], in which the dictionary is initially empty and is constructed incrementally: as the input is read, some of its substrings are chosen as dictionary phrases.

A dynamic dictionary $\mathcal{D}$ initially consists of all possible one-symbol substrings. The codewords of such substrings are the symbols themselves. As the input T is read, $\mathcal{C}$ adds some of its substrings to $\mathcal{D}$ and assigns them unique codewords – such substrings of T are called phrases of $\mathcal{D}$. Each block $T_j = T[b_j : b_{j+1} - 1]$ (where $b_1 = 1$) is identical to a phrase in $\mathcal{D}$: hence $\mathcal{C}$ achieves compression by replacing substrings of T with pointers to these phrases in $\mathcal{D}$. The decompression algorithm $\mathcal{C}^{\leftarrow}$ corresponding to $\mathcal{C}$ takes $C[1 : k]$ as input and computes $T[1 : n]$ by replacing each $C[j]$ by its corresponding block T_j. Because the codeword $C[j]$ is a function of $T[1 : b_j - 1]$ only, the decompression can be correctly performed in an inductive fashion.

Two most famous dictionary based compression methods are: LZ77 [105] and LZ78 [106], its LZW variant [88]. The LZW scheme is the basis for the UNIX `compress` program, the `GIF` image compression format, and is used in the most popular fax and modem standards (V42bis). The LZ77 algorithm is the basis for all `zip` variants. Both algorithms: (1) are asymptotically optimal in the information theoretic sense, (2) are efficient with $O(1)$ processing time per input symbol, (3) require a single pass over the input, and (4) can be applied on-line.

LZ77 Algorithm: The LZ77 algorithm reads the input symbols from left to right while inserting all its substrings in $\mathcal{D}$. In other words, when it reads $T[i]$, all possible substrings of the form $T[j : \ell]$, $j \leq \ell < i$ are in $\mathcal{D}$, together with all substrings of size one. The codeword of the substring $T[j : \ell]$ (as shown in Figure 2.12) is

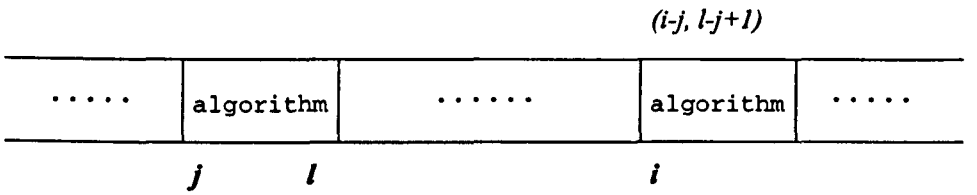


Figure 2.12: Working of the LZ77 algorithm

the 2-tuple $(i - j, \ell - j + 1)$, where the first entry denotes the relative location of $T[j : \ell]$ and the second entry denotes its size. It uses greedy parsing: the mth block $T_m = T[b_m : b_{m+1} - 1]$ is recursively defined as the longest substring which is in $\mathcal{D}$ just before $\mathcal{C}$ reads $T[b_{m+1} - 1]$.

LZ78 Algorithm (LZW variant): The LZW algorithm reads the input symbols from left to right while inserting in $\mathcal{D}$ all substrings of the form $T[b_m : b_{m+1}]$. Hence the phrases of LZW are the substrings obtained by concatenating the blocks of T with the next symbol following them, together with all possible substrings of size one. The codeword of the phrase $T[b_m : b_{m+1}]$ is the integer $|\Sigma| + m$, where $|\Sigma|$ is the size of the alphabet Σ. Thus, the codewords of substrings do not change in the LZW algorithm. LZW uses greedy parsing as well: the mth block T_m is recursively defined as the longest substring which is in $\mathcal{D}$ just before $\mathcal{C}$ reads $T[b_{m+1} - 1]$. Hence, no two phrases can be identical in the LZW algorithm.

The practical performance of these algorithms varies, however, depending on the type of input. For example, the LZ77 algorithm may perform better for English text, and the LZ78 algorithm may perform better for binary data, or DNA sequences. For files of size larger than 1MB, `compress` can improve over `gzip` by up to 20% [52, 53]. A simple *counting argument*, however, shows that there cannot exist a single dictionary construction scheme that is superior to other schemes for all inputs. If a compression algorithm performs well for one set of input strings, it is likely that it will not perform

well for others. The advantage of one dictionary construction scheme over another can only apply with regard to restricted classes of input texts.

2.4.2 Optimal Parsing for Dictionary Based Coding

The dictionary constructed by most dynamic schemes (e.g. [105, 106, 88, 104]) satisfies the *prefix property* for any input string: in any execution step of the algorithm, for any given phrase in the dictionary, all its prefixes are also phrases in the dictionary. Almost all dynamic dictionary based algorithms in the literature, including the Lempel-Ziv methods, use *greedy parsing*, which takes the uncompressed suffix of the input and parses its longest prefix which is a phrase in the dictionary. The next substring to be parsed starts where the currently parsed substring ends. Greedy parsing is fast and can usually be applied on-line, and is hence very suitable for communications applications. However, greedy parsing can be far from optimal for dynamic dictionary construction schemes [54]: for the LZW dictionary method, there are strings T which can be (optimally) parsed to some m phrases, for which greedy parsing obtains $\Omega(m^{3/2})$ phrases.

Flexible parsing ($\mathcal{FP}$) with one step lookahead, however, has been shown to be optimal [54] for dictionaries satisfying the prefix property in every execution step of the algorithm: it partitions any input string to a minimum number of phrases possible while constructing the same dictionary. Two dictionary coding algorithms have been developed based on the $\mathcal{FP}$ parsing method: the first algorithm, LZW-$\mathcal{FP}$, builds the same dictionary as LZW but uses $\mathcal{FP}$ for output generation, and the second algorithm FPA (for $\mathcal{FP}$-based-alternative-dictionary-LZW algorithm) constructs the dictionary by inserting in it the concatenation of each of the substrings parsed with the input symbol following them, with the dictionary still satisfying the prefix property in every

LZW parsing

Input:	a	b	a b	a b a	a b a a	b a	a	a b
LZW Output:	0	1	2	4	5	3	0	

LZWFP parsing

Input:	a	b	a b	a b a	a b a	a b a a	b a	a b
LZWFP Output:	0	1	2	4	4	5	2	

Figure 2.13: Working of greedy parsing and the $\mathcal{FP}$, both using the LZW dictionary construction, on the same input

execution step of the algorithm. A brief description of these algorithms is as follows.

LZW-$\mathcal{FP}$ Algorithm: The LZW-$\mathcal{FP}$ algorithm reads the input symbols from left to right while inserting in $\mathcal{D}$ all substrings of the form $T[b'_m : b'_{m+1}]$, where b'_m denotes the beginning location of block m if the compression algorithm used were LZW. Hence for dictionary construction purposes LZW-$\mathcal{FP}$ emulates LZW: for any input string LZW and LZW-$\mathcal{FP}$ build identical dictionaries. The output generated by these two algorithms, however, is quite different. The codeword of the phrase $T[b'_m : b'_{m+1}]$ is the integer $|\Sigma| + m$, where $|\Sigma|$ is the size of the alphabet Σ. LZW-$\mathcal{FP}$ uses flexible parsing: intuitively, the m^{th} block T_m is recursively defined as the substring which results in the longest advancement in iteration $m + 1$. More precisely, let the function f be defined on the characters of T such that $f(i) = \ell$ where $T[i : \ell]$ is the longest substring starting at $T[i]$, which is in $\mathcal{D}$ just before $\mathcal{C}$ reads $T[\ell]$. Then, given b_m, the integer b_{m+1} is recursively defined as the integer α for which $f(\alpha)$ is the maximum among all α such that $T[b_m : \alpha - 1]$ is in $\mathcal{D}$ just before $\mathcal{C}$ reads $T[\alpha - 1]$. The working of greedy parsing and $\mathcal{FP}$, both using the dictionary construction of LZW, is shown in Figure 2.13.

FPA Algorithm: The FPA algorithm reads the input characters from left to right while inserting in $\mathcal{D}$ all substrings of the form $T[b_m : f(b_m) + 1]$, where the function f is as described in LZW-$\mathcal{FP}$ algorithm. Hence for almost all input strings, FPA constructs an entirely different dictionary from that of LZW-$\mathcal{FP}$. The codeword of the phrase $T[b_m : f(b_m) + 1]$ is the integer $|\Sigma| + m$, where $|\Sigma|$ is the size of the alphabet Σ. FPA again uses flexible parsing: given b_m, the integer b_{m+1} is recursively defined as the integer α for which $f(\alpha)$ is the maximum among all α such that $T[b_m : \alpha - 1]$ is in $\mathcal{D}$.

Both LZW-$\mathcal{FP}$ and FPA have been shown to be superior to both `compress` for all types of input and `gzip` for non-textual data and all data of size 1MB or larger. The improvement on `gzip` is up to 35% for pseudo-random binary input and DNA sequences (see [53] for a detailed comparison).

2.4.3 Statistical Entropy Coding

We have seen that dictionary coding methods parse the input into its components belonging to a dictionary of phrases and replace each component with a codeword. The dictionary, however, does not differentiate between phrases occurring more frequently and those occurring less frequently. It may generate codewords of same length for two phrases having very different frequency of occurrence. A statistical coder tackles this problem by generating longer codewords for input symbols having low frequency of occurrence and shorter codewords for those having high frequency. Two most important coders of this breed are the *Huffman coder* [36] and the *arithmetic coder* [69]. Unlike Huffman coding, which achieves optimality only when probability of occurrence of each input symbol is an exact power of $1/2$, arithmetic coding has been shown to be optimal in achieving compression rates close to the entropy [6]. Following is a brief description of the latter of these techniques.

2.4.4 Arithmetic Coding

Consider a source alphabet consisting of symbols $\Sigma = \{s_0, s_1, s_2, \dots\}$ with their probability of occurrence being $p_0, p_1, p_2, \dots$ respectively. For a finite source, these probabilities could be generated using a frequency table either *statically* before the coding begins or *dynamically* as input from the source is read (in which case, their values may change with time). Let us assume, for the sake of simplicity, that the former is the case. Arithmetic coding encourages separation of the source model from the real encoding algorithm. Each symbol s_k from the alphabet Σ corresponds to an interval $[a_k, b_k)$ where $a_0 = 0, b_{|\Sigma|-1} = 1$ and a_k, b_k are given by

$$a_k = b_{k-1},\ b_k = a_k + p_k. \tag{2.29}$$

In its simplest form, an arithmetic coder encodes each input symbol $T[i]$ separately and encodes an input stream $T[1:n]$ by a real number ζ belonging to a coding interval $[x_n, y_n]$ which uniquely corresponds to T, where $x_0 = 0$ and $y_0 = 1$ and $[x_i, y_i]$ denotes the interval at the ith iteration of the algorithm. After it reads the input symbol $T[i]$, the coder modifies the previous coding interval $[x_{i-1}, y_{i-1}]$ to $[x_i, y_i]$ as follows

$$x_i = x_{i-1} + a_i \delta_{i-1} \tag{2.30}$$

$$y_i = x_{i-1} + b_i \delta_{i-1} \tag{2.31}$$

where $\delta_i = (y_i - x_i)$. When the decoder receives the number ζ, it is able to exactly decode T by performing the inverse operations.

Due to the iterative modification of the coding interval, this interval becomes smaller as the encoding proceeds and thus more bits are required to specify it. But more likely symbols narrow the coding interval by less than the unlikely symbols do, effectively spending fewer bits to encode a symbol with high probability of occurrence than to

encode one with low probability of occurrence. However, a computing machine capable of performing very high precision floating point arithmetic is required. The first limited-precision arithmetic coder appearing in the coding literature was [99]. In our experiments, we will use a modified version of this implementation.

2.5 Chapter Summary

In this chapter, issues related to the image representation for compression purposes were discussed. After a brief review of the representations from the applied harmonic analysis literature, the wavelet transform was selected as a suitable starting point. An explanation of the wavelet subband decomposition of images with the help of filter banks was presented. Components of the transform coding framework other than the wavelet transform, i.e. quantization and entropy coding, were briefly discussed. A new improvement to the dynamic dictionary based entropy coding based on an optimal parsing strategy was presented.

Chapter 3

Fixed Wavelet Image Coding

In this chapter, the coding properties of the wavelet transform are studied. This is followed by the presentation of two wavelet transform based image coding algorithms.

3.1 Haar Wavelet Transform

Let us start with the simplest (and the oldest) wavelet transform, namely the transform invented by Haar in 1910. The Haar scaling function $\phi_{\mathrm{h}}(t)$ is the indicator function $1_{[0,1]}$. It is clear from this definition of the Haar scaling function that

$$\phi_{\mathrm{h}}(t) = \phi_{\mathrm{h}}(2t) + \phi_{\mathrm{h}}(2t-1). \tag{3.1}$$

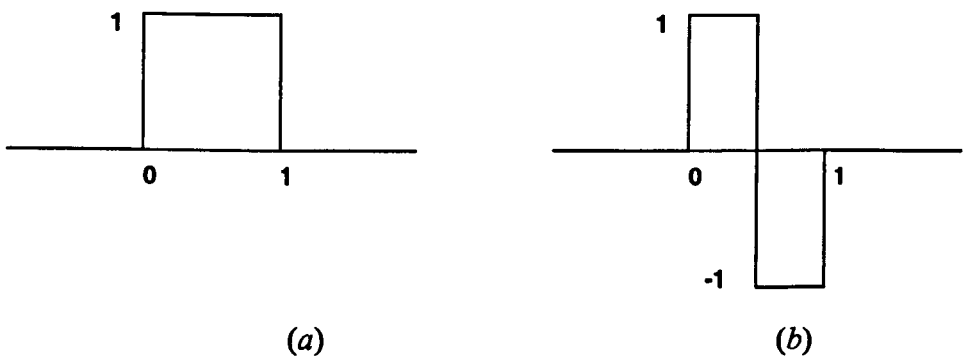
The Haar mother wavelet function is given by

$$\psi_{\mathrm{h}}(t) = \begin{cases} 1 & 0 \le t < \frac{1}{2} \\ -1 & \frac{1}{2} \le t \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

and can also be written as

$$\psi_{\mathrm{h}}(t) = \phi_{\mathrm{h}}(2t) - \phi_{\mathrm{h}}(2t-1). \tag{3.3}$$

The dilations and translations of $\psi_{\mathrm{h}}(t)$ provide a simple way of approximating a given signal with these piecewise constant functions. The Haar *father* and *mother* wavelet



(*a*) (*b*)

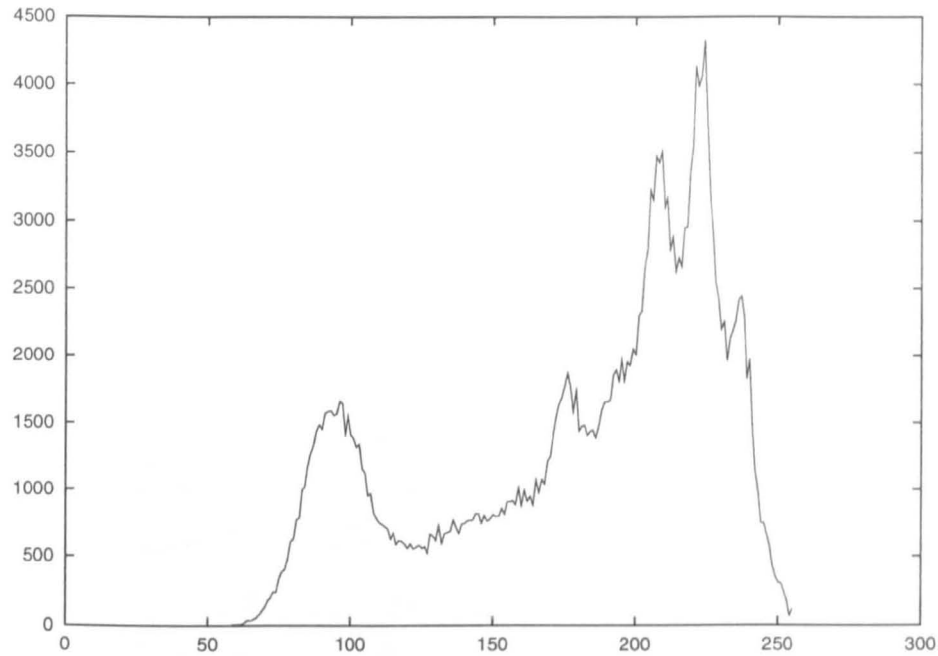
Figure 3.1: Haar Functions
(*a*) scaling function, and (*b*) wavelet function

functions are shown in Figure 3.1. It is intuitively clear that the Haar scaling and wavelet functions correspond to the sum and difference operations respectively. In order to compute the Haar wavelet transform, we use the filter banks concept described in the previous chapter. The coefficients of the Haar lowpass and highpass filters are $\left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$ and $\left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right)$ respectively. The factor $\frac{1}{\sqrt{2}}$ in these filters gives them a unit norm.

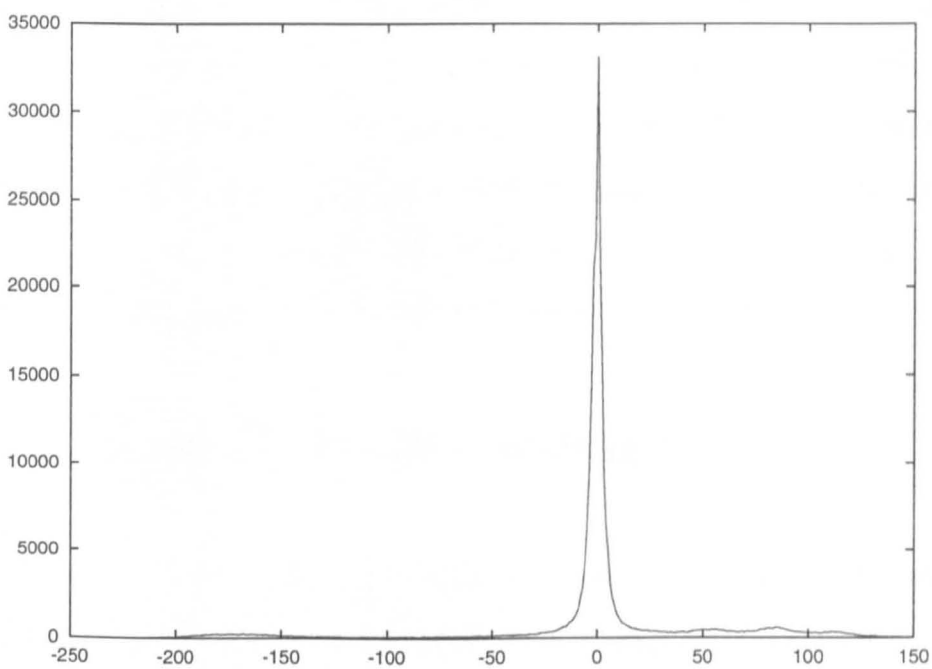
Using the filters mentioned above in an orthogonal two-dimensional filter bank, a given image can be decomposed into Haar wavelet subbands. Even using the wavelet transform as simple as Haar, there can be many wavelet coefficients having relatively very small magnitude. This is also clear from the histograms of original *Lena* image and its Haar decomposed version, as shown in Figure 3.2.

3.1.1 Towards Compression

As discussed in the previous chapter, the wavelet transform is a suitable candidate for representation in image coding: a sparse representation is obtained by taking the wavelet transform, resulting in the majority of the transform coefficients being relatively small. Ignoring some of these small coefficients, we can achieve compression – at



(*a*)



(*b*)

Figure 3.2: Gray-level histograms for the 512×512 *Lena* image
(*a*) original image, and (*b*) its 1-level Haar decomposition
Note the difference in vertical scale.

the cost of some loss of information, however. One simple way of ignoring the small coefficients is using a thresholding function, which sets as zero all the coefficients below a certain threshold. If θ denotes the threshold value and c_{ij}^k denotes the transform coefficient at the spatial location (i, j) and at transform level k, then the thresholding does as following.

$$c_{ij}^k = \begin{cases} 0 & \text{if } |c_{ij}^k| < \theta \\ c_{ij}^k & \text{if } |c_{ij}^k| \geq \theta \end{cases}$$

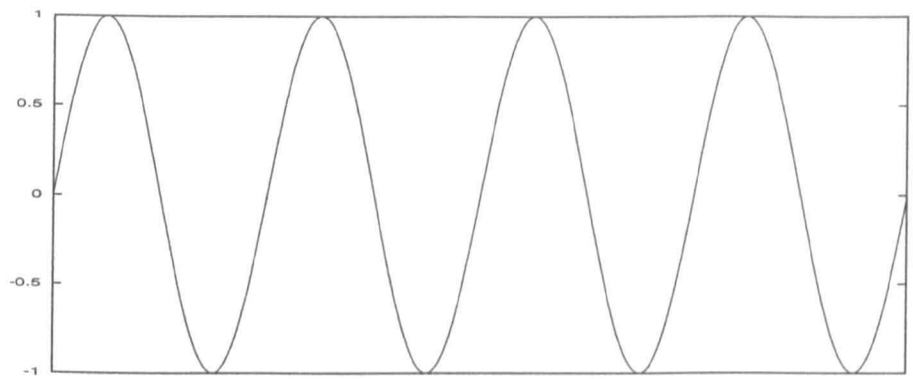
In order to see how thresholding affects the visual quality of the reconstructed image, consider Figures 3.3 and 3.4 which show a reconstruction of the 512×512 *Lena* image after taking only the first 10% and 5% of the transform coefficients (by decreasing order of the magnitude) of a 3-level Haar wavelet transform. While the first image compressed roughly by a factor of 10 has reasonably good quality, the second image, roughly providing a compression factor of 20, has many blocking artifacts. This is because the Haar wavelet and scaling functions provide a piecewise constant approximation of the given function, as shown in Figure 3.5. By removing the coefficients of small magnitude, which provide the detail, the step-nature of Haar functions emerges. When Haar functions with long support only are used to reconstruct the sine wave while ignoring those with short support, blocking artifacts appear in the signal reconstruction. This limits the use of Haar wavelet transform for image coding.

3.2 Properties of Wavelet Coefficients

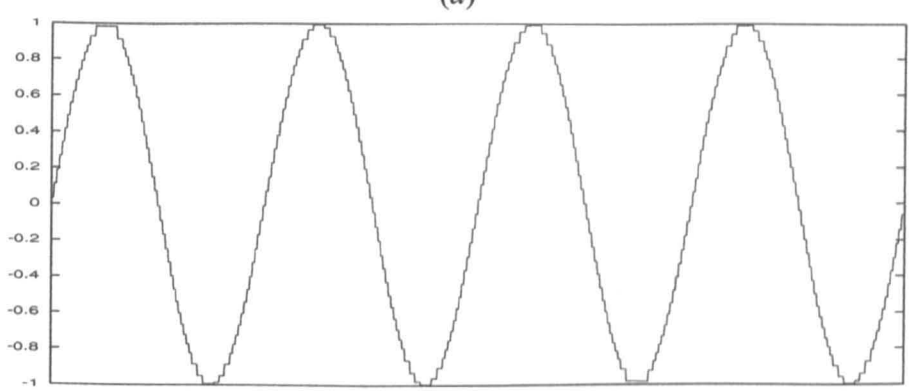
Ignoring small wavelet coefficients to achieve compression is equivalent to chopping the tail from a list of all wavelet coefficients sorted by magnitude in a descending order. The amount of compression achieved and the amount of loss incurred by doing so both depend upon the rate of decay of these coefficients. The faster the decay is, the smaller the error of reconstruction will be. It is, therefore, of some importance to study the

Figure 3.3: Reconstruction of *Lena* from the largest 10% Haar wavelet coefficients
PSNR=33.95dB

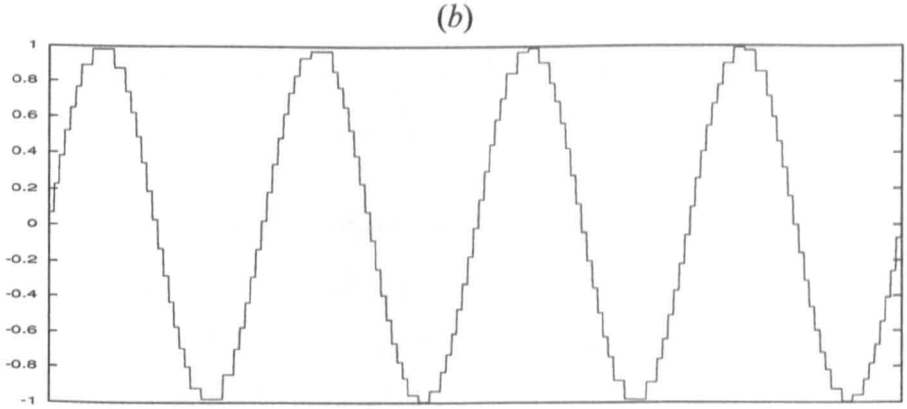
Figure 3.4: Reconstruction of *Lena* from the largest 5% Haar wavelet coefficients
PSNR=32.43dB



(*a*)



(*b*)



(c)

Figure 3.5: Reconstruction of a sine wave from Haar wavelet coefficients
(*a*) The sine wave, (*b*) its reconstruction from the largest 20% coefficients, and (*c*) its reconstruction from the largest 10% coefficients

characteristics of the decay of wavelet transform coefficients [77].

It is known that the asymptotic decay of the Fourier transform $|F_f(\omega)|$ of a signal $f(t)$ is related to the global regularity of f. If f is uniformly Lipschitz[1] α over $\mathcal{R}$, then $|F_f(\omega)|$ decays faster than $|\omega|^{-\alpha}$ [85]. The asymptotic decay of wavelet transform coefficients $|F_w(a,b)|$, however, depends upon the uniform signal regularity outside the discontinuities and remains unaffected by a finite number of discontinuities. This relates the asymptotic decay of wavelet transform coefficients to the localized regularity of signal f, due to the relatively better localization of wavelets in time, as stated in the following theorem.

Theorem 3.1 (Mallat [47]) *If $\psi(t)$ has n vanishing moments and f is uniformly Lipschitz $\alpha \leq n$ over an interval $[x,y]$, then $\exists A > 0$ such that*

$$|F_w(a,b)| \leq A\, a^{\alpha+\frac{1}{2}} \quad \forall\, (a,b) \in \mathcal{R}^+ \times [x,y]$$

The scale a in the above inequality is equivalent to the 'inverse frequency' ω^{-1} when compared to the decay of the Fourier spectrum.

Let $c_s[k]$ denote the wavelet transform coefficients, excluding the scaling coefficients, sorted by magnitude in a descending order such that $c_s[k] \geq c_s[k+1]$, $\forall\, k \geq 1$. Then the following theorem establishes that if f has bounded variation, the decay of sorted wavelet coefficients is of the order of k^{-1}.

[1] A function f is bounded and uniformly Lipschitz α over $\mathcal{R}$ if

$$\int_{-\infty}^{+\infty} |F_f(\omega)|(1+|\omega|^{\alpha})d\omega < \infty$$

Theorem 3.2 (Mallat [47]) *If* $\|f\|_v < +\infty$, *then* $\exists\, B$ *such that:*

$$|c_s[k]| \leq B \|f\|_v k^{-1}$$

We looked at the decay of sorted scaling and wavelet coefficients separately. The log-log plots of sorted scaling and wavelet coefficients using Haar and Daubechies-4 (or Daub4) [25] wavelets for *Lena* at different number of wavelet transform levels are shown in Figures 3.6–3.9. The following observations can be made from these curves:

- As the number of levels of the wavelet transform increases, the rate of decay of wavelet coefficients increases, in accordance with Theorem 3.1.
- The *dynamic range* of both scaling and wavelet coefficients increases with the increase in number of levels of the transform.
- The asymptotic slope of the decay of sorted wavelet coefficients $\log_2 |c_s[k]|$ for *Lena*, when plotted against $\log_2 k$, is -1, verifying that *Lena* behaves like a function with a bounded total variation, according to Theorem 3.2.
- In almost all cases, the rate of decay of wavelet coefficients is greater when using Daub4 filters than when Haar filters are used. There is, however, very little observable difference in the decay of scaling coefficients using these two filters.
- There are many wavelet coefficients having relatively very small magnitude. For example, there are almost 97% wavelet coefficients in a 5-level Daub4 wavelet transform of *Lena* whose magnitudes are less than 3% of the highest magnitude wavelet coefficient.

Based on these observations, a simple wavelet-thresholding image coder is developed in Section 3.3.

3.2.1 Statistical Properties of Wavelet Coefficients

It is a well known experimental fact that the statistics of high frequency wavelet (packet) coefficients and DCT coefficients can best be approximated by a Laplacian distribution for most natural images. Earlier work [59] on DCT coding schemes used the intuitive assumption that the dc and the non-dc DCT coefficients for images are more likely to follow Gaussian and Laplacian distributions respectively. Later on, these assumptions were experimentally verified to a reasonable extent by goodness-of-fit tests using l_∞-norm as a distance measure [68] between the distributions. Some researchers have also proposed generalized Gaussian distribution (GGD) for both DCT and subband coefficients in image coding applications [80]. However, it was shown recently [7] that the GGD assumption performs better than the Laplacian one by no more than 0.08 bits/pixel using an MSE distortion measure.

Based on the assumption that the high frequency wavelet coefficients follow a Laplacian distribution, with parameter λ, the distortion D and the rate R for the step size Δ can be written as[2]

$$D(\Delta) = \frac{2}{\lambda^2} - \frac{\Delta}{\lambda}\left(1 + \coth\frac{\lambda\Delta}{2}\right)e^{-\frac{\lambda\Delta}{2}} \tag{3.4}$$

$$R(\Delta) = -P(0)\log_2 P(0) - e^{-\frac{\lambda\Delta}{2}}\log_2 \sinh\frac{\lambda\Delta}{2} + \frac{\lambda}{\log 2}S \tag{3.5}$$

where $P(0) = 1 - e^{-\lambda\Delta/2}$ and S is given by

$$S = \sum_i |x_i| \cdot P(x_i) = 2\Delta \cdot \sinh\frac{\lambda\Delta}{2}\sum_{i=0}^{n} i e^{-i\lambda\Delta}. \tag{3.6}$$

[2]For a derivation of these relations, refer to Appendix A.

We generated data sets of various sizes from a memoryless Laplacian source with $\lambda = 1$. The operational rate-distortion curve was obtained by simply quantizing the data uniformly with different step size Δ values and then computing the corresponding values for distortion (MSE) and rate (order-1 entropy). To obtain a fast approximation to the operational rate-distortion curve, (3.4) and (3.5) were used to compute the (R,D) points. Both the empirical and the approximated operational curves for rate-distortion for the number of data points N=4K and N=16K are shown in Figure 3.10.

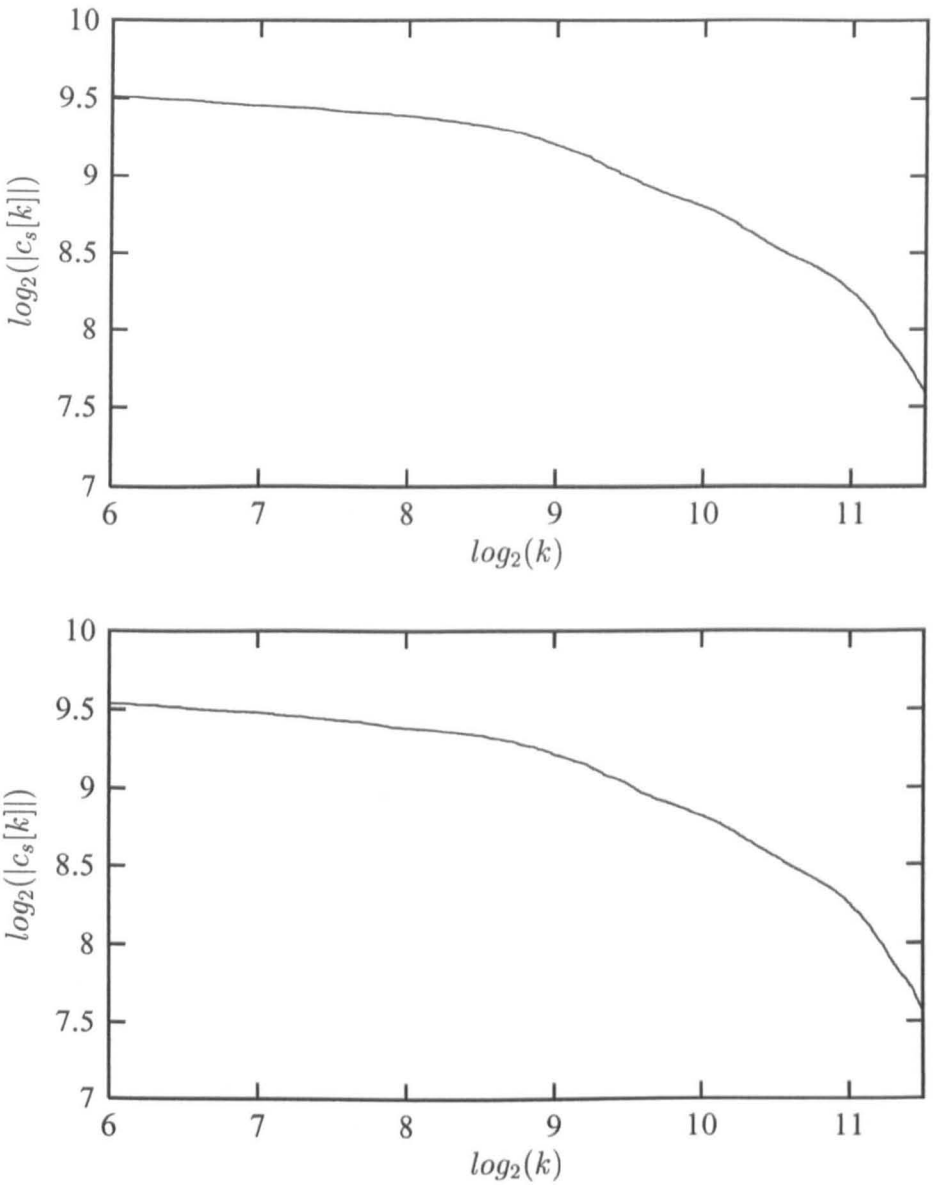


Figure 3.6: Decay of 3-level sorted scaling coefficients $|c_s[k]|$ for *Lena*
Top: Haar, and *Bottom*: Daub4

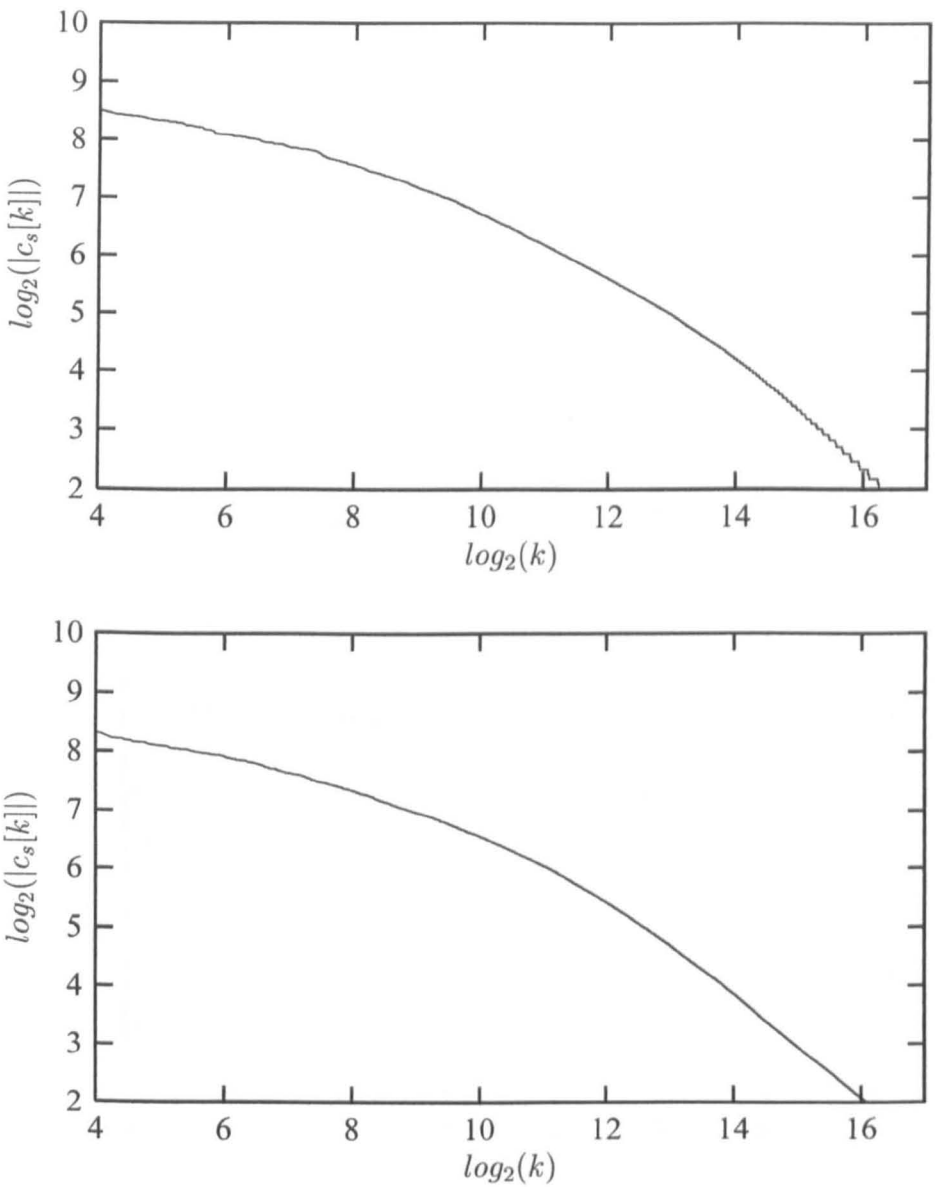


Figure 3.7: Decay of 3-level sorted wavelet coefficients $|c_s[k]|$ for *Lena*
Top: Haar , and *Bottom*: Daub4

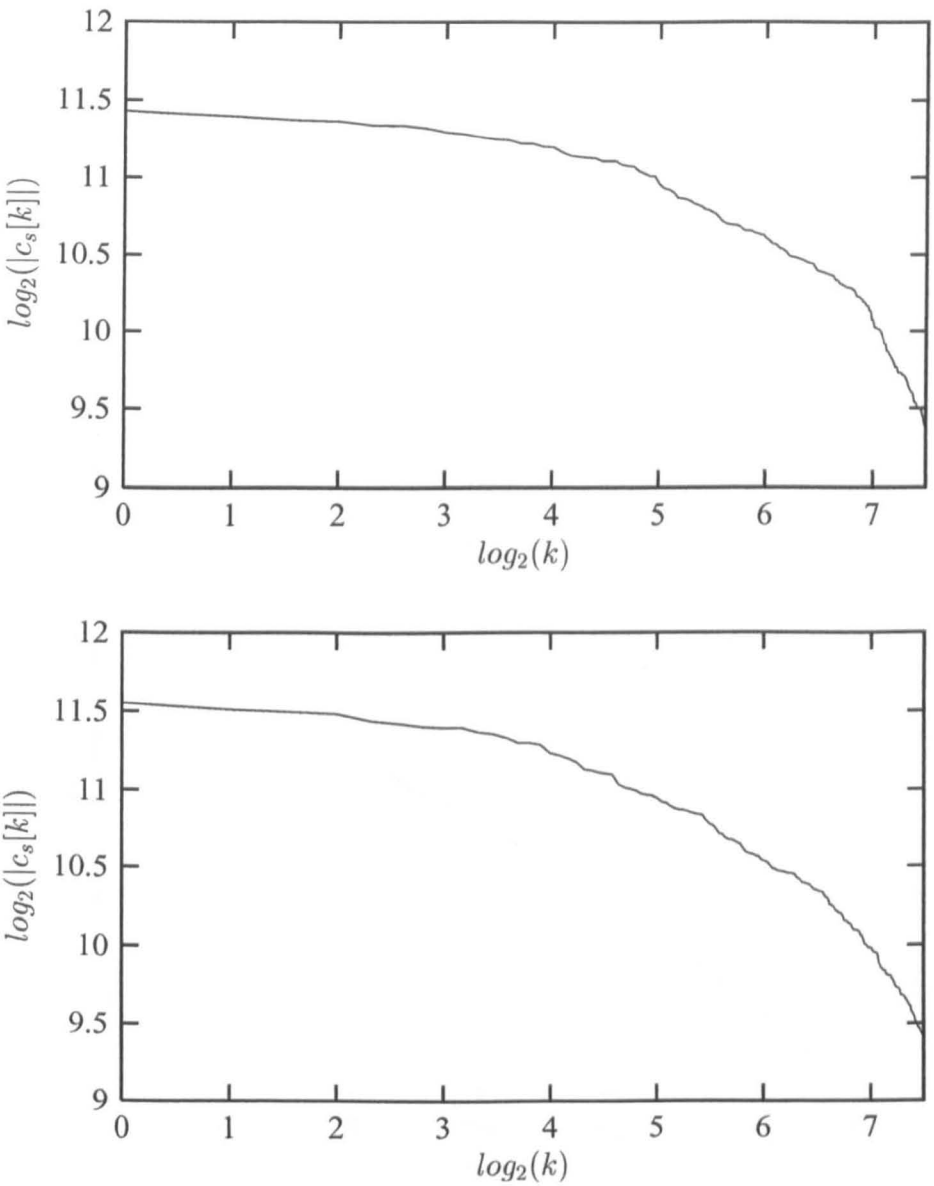


Figure 3.8: Decay of 5-level sorted scaling coefficients $|c_s[k]|$ for *Lena*
Top: Haar, and *Bottom*: Daub4

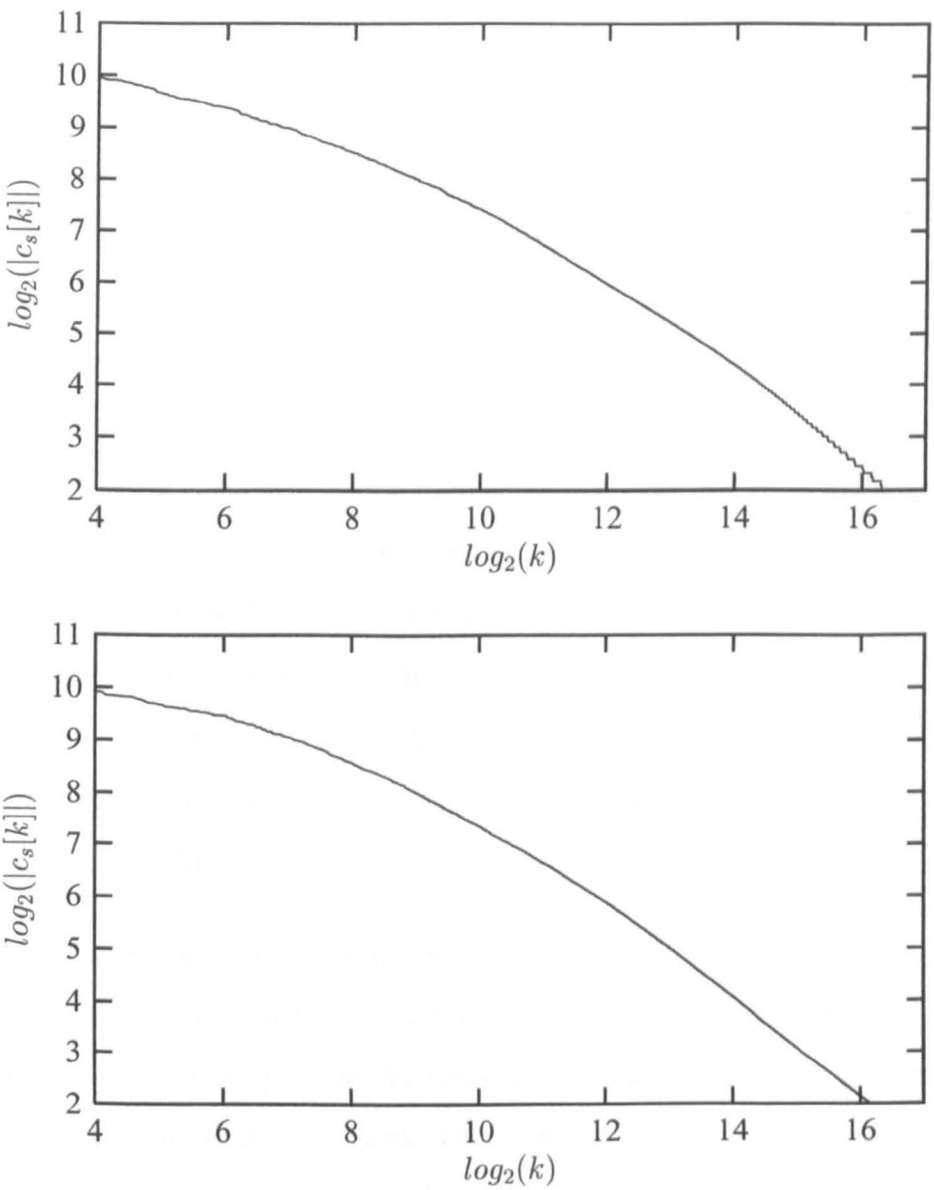


Figure 3.9: Decay of 5-level sorted wavelet coefficients $|c_s[k]|$ for *Lena*
Top: Haar, and *Bottom*: Daub4

3.3 A Simple Wavelet-Thresholding Image Coder

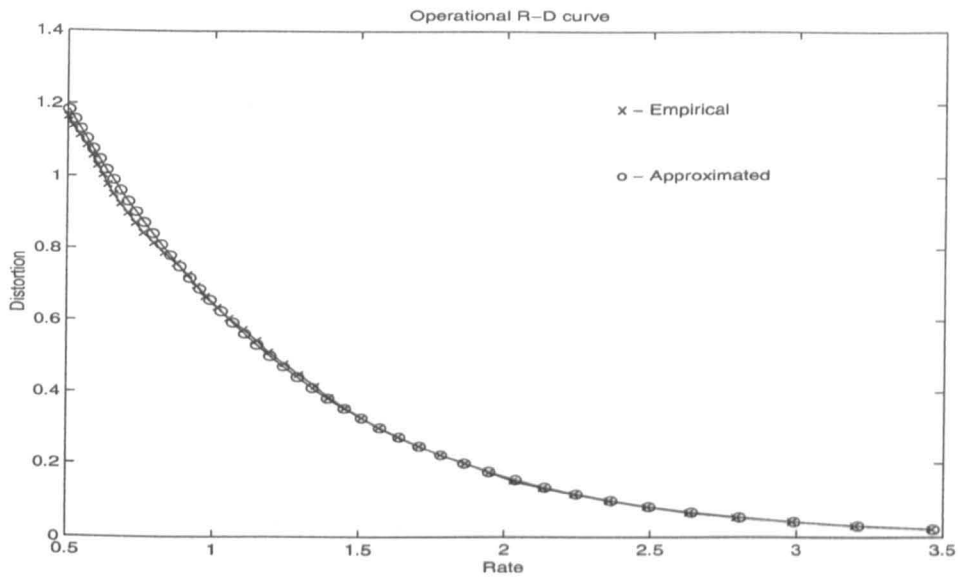
Using a thresholding function as a quantizer is a simple way of ignoring relatively small wavelet transform coefficients and thus achieving compression[3]. We assume here that the wavelet transform coefficients have zero mean, which can be easily achieved by removing the mean level from the image and then taking the wavelet transform. In order for the thresholding function to generate only integer output values, we modify it such that the new thresholding function $Q_\theta^\gamma(c)$ is given by,

$$Q_\theta^\gamma(c) = \begin{cases} \gamma(c) & \text{if } |c| \geq \theta \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

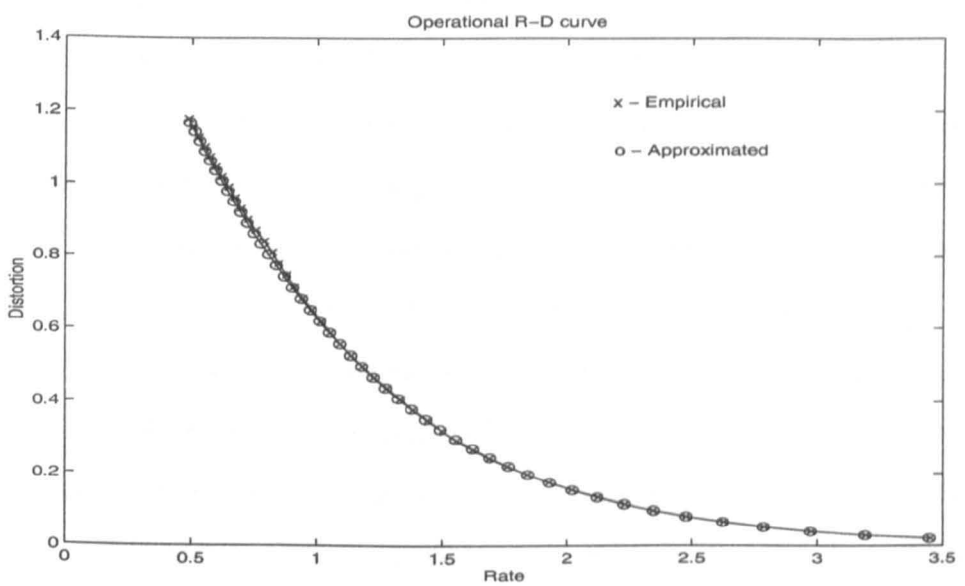
where $\gamma(\cdot)$ denotes the ordinary rounding function which returns the integer nearest to its argument value. This new thresholding function returns two kinds of output values: 0 and non-zero integer values greater than θ in magnitude. We made this simple modification so that we could feed the output of Q_θ^γ to an entropy coder directly, for which we employ two different settings. In *setting I*, the sign and absolute value of $Q_\theta^\gamma(c)$ are encoded separately, whereas all the coefficients are translated before quantization and encoding (so that they lie in the range $0 \ldots 255$, and no extra sign information is needed to be encoded) in *setting II*.

In order to make sure that the dynamic range of wavelet transform coefficients does not go below -128 and above 127 (for a zero-meaned 8-bit greyscale image), the filter coefficients are divided by $\sqrt{2}$ on the analysis side and multiplied by the same factor on the synthesis side of the filter bank. This facilitates the use of entropy coders which take as input the symbols emanating from the wavelet-thresholding source whose alphabet size is 256. The output of Q_θ^γ is fed into an entropy coder in both settings. Two entropy coders were used: the LZW-$\mathcal{FP}$ (or simply $\mathcal{FP}$ coder) which performs

[3]Thresholding the wavelet coefficients, also known as *hard thresholding*, has been shown to be quite useful for removing the white noise in denoising applications [26].



(*a*)



(*b*)

Figure 3.10: Operational rate-distortion curves for a Laplacian distribution $\lambda = 1$; (*a*) N=4K and (*b*) N=16K

optimal parsing for an LZ78-like dictionary coding scheme, and an order-1 arithmetic coder (*AC-1*) implementation similar to that of [99] which takes previous symbol as context.

Experimental results of this simple wavelet-thresholding coder for different numbers of wavelet transform levels N_l, using Haar and Daub4 filters, are given in Table 3.3. The compression performance is measured in terms of bit rate (bits per pixel or bpp.) and PSNR (decibels or dB). For a threshold $\theta = 0$, the quantizer function Q_θ^γ is same as $\gamma(\cdot)$ or the rounding function. The subbands were encoded in increasing frequency order – i.e. the lowest frequency subband was encoded first, followed by the high frequency bands (HL, LH, and HH) at transform level N_l, followed by high frequency bands at level $N_l - 1$, and so on. The bit rate and PSNR values shown in Table 3.3 are computed after encoding all the $3N_l + 1$ subbands. It is, however, possible to stop the encoding procedure at any point when a desired bit rate or PSNR has been achieved. This is known as *progressive* or *embedded* encoding and will be used in our other coding algorithms. It is clear from these results that the arithmetic coder *AC-1* outperforms the dictionary coder $\mathcal{FP}$ for all the experiments, verifying that there are generally more statistical than structural redundancies in the output of a simple thresholding quantizer for wavelet coefficients, as one would expect. At any given PSNR, the coder setting II always yields better results than the coder setting I by generating fewer bits. This is because the coefficients' sign is nearly random in nature. In our experiments for coder setting I, the number of bits spent on separate encoding of sign was 20–30% of the total bit budget, which also the included number of bits spent on encoding the magnitude of quantizer output for all the subbands.

The Daub4 filters came out as better performers than the Haar filters for all the experiments. This verifies that a steeper decay generally corresponds to a better compression performance. However, there is a catch here! A steeper decay also means removal of

more coefficients by Q_θ or Q_θ^γ for a non-zero value of θ. This will naturally result in worse performance in terms of PSNR, as can be seen from these results for $\theta = 1$ and $\theta = 2$. It is also to be noted that the difference between Haar and Daub4 PSNR values for non-zero θ decreases as the number of transform level increases. Also, there is almost no reduction in bit rate when going from a 5-level to a 6-level transform for both Haar and Daub4 wavelets, but there is a considerable loss (0.5–0.7dB) in the PSNR of the decoded image. This is also clear from the fact that there is no significant difference between the information cost of these two transforms, as shown in Table 3.3, where information cost C is defined as [22]

$$C = \sqrt{\sum_c c^2 \log c^2}. \tag{3.8}$$

In the end, the bit rate vs. PSNR curves for these experiments, shown in Figure 3.11, show that although there is a plenty of room for improvement, a PSNR value of 44.75dB for just over 2.5 bpp. (a compression factor of over 3) is comparable to 44.18dB at 2.6 bpp. obtained by the standard JPEG algorithm[4]. It is interesting to note that the Daub4+$\mathcal{FP}$ combination outperforms the Haar+*AC-1* combination for some instances.

3.4 Exploiting Self-Similarities Among the Subbands

There are two main factors contributing to the efficiency of wavelet coding. First, the presence of many near-zero values (or *insignificant* information) encourages one to apply an entropy coding technique to eliminate this redundancy after quantization. Secondly, a closer look at the wavelet transform coefficients reveals that there are certain dependences between the wavelet coefficients in a coarser scale high frequency

[4] generated by the popular *xv* program

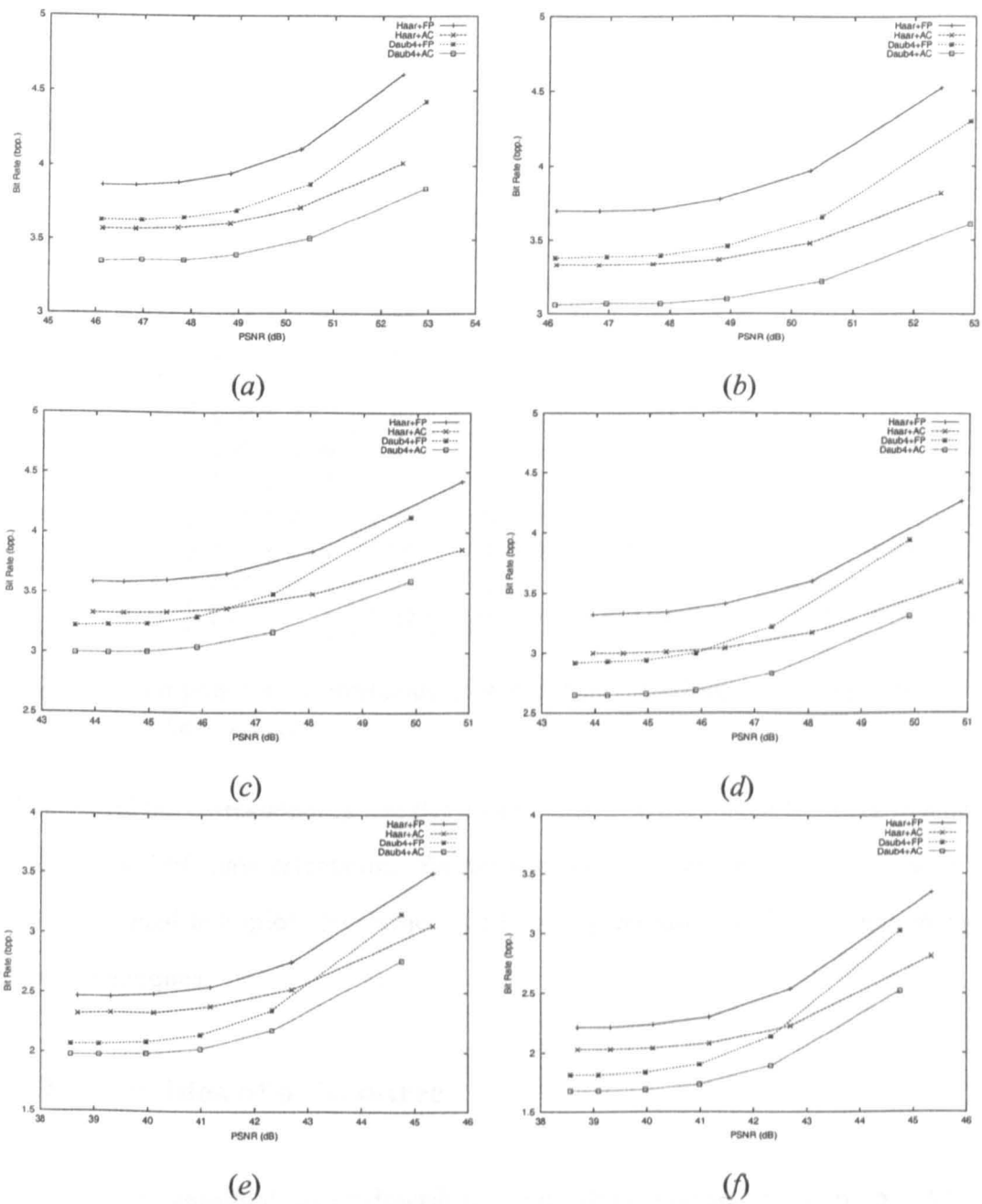


Figure 3.11: Performance of the wavelet-thresholding image coders for *Lena* (*a*) $\theta = 0$ (setting I), (*b*) $\theta = 0$ (setting II), (*c*) $\theta = 1.0$ (setting I), (*d*) $\theta = 1.0$ (setting II), (*e*) $\theta = 2.0$ (setting I), and (*f*) $\theta = 2.0$ (setting II)

Quantizer Threshold θ	N_l	*Bitrate* (Haar)				*PSNR* (dB)	*Bitrate* (Daub4)				*PSNR* (dB)
		Setting I		*Setting II*			*Setting I*		*Setting II*		
		FP	*AC-1*	*FP*	*AC-1*		*FP*	*AC-1*	*FP*	*AC-1*	
0.0	1	4.61	4.02	4.52	3.82	52.44	4.43	3.85	4.30	3.61	52.92
	2	4.11	3.72	3.97	3.48	50.29	3.88	3.51	3.66	3.22	50.49
	3	3.95	3.61	3.78	3.37	48.81	3.70	3.40	3.46	3.10	48.94
	4	3.89	3.58	3.71	3.34	47.72	3.65	3.36	3.40	3.07	47.84
	5	3.87	3.57	3.70	3.33	46.83	3.63	3.36	3.39	3.07	46.96
	6	3.87	3.57	3.70	3.33	46.13	3.63	3.35	3.38	3.06	46.11
1.0	1	4.42	3.86	4.26	3.59	50.86	4.13	3.60	3.94	3.31	49.89
	2	3.84	3.49	3.60	3.17	48.06	3.49	3.17	3.22	2.83	47.31
	3	3.66	3.37	3.41	3.04	46.44	3.30	3.05	3.00	2.69	45.89
	4	3.61	3.34	3.34	3.01	45.33	3.24	3.01	2.94	2.66	44.96
	5	3.59	3.33	3.33	3.00	44.52	3.23	3.00	2.93	2.65	44.24
	6	3.59	3.33	3.32	3.00	43.95	3.22	3.00	2.92	2.65	43.62
2.0	1	3.50	3.06	3.34	2.81	45.34	3.16	2.77	3.02	2.52	44.75
	2	2.76	2.53	2.54	2.23	42.69	2.36	2.19	2.14	1.89	42.32
	3	2.55	2.39	2.30	2.08	41.17	2.15	2.03	1.90	1.73	40.98
	4	2.49	2.34	2.24	2.04	40.11	2.09	1.99	1.83	1.69	39.97
	5	2.47	2.34	2.22	2.03	39.31	2.07	1.98	1.81	1.68	39.09
	6	2.47	2.33	2.22	2.03	38.69	2.07	1.98	1.81	1.68	38.56

Table 3.1: Compression performance of wavelet-thresholding coding algorithms for the 512×512 *Lena image*

subband and the corresponding wavelet coefficients, at same spatial location, in a finer scale subband of same orientation. Earlier wavelet compression techniques, such as [100, 4], tended to exploit the former fact by using certain quantization and entropy coding techniques.

3.4.1 The Idea of a Zero-tree

A zero-tree (or zerotree) can be described as a quadtree representing a group of transform coefficients which are insignificant. The history of quadtrees in coding dates back to Wilson's work on quadtree predictive coding [95]. The predictive coder of [95] recursively builds a spatial quadtree by taking a linear combination of the four

N_l	C ($\times$1000)	
	Haar	*Daub4*
1	36.06	36.12
2	18.11	18.06
3	9.62	9.33
4	6.11	5.50
5	5.01	4.18
6	4.74	3.85

Table 3.2: Information cost of N_l-level wavelet coefficients for the *Lena*

neighbouring pixels to represent coefficients that are above a certain threshold. Lewis and Knowles [41] hypothesized that in this tree-like structure, if the magnitude of a parent coefficient is below a given threshold, all of its children coefficients are most likely to follow this course too. A zerotree combines wavelet coefficients belonging to different spatial frequency subbands, as explained in the following section. Various extensions of zerotree quantization, such as [28, 102], have been proposed since its introduction.

3.4.2 Shapiro's EZW Compression

The following are the main features of Shapiro's embedded zerotree wavelets (EZW) image coding algorithm [73].

a) It exploits the self-similarities across different scales of an image wavelet transform. This is done with the help of zerotrees. In this quadtree data structure, every node (except the leaves) has four children. A *zerotree root* node means that the information in wavelet transform coefficients corresponding to all the nodes in this tree is insignificant, with respect to a particular threshold. It means that we do not need to encode the whole tree and encoding only the root of this tree would do the job, thus providing compression. This is based upon Shapiro's hypothesis: *if a wavelet coeffici-*

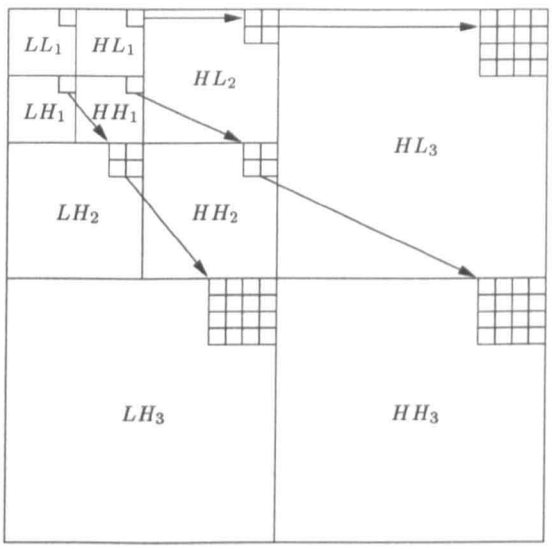


Figure 3.12: Parent-offspring dependences in a 3-level DWT

ent at a coarser scale is insignificant, all wavelet coefficients at the same orientation and at the same spatial locations at the finer scales are likely to be insignificant. In order to achieve very high compression, the probability of the most likely symbol after quantization, i.e. the zero symbol, must be extremely high. Figure 3.12 shows these parent-offspring dependences. The order in which the coefficients are scanned and then coded is shown in Figure 3.13. The scanning starts from the lowest-frequency subband LL_3 (for a 3-level DWT) and ends at HH_1. All positions in a subband are scanned before the scan moves to the next subband, and all the parents are scanned before their children. Obviously, this scanning order is employed by both the coder and the decoder.

b) The *significance map*, which outlines the positions of significant wavelet coefficients, is encoded during the *dominant pass*, using the following codewords in the scan order mentioned above:

i) POS (positive significant),

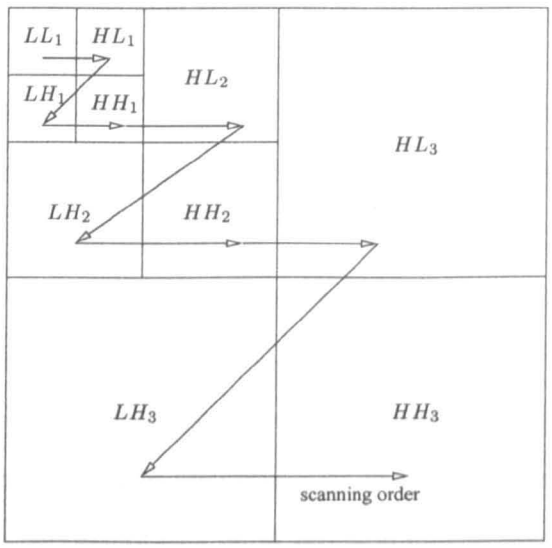


Figure 3.13: Scanning order of a 3-level DWT's coefficients

ii) NEG (negative significant),

iii) IZ (isolated zero/insignificant), and

iv) ZTR (root of a zerotree).

After every dominant pass, the *subordinate pass* encodes single bit information for the decoder to decide whether a wavelet coefficient which has already been detected to be significant is in the upper half or in the lower half of the uncertainty interval.

c) The zerotree approach had previously been used by other researchers in their work [41] on image and video compression. The main thrust of EZW was *embedded coding* of the image data which makes possible the progressive coding (reconstruction) of an image by the coder (decoder) and the precise control of target bit rate.

Embedded coding was achieved by the *successive approximation quantization* (SAQ) in EZW. The threshold is halved after each execution of dominant and subordinate passes. This corresponds, in a sense, to the bit-plane coding of the wavelet coefficients.

3.4.3 Validating the Zerotree Hypothesis

The EZW coder of Shapiro produced results which were significantly better than prevailing coding methods of the time, verifying the success of zerotree hypothesis and proving that the symmetry across wavelet subbands was worth exploiting for coding purposes. An alternate approach to test the success of the zerotree hypothesis is to compute the conditional probability $p(x|y)$ of the significance of a child coefficient x given the significance of its parent coefficient y for a specific threshold value. We plotted both joint and the conditional histograms to see how successfully the prediction for insignificance of a coefficient in a child subband can be made, given the insignificance of its parent coefficient. The conditional histograms for absolute values of wavelet transform coefficients belonging to various wavelet subbands and their immediate children are shown in Figures 3.14–3.15, where the subband naming convention used is the same as that of Figure 3.13. The concentration of these histograms in the low significance range of child coefficients given the low significance of a parent coefficient for subbands of similar orientation is an empirical verification of the zerotree hypothesis. The success of the zerotree hypothesis, and thus the performance of zerotree encoding, depends largely upon the mutual information between *same orientation* subbands for a given image. The larger the value of this measure, the more successful the zerotree hypothesis is going to be and thus more efficient the encoding will be.

3.4.4 Set Partitioning in Hierarchical Trees (SPIHT)

Shapiro's EZW coding scheme generates an embedded bitstream. It achieves low bit rate image compression and beats the standard image coding technique JPEG, in terms of PSNR at a desired bit rate. Although the codeword for a zerotree root (ZTR) is useful in avoiding the encoding of insignificant information, it was observed that the

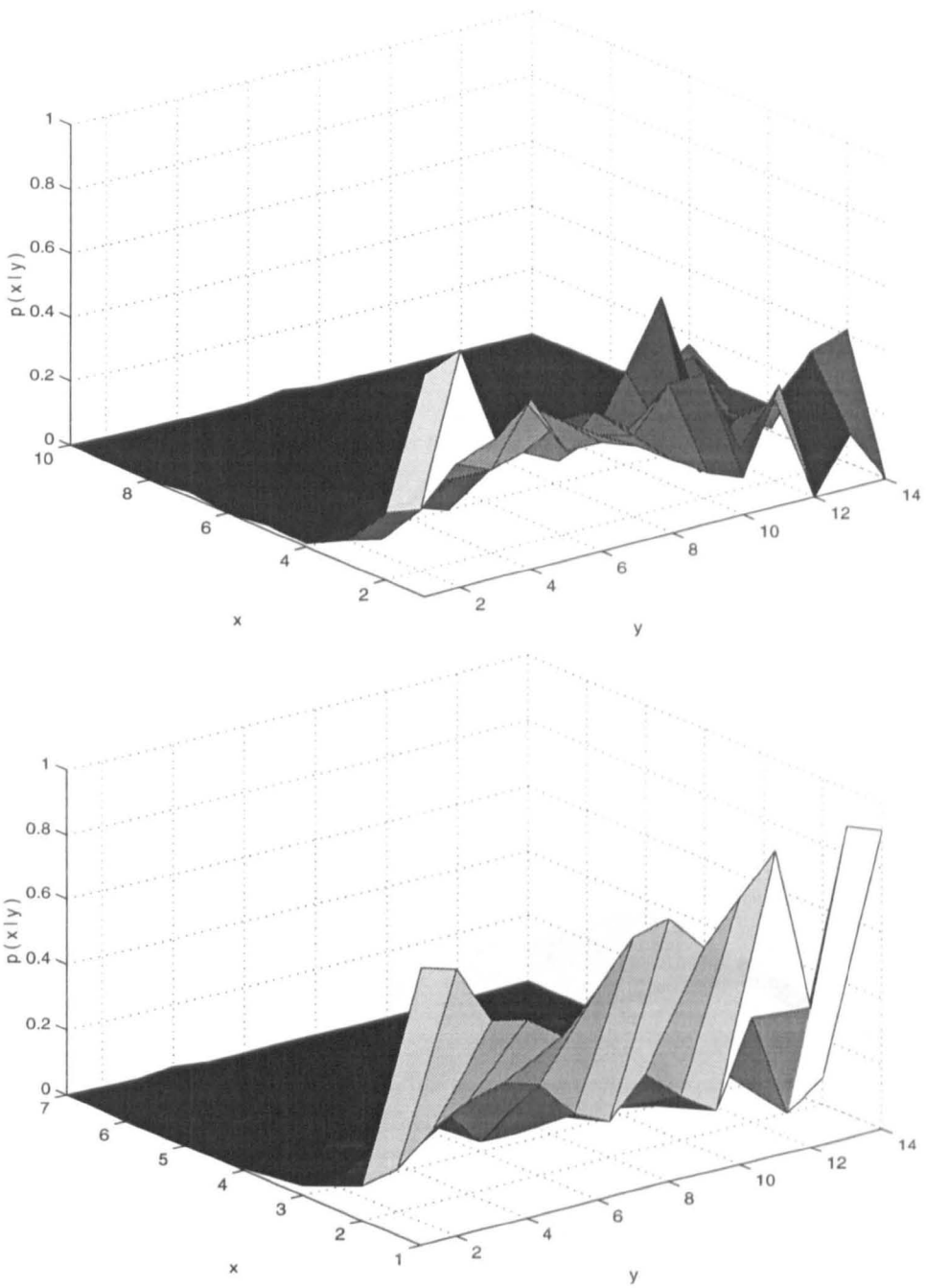


Figure 3.14: Conditional histograms for 5-level wavelet decomposition of *Lena* - I *Top*: $x \in HL_2$, $y \in HL_1$, $\Delta = 80$, and *Bottom*: $x \in LH_2$, $y \in HL_1$, $\Delta = 80$.

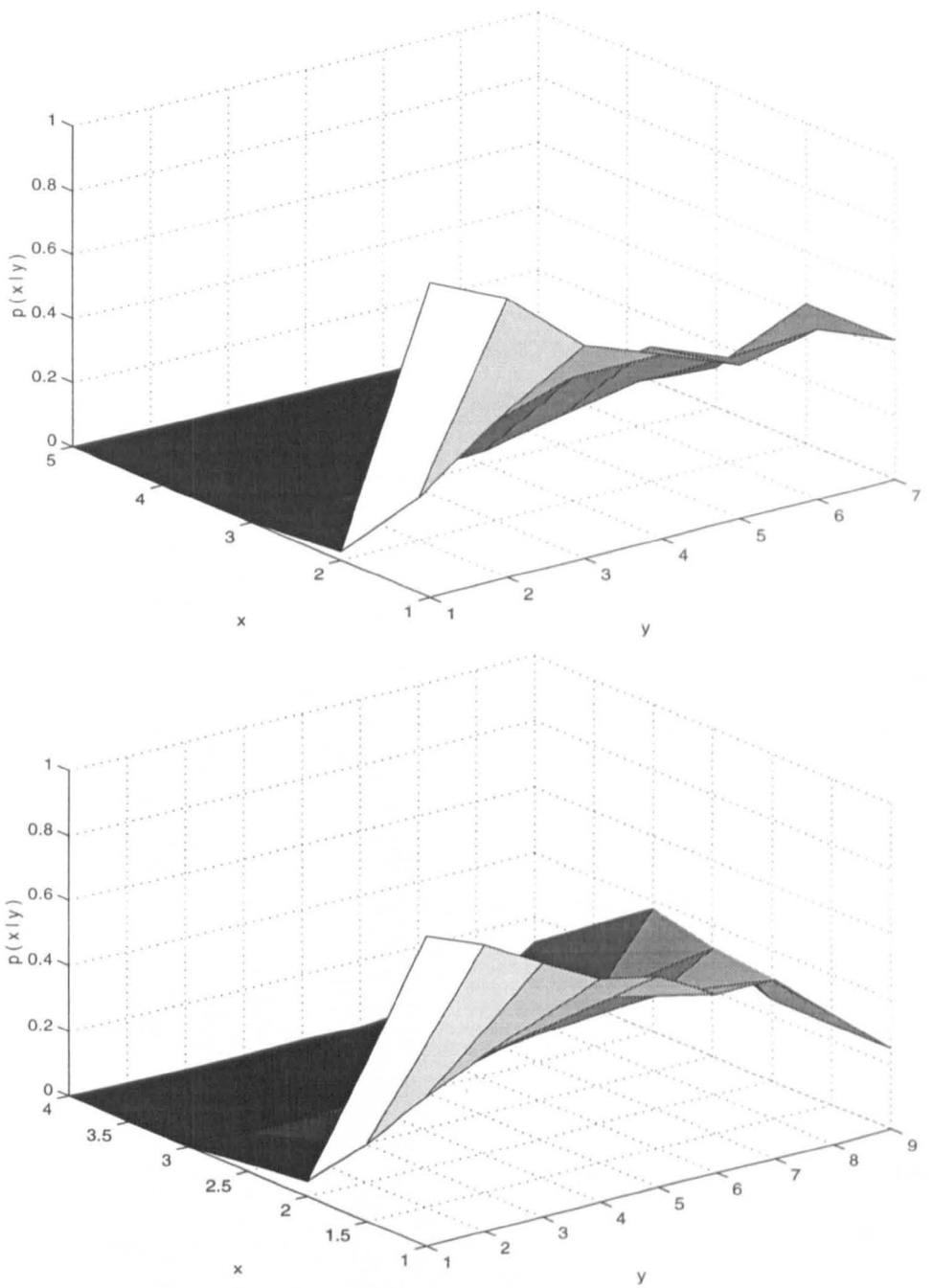


Figure 3.15: Conditional histograms for 5-level wavelet decomposition of *Lena* - II
Top: $x \in LH_4$, $y \in LH_3$, $\Delta = 50$, and *Bottom*: $x \in HH_5$, $y \in HH_4$, $\Delta = 10$.

bitstream generated by EZW contains redundant information. As described earlier, if a coefficient at a particular coordinate is insignificant, with respect to an absolute threshold, but is not a zerotree root, it is encoded as IZ. It was found that there are a lot of IZ codewords in the EZW-generated bitstream. This means that EZW spends a significant part of the bit budget in encoding insignificant information.

The set partitioning in hierarchical trees (SPIHT) image compression scheme of Said and Pearlman [71] could be regarded as an extension of EZW. It eliminates the redundancies in EZW-generated bitstream to some extent. The SPIHT coding algorithm employs two concepts from EZW: parent-offspring dependences across scale, and the bit-plane coding of significant wavelet coefficients during the refinement stages.

The spatial orientation trees of SPIHT are similar to the zerotrees of EZW. Both the coder and decoder maintain three lists: a list of insignificant sets (*LIS*), a list of insignificant pixels (*LIP*), and a list of significant pixels (*LSP*). The following are the four kinds of sets used in SPIHT: the set $\mathcal{H}$ denotes the set of all coordinates in the highest pyramid level. The set $\mathcal{O}(i,j)$ denotes the set of coordinates of all immediate offspring of the node (i,j), whereas the set of all descendants of node (i,j) is denoted by $\mathcal{D}(i,j)$. Another type of set is $\mathcal{L}(i,j)$ which is defined as $\mathcal{L}(i,j) = \mathcal{D}(i,j) - \mathcal{O}(i,j)$. All the three sets $\mathcal{O}(i,j)$, $\mathcal{D}(i,j)$ and $\mathcal{L}(i,j)$ make use of parent-child dependences among subbands across the scales.

The sorting and refinement stages employed in the SPIHT are a replica of the dominant and subordinate passes of the EZW. The sorting stage of the SPIHT coder, like the dominant pass of an EZW coder, sends information regarding the significance of coefficients to its decoder. The only difference is that EZW employs the scanning order described in Section 3.4.2, whereas SPIHT utilizes set partitioning rules to dictate the scanning order. The refinement stage of SPIHT, like the subordinate pass of the

EZW, refines the magnitude available to the decoder.

We note that the set partitioning rules used in SPIHT are essentially the same as those implicit in EZW. The initial partition adds all coordinates (i, j) to the list *LIP* and $\mathcal{D}(i, j)$ to the list *LIS*, where (i, j) are the coordinates belonging to $\mathcal{H}$. This is exactly how the EZW coder starts. The SPIHT coder then proceeds as follows. If found significant, the set $\mathcal{D}(i, j)$ is partitioned into the four offspring of (i, j) and the set $\mathcal{L}(i, j)$. The set $\mathcal{L}(i, j)$, if significant, is partitioned into four sets $\mathcal{D}(k, l)$, where (k, l) belongs to the set $\mathcal{O}(i, j)$. Both of these partitioning rules are quite like the ones assumed during the dominant pass of EZW, although in a different scanning order.

Our findings are that the following factors contribute towards the improved compression performance of SPIHT over EZW:

1. One bit is required, in SPIHT, to encode whether or not a set of coordinates is significant, with respect to an absolute threshold. This helps SPIHT eliminate the redundancies in an EZW-generated bitstream, discussed earlier in this section.

2. As reported in [71], SPIHT uses the 9-7-pair biorthogonal wavelet[5] filters of [17]. An improvement in the compression performance is inevitable, due to the better frequency localization properties of these filters, as compared to the 9-tap near-orthogonal wavelet filters of [1], which are used in EZW.

3. Like EZW, the adaptive arithmetic coding used in SPIHT also exploits the statistical dependence between the significance of a pixel and the significance of all of its descendants. The SPIHT image compression scheme also exploits the dependences between the magnitudes of adjacent pixels by keeping together a

[5] In case of biorthogonal wavelets, the detail space W_j is not orthogonal to the approximation space V_j.

group of 2×2 adjacent pixels and then using an *order-m* arithmetic coder, where m is the number of insignificant pixels in that group.

3.4.5 Augmented Zerotree Image Coder (AZIC)

Since Shapiro introduced the embedded coding of images using the zerotrees of the wavelet coefficients, there have been many suggestions for the improvement of its compression performance. The SPIHT algorithm is an extension of EZW, as discussed in the previous section. The efficiency of EZW can be improved by introducing unique scanning directions for each of three types of high frequency subbands and using a higher-order arithmetic coder [5].

We propose a progressive image coding algorithm which is based upon *augmented zerotrees* of wavelet coefficients. The augmented zerotrees are different from the spatial orientation trees of SPIHT in two ways. An augmented zerotree helps us to exploit the dependences between the coefficients in the lowest frequency subband and the corresponding coefficients at the same spatial location in three coarsest scale high frequency subbands. Secondly, it takes into account the zerotree root nodes which are not insignificant themselves. The algorithm employs a recursive detection of the augmented zerotrees by the coder in an efficient manner. This method is memory efficient since it does not require the maintenance of more than one list of coefficients. There is no sorting required for this list by the coder or the decoder. The same list helps to deal with the coefficients already detected, so as to reduce the distortion at a target bit rate. A new scanning order is introduced to exploit the inter-band dependences between the wavelet coefficients.

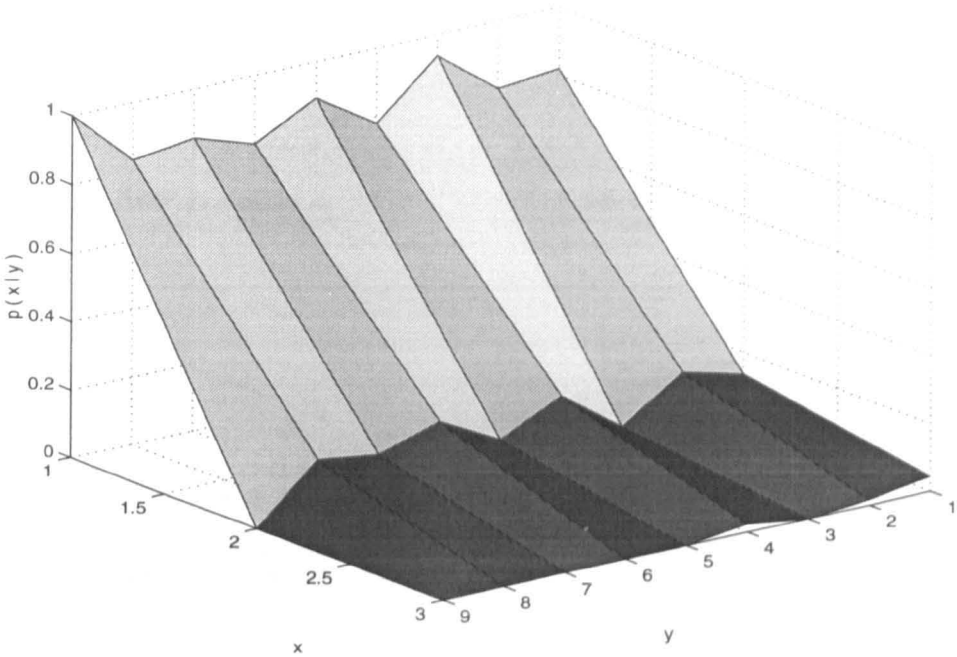


Figure 3.16: Conditional histograms for 5-level wavelet decomposition of *Lena* - III $x \in LH_1, y \in LL_1, \Delta = 350$.

The Augmented Zerotrees

In addition to Shapiro's hypothesis described in Section 3.4.2, a second hypothesis defines the augmented zerotrees as follows: *if a coefficient at the lowest frequency subband is insignificant, it is more likely that the corresponding coefficients at the same spatial location in three coarsest scale high frequency subbands would be insignificant as well.* This hypothesis was proved empirically, as illustrated in Figure 3.16. It was observed that the improvement is greater for the initial coding stages that correspond to a higher value of the threshold. As the threshold decreases, after a certain point, the improvement in the bit rate starts declining. This implies that using these dependences would help in improving the compression performance at low bit rates.

We augment the symbol set of the entropy coder by adding two more symbols to it,

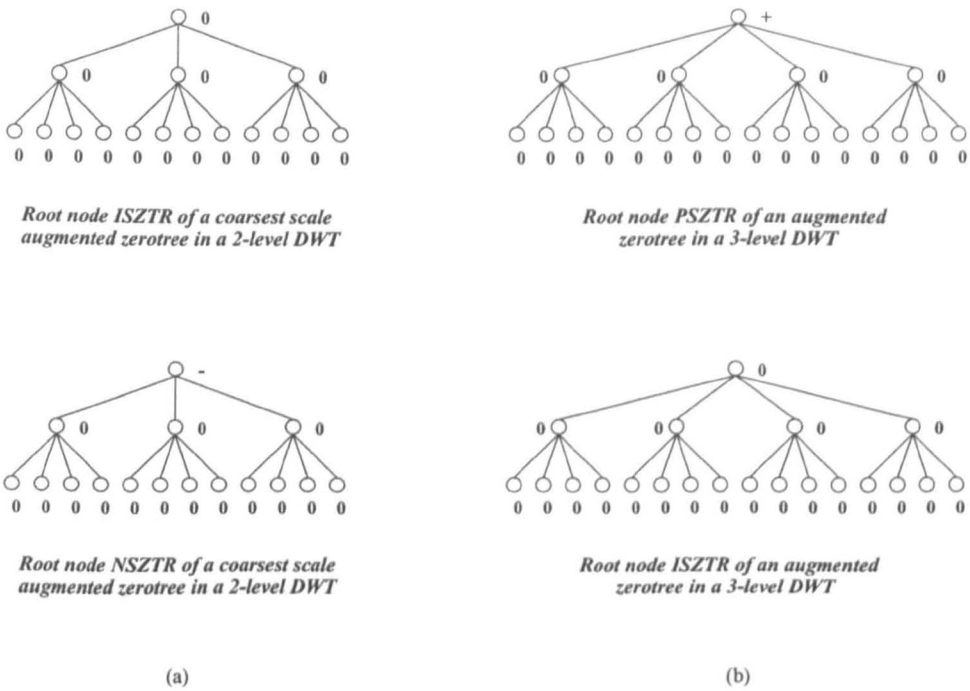


Figure 3.17: A graphical illustration of the augmented zerotrees

to take into account the zerotree root nodes which are not insignificant themselves. This helps in reducing the generation of redundant information for the zerotree root node which is not insignificant itself. Therefore, the augmented symbol set consists of the symbols POS, NEG, IZ, ISZTR (similar to the ZTR in EZW, which represents a zerotree root node which is insignificant itself as well), and two new symbols PSZTR, and NSZTR which represent zerotree root nodes, that are either positive significant or negative significant respectively.

It should be noted that this algorithm does not use an explicit zerotree data structure. The construction of zerotrees, as described in Section 3.4.5, is done in an efficient recursive manner without using any zerotree data structure. A graphical illustration of the augmented zerotrees is shown in Figure 3.17.

The Coding Stages

The compression algorithm runs two stages iteratively, for every new value of the threshold. These are: the detection stage and the fine-tuning stage. The *detection stage* determines the significance of coefficients, detects the augmented zerotrees, and encodes the significance map using the new symbol set, described in Section 3.4.5. The *fine-tuning stage* refines the values of coefficients already detected by sending single bits in a bitplane coding fashion. These stages are described in greater detail, in Section 3.4.5.

The List of Detected Coefficients (LDC)

There is an issue of what to do with a coefficient that has already been detected as significant during one of the previous detection stages. One solution is to assign it the value zero, immediately after it has been encoded as being significant, to keep a copy of its coordinates and actual value for further fine tuning stages, and to ignore it during further detection stages. This approach has the disadvantage that if the coefficient is ignored, there is a risk of losing the opportunity of finding a zerotree at the coordinates of this coefficient. Our approach is not to ignore this coefficient during further detection stages, in order to exploit the chances of occurrence of a zerotree with this coefficient as root of the zerotree. It was observed that the improvement using this approach is greater for low bit rates, mainly due to the fact that the approach succeeds in finding more zerotrees. This improvement seems to cease at high bit rates, or at low values of the threshold, when the finest scale coefficients start getting detected.

In order to implement the above idea, a list of detected coefficients $\mathcal{D}$ is maintained. It contains the coordinates and original value of the coefficient detected as significant. During every detection stage, entries are added to this list in the scanning order de-

scribed later in Section 3.4.5.

It is to be noted that the order in which the entries are added to $\mathcal{D}$ in the decoder should be the same as the one employed by the coder. Our coding algorithm requires no other list but $\mathcal{D}$, as compared to two lists (dominant and subordinate lists) in EZW and three lists (*LIS*, *LIP*, and *LSP*) in SPIHT. No sorting operation is performed on the list $\mathcal{D}$.

Detection/Construction of the Augmented Zerotrees

This part of the algorithm looks for augmented zerotrees (if any) and marks the significance of all of its elements, except the root node, as DNC (Do Not Code). All such coefficients, whose significance has been marked as DNC, are ignored during the coding part of the detection stage. The augmented zerotree detection algorithm, employed by the coder, is as follows:

ALGORITHM – I

boolean *IsZerotree*$(x, y, Level)$

1. if $Level \rightarrow FinestScale$

   `if` *Significance*$(x, y) \rightarrow IZ$

   `return` *TRUE*

   `else`

   `return` *FALSE*

2. `if` *IsZerotree*$(i, j, Level + 1), \forall\ (i, j) \in \mathcal{O}(x, y)$

   Significance$(\mathcal{O}(x,y)) \leftarrow DNC$

   `if` *Significance*$(x, y) \rightarrow IZ$

Significance$(x, y) \leftarrow$ *ISZTR*

return *TRUE*

if *Significance*$(x, y) \rightarrow$ *POS*

Significance$(x, y) \leftarrow$ *PSZTR*

return *FALSE*

if *Significance*$(x, y) \rightarrow$ *NEG*

Significance$(x, y) \leftarrow$ *NSZTR*

return *FALSE*

else

return *FALSE*

A short note about the decoder is included here. The decoder operates in the same sequence. When the decoder reads one of the augmented zerotree root nodes, it constructs the zerotree in the same recursive manner. Following is the algorithm for constructing the augmented zerotrees when the decoder reads one of these symbols: ISZTR, PSZTR, or NSZTR.

ALGORITHM – II

ConstructAugmentedZerotree$(x, y,$ *Level*$)$

1. if *Level* $\rightarrow$ *FinestScale*

   Significance$(x, y) \leftarrow$ *DNC*

   return

   else

   Significance$(i, j) \leftarrow$ *DNC*, $\forall\ (i, j) \in \mathcal{O}(x, y)$

2. *ConstructAugmentedZerotree*(i, j), $\forall\ (i, j) \in \mathcal{O}(x, y)$

Subband Dependent Scanning Order

The high frequency subbands of an image represent the edges at a particular orientation and scale. As discussed earlier, the subbands $\mathbf{HL_k}$, $\mathbf{LH_k}$, and $\mathbf{HH_k}$ represent the edges in horizontal, vertical, and diagonal directions respectively. It seems reasonable, therefore, to encode these subbands in an order compatible with their respective orientations.

We propose the scanning order illustrated in Figure 3.18. According to this approach, the $\mathbf{HL_k}$ subbands are scanned in the horizontal direction, the $\mathbf{LH_k}$ subbands are scanned in the vertical direction, and the $\mathbf{HH_k}$ subbands are scanned in the diagonal direction. The diagonal scan is similar to the one employed by JPEG.

Embedded Coding of the Augmented Zerotrees

A simple rule for generating the embedded bitstream is that the more significant information should be encoded before the less significant. It is to be noted that if $\mathbf{T}$ is a unitary transform, then encoding the larger magnitude coefficients before the smaller magnitude ones decreases the MSE between the original and the reconstructed image. This corresponds to the bit-plane method for progressive transmission [63].

In this coder, uniform threshold quantization is applied to the wavelet coefficients using a fixed threshold across all the subbands. A coefficient is said to be *insignificant* if its absolute value is below a certain threshold, and it is called *significant* otherwise. The benefit of applying an absolute fixed threshold across all the subbands is twofold: first, it ensures that a coefficient in a lower energy subband is not ignored due to its energy; secondly, it produces many zerotrees, thus providing more compression. The *detection* and the *fine-tuning* stages follow the thresholding. The output is then entropy coded

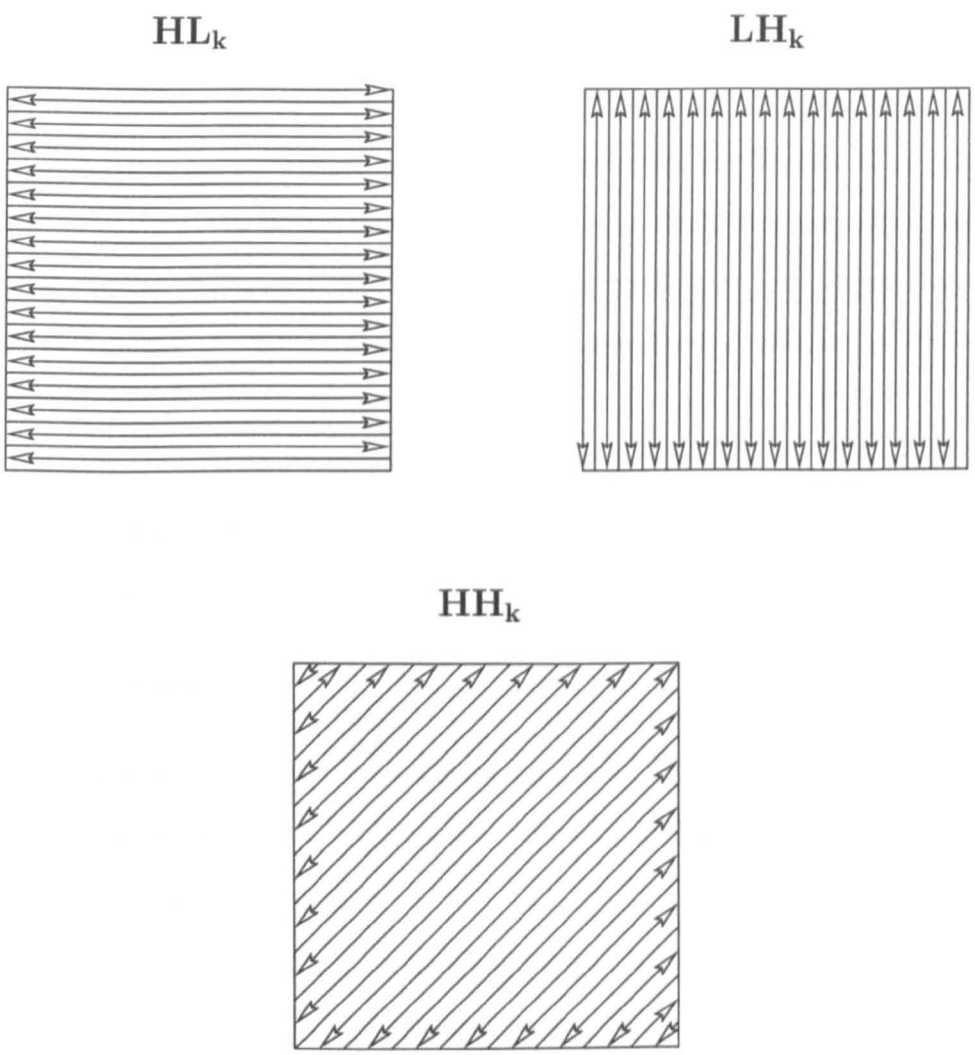


Figure 3.18: New scanning order for high frequency subbands

using a context-based adaptive arithmetic coder. The threshold is halved after every stage and this cycle – quantization, detection, fine-tuning, and entropy coding – is iterated until a target bit rate is achieved.

The Coder Algorithm

Following is pseudocode for the compression algorithm, excluding the wavelet transform and the entropy coding parts. The list of detected coefficients $\mathcal{D}$ is empty in the beginning, requiring no memory at all. It grows as the coding/decoding proceeds when the wavelet coefficients get detected as being significant, with respect to one of the threshold values T_i. Let c_{ij} denote the wavelet transform coefficient at the coordinate (i, j).

1. **Initial Quantization**

   Set $T = \frac{max_{(i,j)}\{|c_{ij}|\}}{2}$.

2. **Detection Stage**

   a) For each coefficient c_{ij} at the coordinate (i, j), where $(i, j) \in$ set of all coordinates of the image, do the following in the scanning order described in Section 3.4.5:

      If $|c_{ij}| \geq T$ (i.e. c_{ij} is *significant*),
      Add $\big((i, j), c_{ij}\big)$ to the list $\mathcal{D}$.

   b) For each coefficient c_{ij} at the coordinate (i, j), where $(i, j) \in H_c$, the set of all coordinates in the three coarsest scale high frequency subbands, Do:

      If $|c_{ij}| < T$ (i.e. c_{ij} is *insignificant*),
      Detect the augmented zerotrees (if any) with root
      at (i, j) or at any of its descendants.

c) For each coefficient c_{ij} at the coordinate (i, j), where $(i, j) \in L_c$, the set of all coordinates in the lowest frequency subband, Do:

If c_{ij} is *insignificant*,

Detect the coarsest scale augmented zerotrees (if possible) with root at (i, j).

d) For each coefficient c_{ij} at the coordinate (i, j), where $(i, j) \in$ set of all coordinates of the image, do the following in the scanning order described in Section 3.4.5:

- If c_{ij}'s significance is *DNC*,

  Do nothing.

- If c_{ij} is *significant*,

  If all four of the offspring of (i, j) have been detected as ISZTR,

  Output PSZTR or NSZTR, depending upon the sign of c_{ij}.

  Otherwise,

  Output POS or NEG, depending upon the sign of c_{ij}.

- If c_{ij} is *insignificant*,

  If all four of the offspring of (i, j) have been detected as ISZTR,

  Output ISZTR.

  Otherwise,

  Output IZ.

3. **Fine-Tuning Stage**

a) For each coordinate $(i, j) \in \mathcal{D}$, do the following:

- If c_{ij} is in lower half of the corresponding dequantized interval,

  Output 0.

  Otherwise,

Output 1.

Experimental Results

The coder takes the input image file and generates an output data file for a specified target bit rate. Experimental results of both AZIC and SPIHT for three 8-bit greyscale test images namely *Lena*, *Goldhill*, and *Barbara* (all having a resolution of 512×512) are shown in Tables 3.3–3.5. The wavelet filters used are the 9-7 biorthogonal filters of [17], which have excellent frequency localization properties. A 5-level wavelet decomposition is used for both *Lena* and *Barbara*, whereas *Goldhill* is decomposed with a 6-level wavelet transform. The PSNR vs. bit rate curves for all these images are shown in Figure 3.19. From these results, it is clear that the coder's performance is comparable to the SPIHT coder.

Bitrate (bpp.)	Compression Ratio (:1)	PSNR (dB)	
		AZIC	SPIHT
1.0	8	40.95	40.87
0.8	10	39.13	39.97
0.7	11.43	38.70	39.10
0.6	13.33	38.22	38.50
0.5	16	37.74	37.83
0.4	20	36.53	37.05
0.3	26.67	35.22	35.49
0.25	32	34.67	34.75
0.2	40	33.65	33.97
0.1	80	30.74	30.85
0.08	100	29.44	29.61
0.05	160	28.06	28.02
0.025	320	25.36	25.39

Table 3.3: AZIC compression results for 512 × 512 *Lena* image

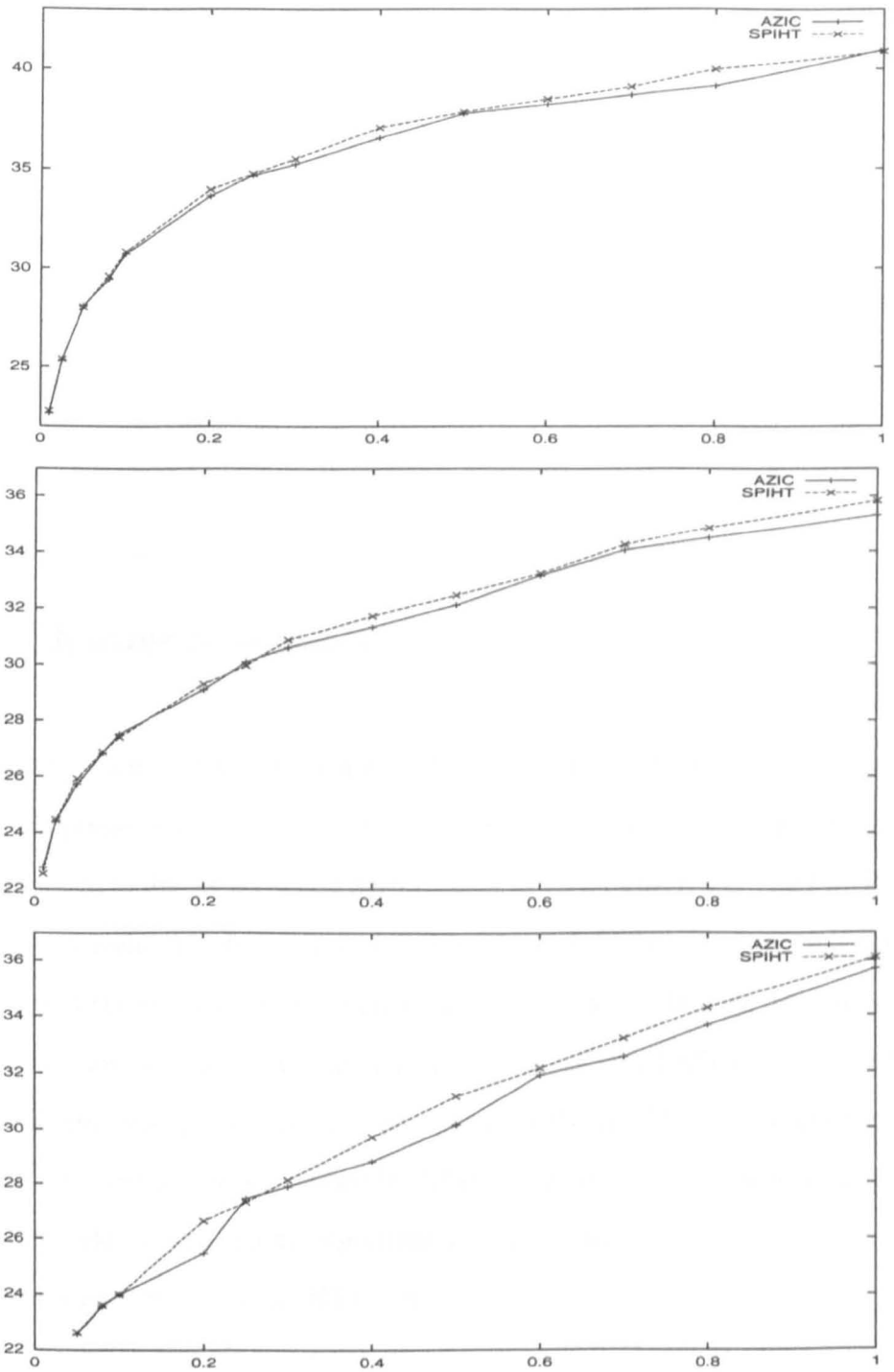


Figure 3.19: PSNR (dB) vs. bit rate (bpp.) curves for AZIC and SPIHT
Top: *Lena*, Middle: *Goldhill*, and Bottom: *Barbara*
All images have a 512×512 resolution.

Bitrate (bpp.)	Compression Ratio (:1)	PSNR (dB)	
		AZIC	SPIHT
1.0	8	35.29	35.82
0.8	10	34.47	34.82
0.7	11.43	34.05	34.24
0.6	13.33	33.14	33.21
0.5	16	32.09	32.44
0.4	20	31.31	31.70
0.3	26.67	30.57	30.84
0.25	32	30.07	29.94
0.2	40	29.08	29.28
0.1	80	27.52	27.41
0.08	100	26.85	26.88
0.05	160	25.72	25.92

Table 3.4: AZIC compression results for 512×512 *Goldhill* image

3.5 Chapter Summary

In this chapter, some of the important properties of the wavelet transform were studied from a compression viewpoint. It was argued that the decay of wavelet coefficients is directly related to the compression performance at various bit rates. The PSNR results of a simple wavelet-thresholding image coder showed that better potential gains could be obtained by employing a more sophisticated quantization strategy. The idea of zerotree quantization was explained, and an analysis of both the EZW [73] and the SPIHT [71] algorithms was presented. The zerotree hypothesis of Shapiro was extended to include the lowest frequency subbands. After its successful validation, an augmented zerotree wavelet image coding algorithm was presented whose compression performance is comparable to the SPIHT coder.

Bitrate (bpp.)	Compression Ratio (:1)	PSNR (dB)	
		AZIC	SPIHT
1.0	8	35.69	36.10
0.8	10	33.64	34.27
0.7	11.43	32.54	33.19
0.6	13.33	31.93	32.17
0.5	16	30.18	31.19
0.4	20	28.84	29.73
0.3	26.67	27.91	28.13
0.25	32	27.50	27.36
0.2	40	25.49	26.65
0.1	80	23.99	23.99

Table 3.5: AZIC compression results for 512×512 *Barbara* image

Chapter 4

Adapting the Wavelet Basis

It is noticeable from the experimental results in the previous chapter that a wavelet image coder performs much better on images consisting of smooth regions than on relatively complex textured images. A simple explanation for this fact is that the wavelet transform of an image produces a large number of very small transform coefficients corresponding to those parts of the image that are smooth in nature. The recursive dyadic decomposition of an image at different scales stops short of exploiting image patterns that consist of rapid oscillations of various frequencies. The assumption that images from natural scenes consist of smooth regions does not prove to be universally true, leading to the conclusion that the imposition of a fixed wavelet basis does a good job of compressing many, but not all images. The need thus arises to explore the wavelet bases whose time-frequency tiling is adapted to a given signal in such a way that the resulting waveforms closely resemble those present in the signal and consequently result in a sparse representation, suitable for compression purposes.

In this chapter, we investigate issues related to a generalized wavelet basis adapted to the signal or image contents, the so-called *best basis*. The question of whether a basis better than this selected 'best basis' can be found is addressed. A necessary

condition for one 'best basis' to be better than another 'best basis' is presented. The use of different cost functions may result in different *best* bases which, in turn, may produce different coding results using the same quantization method. It is, therefore, important to take into account the quantization strategy at the time of basis selection, to ensure that the basis chosen by employing a certain criterion will actually result in better performance in terms of coding gains. A new paradigm for wavelet packet basis selection is presented, which emphasizes the role of quantization strategy at the time of basis selection[1].

4.1 Wavelet Packets

Coifman, Meyer and Wickerhauser [21] introduced a generalization of the wavelet transform, which was termed as *wavelet packets*. Instead of fixing the frequency segmentation of the wavelet transform, they established that orthogonal general wavelet bases can also be found by adaptively segmenting the frequency axis into intervals of possibly different lengths, in order to better represent a given signal.

In the wavelet transform of a signal f, recall that a space V_j of its approximation at a resolution 2^{-j} is decomposed into a lower resolution space V_{j+1} and a detail space W_{j+1}. This is done by dividing the orthogonal basis $\{\phi_j(t-2^j n)\}_{n\in Z}$ of V_j into two bases $\{\phi_{j+1}(t-2^{j+1}n)\}_{n\in Z}$ of V_{j+1} and $\{\psi_{j+1}(t-2^{j+1}n)\}_{n\in Z}$ of W_{j+1} using lowpass and highpass filters $h[n]$ and $g[n]$ respectively.

In order to approximate the signal f belonging to an approximation space $V_j = W_j^0$ at a scale 2^{-j} using wavelet packets, the space W_j^0 is decomposed into two orthogonal spaces W_{j+1}^0 and W_{j+1}^1. Each of these spaces can be further decomposed into two orth-

[1] For the sake of simplicity, we discuss these issues for a 1-*d* signal first and then extend the results to images.

ogonal spaces. In general, Coifman and Meyer [18] proved that if $\{\psi_j^p(t-2^jn)\}_{n\in\mathcal{Z}}$ is an orthonormal basis of W_j^p, where $W_j^0 = V_j$, $W_j^1 = W_j$ and $\psi_j^0 = \phi_j$, $\psi_j^1 = \psi_j$, then $\{\psi_{j+1}^{2p}(t-2^{j+1}n), \psi_{j+1}^{2p+1}(t-2^{j+1}n)\}_{n\in\mathcal{Z}}$ is also an orthonormal basis of W_j^p, where

$$\psi_{j+1}^{2p}(t) = \sum_{n=-\infty}^{+\infty} h[n]\psi_j^p(t-2^jn) \tag{4.1}$$

$$\psi_{j+1}^{2p+1}(t) = \sum_{n=-\infty}^{+\infty} g[n]\psi_j^p(t-2^jn), \tag{4.2}$$

and $W_j^p = \left(W_{j+1}^{2p} \oplus W_{j+1}^{2p+1}\right)$. The support in time of a scaling and wavelet function corresponding to lowpass and highpass filters (respectively) having K non-zero coefficients is $K-1$. This implies that the time support of elongated wavelet packets ψ_j^p is simply $(K-1)2^j$.

The discrete wavelet packet transform coefficients of a 1-d signal, $\mathbf{x} = \{x_n\}_{0\le n<N}$ of length $N = 2^{-J}$ and belonging to the space V_J or W_J^0, can be computed as follows

$$\begin{aligned} w_{2n,j+1,l} &= \sum_k g_{k-2l}\, w_{n,j,k} \quad 0\le l<N2^{J-j-1} \\ w_{2n+1,j+1,l} &= \sum_k h_{k-2l}\, w_{n,j,k} \quad 0\le l<N2^{J-j-1} \\ w_{0,J,l} &= x_l \qquad\qquad\qquad\quad 0\le l<N \end{aligned}$$

where $w_{n,j,l}$ is the transform coefficient corresponding to the wavelet packet having support size $2^{j-J}(K-1)$, n and l denote the frequency and position indices, and $h[n]$, $g[n]$ are the lowpass and highpass filters respectively for a two-channel filter bank. The transform is invertible if appropriate dual filters $\tilde{h}[n]$, $\tilde{g}[n]$ exist and are used on the synthesis side of this filter bank.

Since this library of basis functions provides an overcomplete representation of the signal $\mathbf{x}$, a combination of basis vectors from this library is sought, which is well suited to the signal and also forms an orthogonal basis. Although the overall variance is preserved by the decomposition of a given image (or subband) into low and high

frequency subbands [11], it is the distribution of variance that determines if and how efficient the decomposition is from a coding viewpoint. Various criteria have been used for computing the cost of a split, such as entropy [22], actual coding cost [55], and a Lagrangian cost [67] function involving both rate and distortion. The coupling of such criteria with a fast, dynamic programming algorithm, proposed first by Coifman and Wickerhauser [22], finds the *optimal* basis adapted to the image contents with respect to that particular criterion.

To illustrate how successfully wavelet packets can localize the frequency contents of a given signal, consider the quadratic chirp signal of Figure 4.1 and its 5-level wavelet and full-tree wavelet packet contents as shown in Figure 4.2. The time-frequency tilings of its wavelet transform and a wavelet packet transform, using a new cost function presented later in Section 4.4, demonstrate that the wavelet packet transform picks up the frequency contents of this signal far better than the wavelet transform does, as is clear from Figure 4.3.

Also shown in Figures 4.5 and 4.6 are the reconstructions of the 512×512 *Barbara* image (Figure 4.4) from the largest 5% of the 5-level wavelet and wavelet packet coefficients, using Daub4 filters, respectively. The PSNR difference of more than 2dB and a clear difference in the visual quality, especially on the table cloth and lady's scarf, proves the superiority of wavelet packet transform over a wavelet transform for such images.

4.1.1 Number of Possible Bases

There are 2^d terminal nodes at each depth $d = j - L \geq 0$ (where j denotes the scale) of the wavelet packet tree in the case of a one-dimensional signal. This makes a total of $2^{D+1} - 1$ nodes in a wavelet packet tree of maximum depth D, where $D = \log_2 N$.

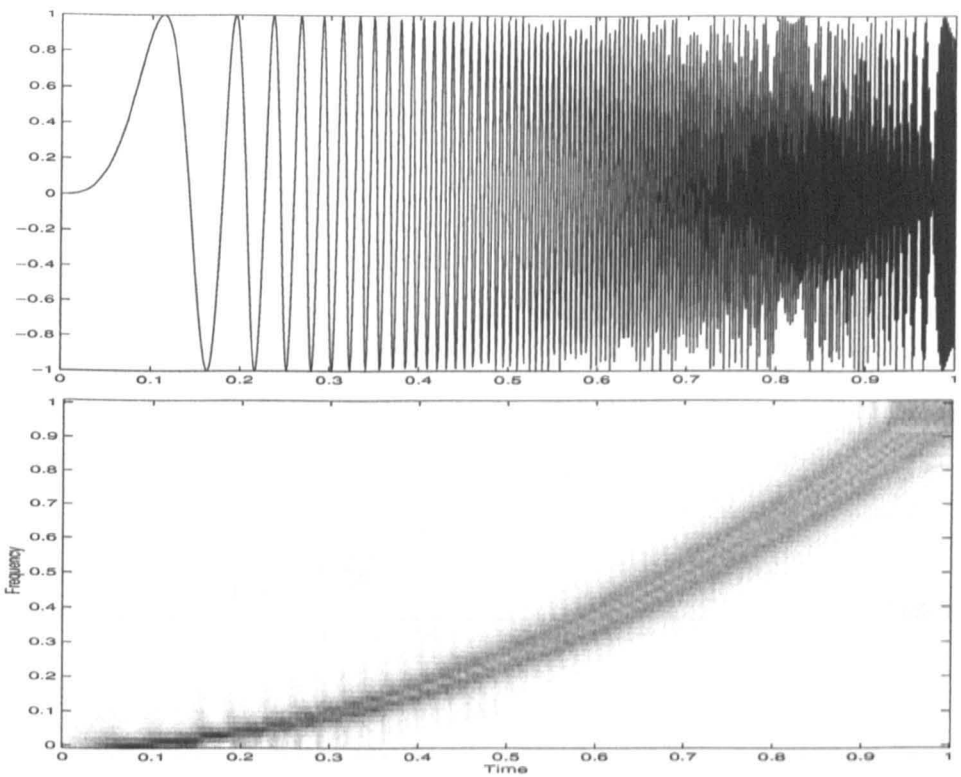


Figure 4.1: A quadratic chirp signal and its spectrogram

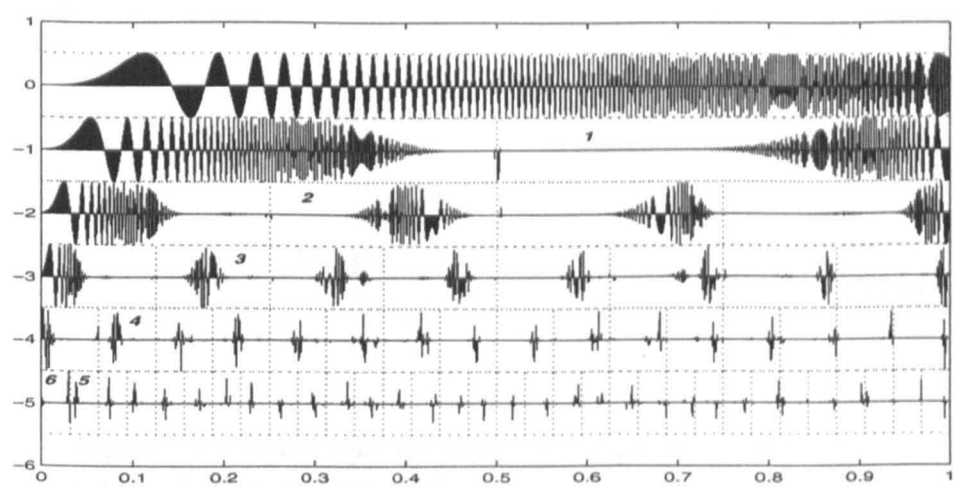
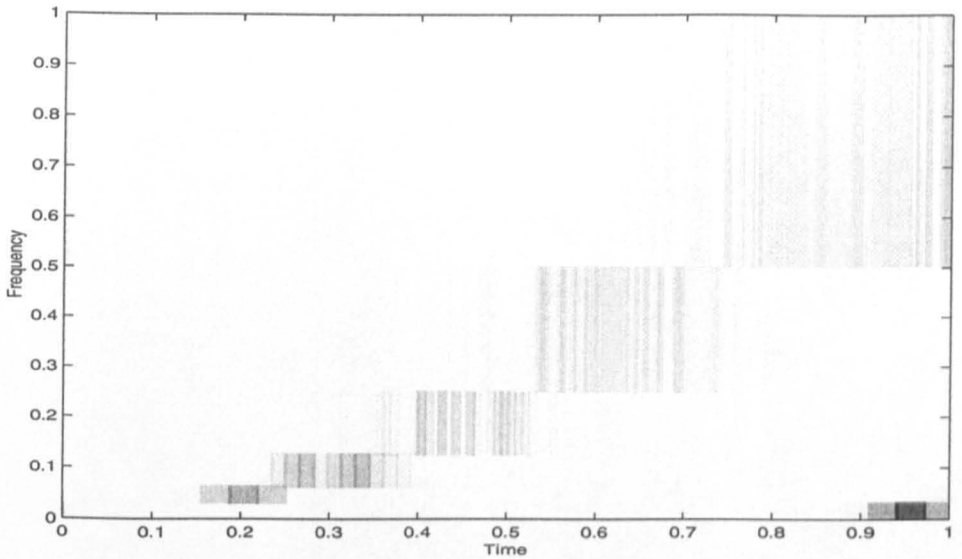
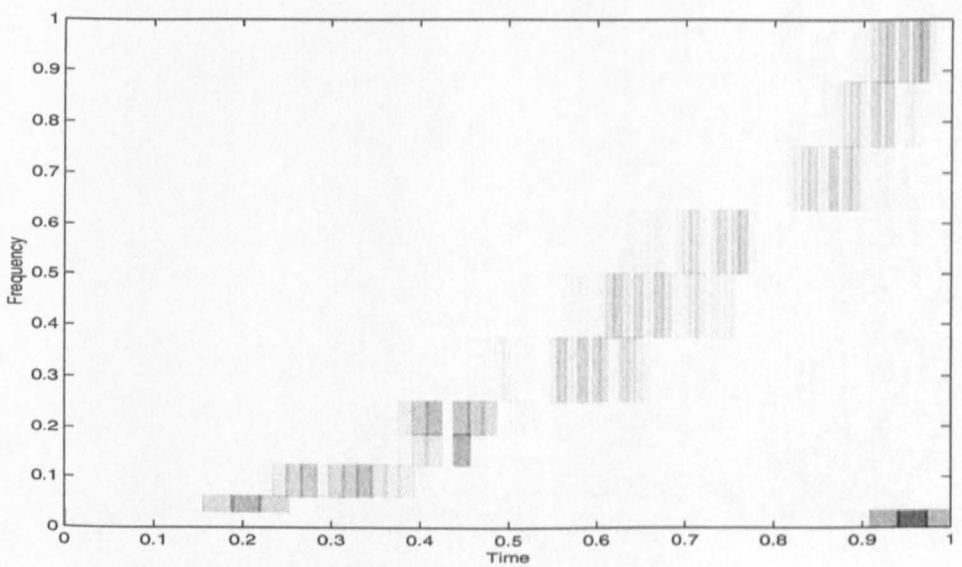


Figure 4.2: Full-tree wavelet packet subbands for signal in Figure 4.1
The numbered subbands (1...6) correspond to the wavelet transform.



(a)



(b)

Figure 4.3: Time-frequency tilings of the quadratic chirp signal using a 5-level (*a*) wavelet transform and (*b*) wavelet packet transform

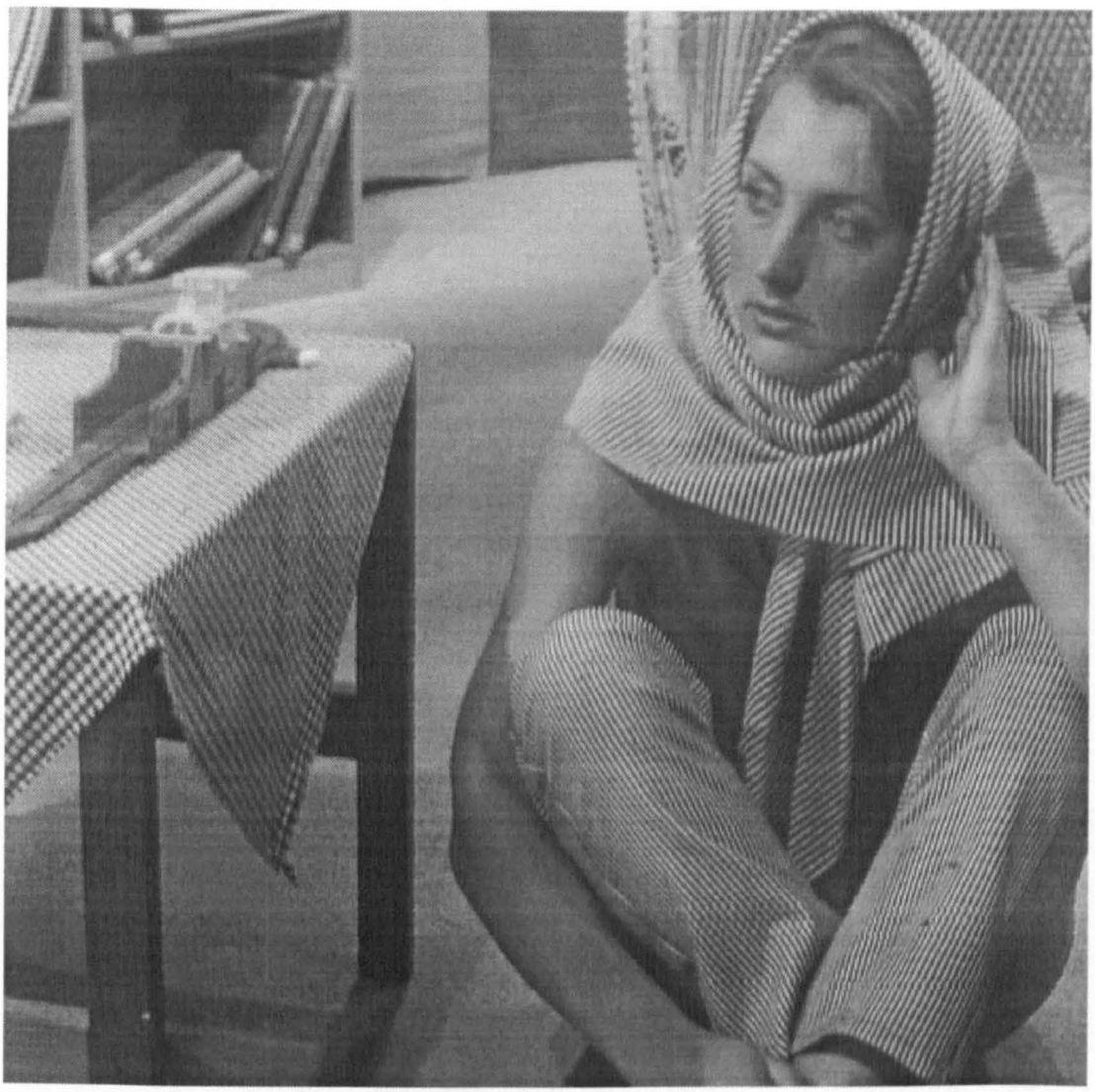


Figure 4.4: The original 512×512 *Barbara* image

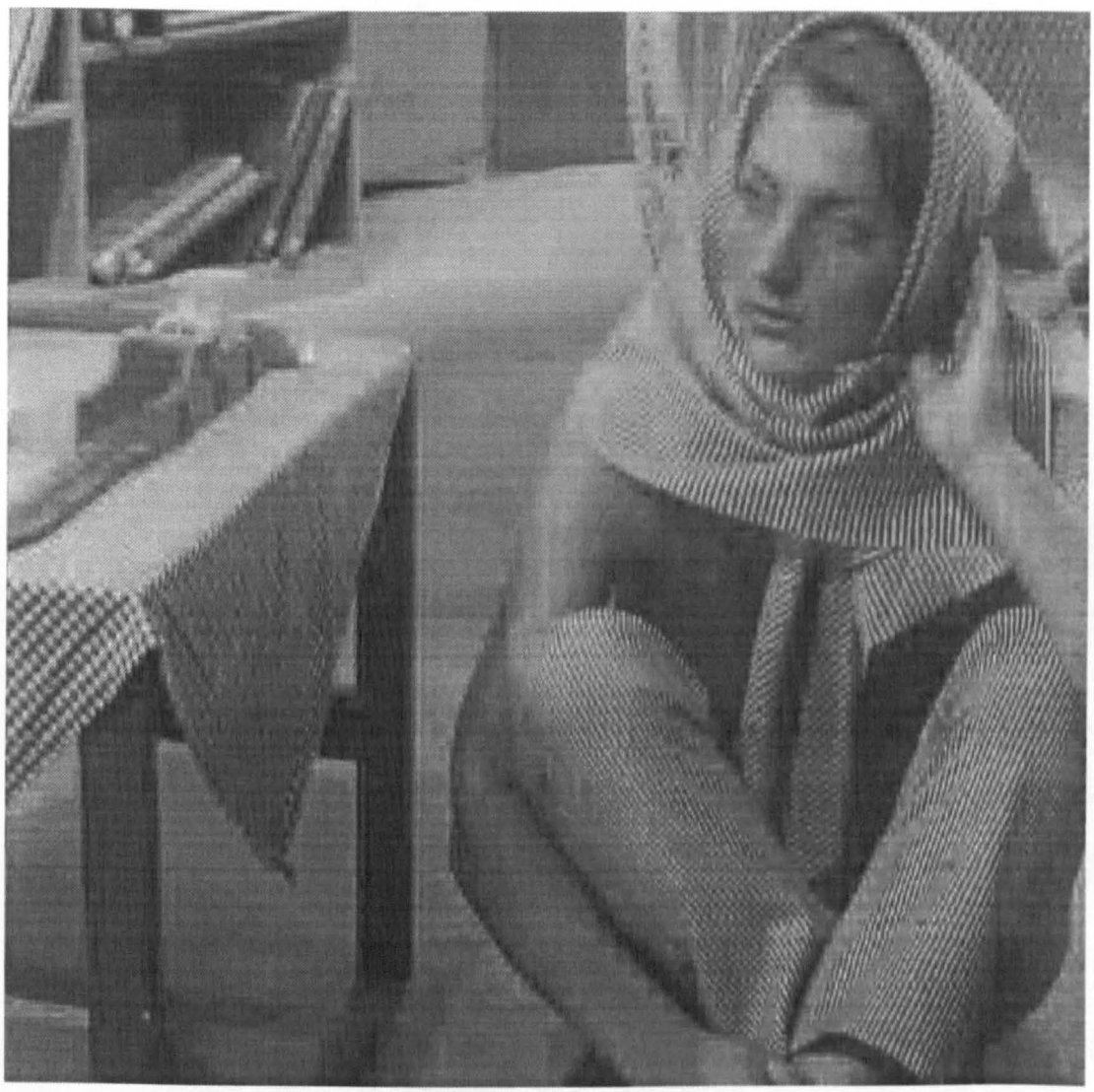


Figure 4.5: Reconstruction of *Barbara* from the largest 5% wavelet coefficients
PSNR=27.21dB

Figure 4.6: Reconstruction of *Barbara* from the largest 5% wavelet packet coefficients
PSNR=29.32dB

Keeping all the nodes of this tree is not necessary. An *admissible binary wavelet packet tree* has either zero or two children. A set of bases corresponding to all the terminal nodes of an admissible wavelet packet tree, therefore, forms a complete orthonormal basis for $V_L = W_L^0$ (used to approximate f at the scale 2^L). The binary nature of a wavelet packet tree dictates that the number of possible bases N_d at a depth d satisfies the following recursion relation

$$N_d = N_{d-1}^2 + 1$$

with $N_0 = 1$ and is bounded as follows [47]

$$2^{2^d} \leq N_{d+1} \leq 2^{\frac{5}{4}2^d}. \tag{4.3}$$

It was, however, found empirically (Table 4.1) that N_d can be bounded above more tightly by the following relation,

$$N_{d+1} \leq 2^{\frac{6}{5}2^d}.$$

This huge collection of wavelet packet bases needs to be searched to find an appropriate basis for a given signal. Issues related to searching will be discussed in Section 4.3.

d	N_d	$\log_2(\log_2 N_d)$
1	2	0
2	5	1.2153
3	26	2.2328
4	677	3.2331
5	458330	4.2331
6	2.1007×10^{11}	5.2331
7	4.4128×10^{22}	6.2331
8	1.9473×10^{45}	7.2331
9	3.7919×10^{90}	8.2331
10	1.4378×10^{181}	9.2331

Table 4.1: Number of possible wavelet packet bases N_d at depth d
$\log_2(\log_2(N_{d+1})) < (d + 0.2332)$, and $2^{0.2331} < \frac{6}{5}$.

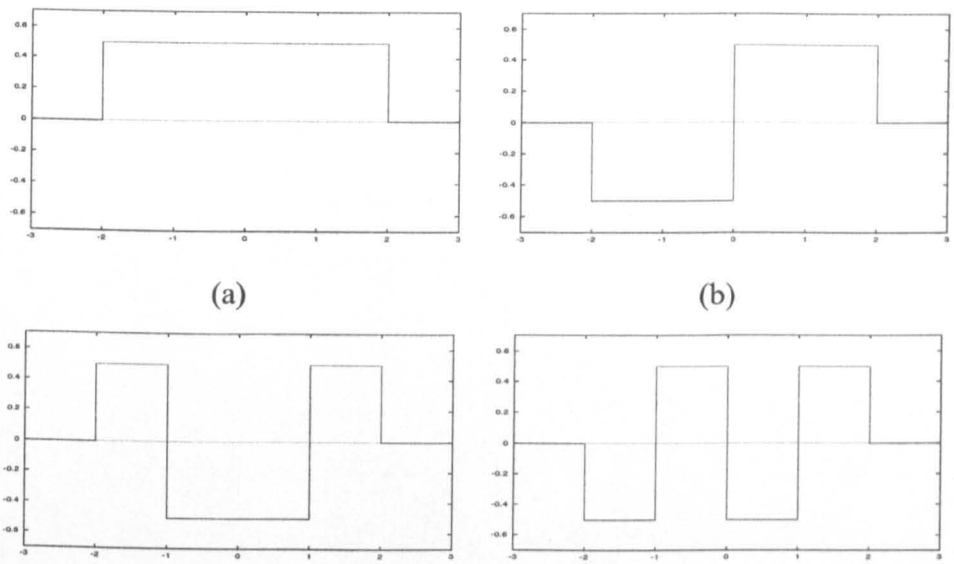


Figure 4.7: Frequency-ordered Haar-Walsh wavelet packets at depth 2 of the full wavelet packet tree
(*a*) ψ_2^0 [HH], (*b*) ψ_2^1 [HG], (*c*) ψ_2^3 [GG], and (*d*) ψ_2^2 [GH]

4.1.2 Frequency Ordering of the Wavelet Packets

It can be proved [89] that for both Shannon wavelet packets, which have a well localized frequency support, and the Haar-Walsh wavelet packets, which have a very well localized time support, the frequency support of ψ_j^p comes at the kth rank in order of increasing frequency, where k is the integer equivalent of the inverse gray code of the binary string of p. Based on the realistic assumption that the energy of $\hat{h}(\omega)$ is mostly concentrated in $[\frac{-\pi}{2}, \frac{\pi}{2}]$, this result can be extended to other compactly supported wavelet packets [47]. Some frequency-ordered Haar-Walsh wavelet packets, their Fourier transforms, and some Daub4 wavelet packets are shown in Figures 4.7–4.9.

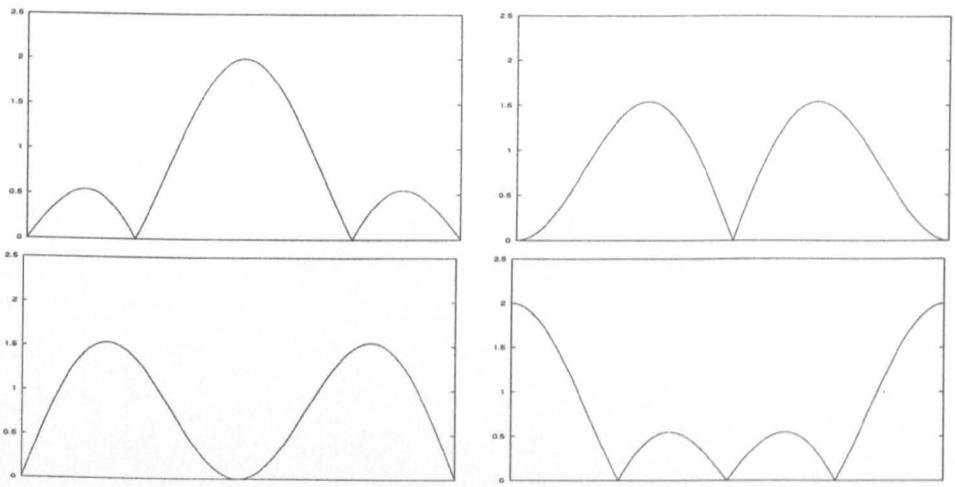


Figure 4.8: Fourier transform of Haar-Walsh wavelet packets in Figure 4.7

4.1.3 Two-Dimensional Wavelet Packets

The separable products of two one-dimensional wavelet packets $\psi_j^p(x_1 - 2^j n_1)$ and $\psi_j^q(x_2 - 2^j n_2)$ in both dimensions at same scale 2^j yield two-dimensional wavelet packets

$$\psi_j^{p,q}(x_1, x_2) = \psi_j^p(x_1)\ \psi_j^q(x_2)$$

which provide a basis for the space $W_j^{p,q}$ given by

$$W_j^{p,q} = W_j^p \otimes W_j^q.$$

Using one-dimensional wavelet packets, the space W_j^p can be decomposed into two orthogonal subspaces W_{j+1}^{2p} and W_{j+1}^{2p+1} and the space W_j^q can be decomposed into two orthogonal subspaces W_{j+1}^{2q} and W_{j+1}^{2q+1}. The joint space $W_j^{p,q}$ can, therefore, be written as

$$W_j^{p,q} = W_{j+1}^{2p,2q} \oplus W_{j+1}^{2p,2q+1} \oplus W_{j+1}^{2p+1,2q} \oplus W_{j+1}^{2p+1,2q+1}. \tag{4.4}$$

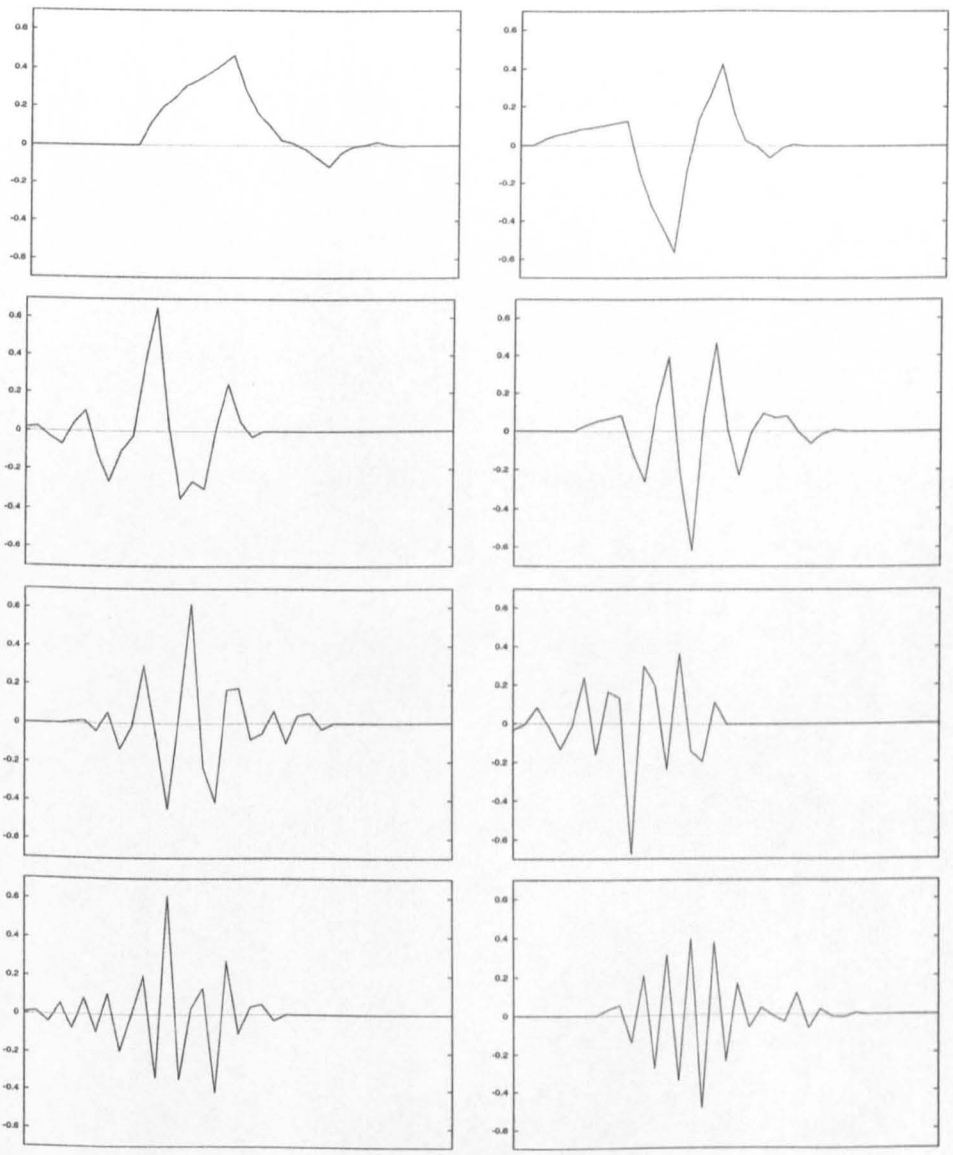


Figure 4.9: Frequency-ordered Daubechies-4 wavelet packets at depth 3 of the full wavelet packet tree

Left to *Right, Top* to *Bottom*: ψ_3^0 [HHH], ψ_3^1 [HHG], ψ_3^3 [HGG], ψ_3^2 [HGH], ψ_3^7 [GGG], ψ_3^6 [GGH], ψ_3^4 [GHH], and ψ_3^5 [GHG]

The wavelet packet tree now is a quadtree with the number of possible bases N_d now satisfying the following relation [47]

$$N_d = N_{d-1}^4 + 1.$$

An empirical method similar to that used in the one-dimensional case results in the following bounds for N_d

$$2^{4^d} \leq N_{d+1} \leq 2^{\frac{41}{40}4^d}.$$

The time support of $\psi_j^{p,q}(x_1, x_2)$ is a square of width $(K-1)2^j$, where K is the number of non-zero coefficients of the filters h and g. The energy of $\hat{\psi}_j^{p,q}(w_1, w_2)$ is mostly concentrated in the product of supports of $\hat{\psi}_j^p$ and $\hat{\psi}_j^q$. Eight two-dimensional Haar-Walsh wavelet packets associated with the terminal nodes of a two-dimensional full wavelet packet tree of depth 2 are shown in Figure 4.10.

The complexity of a two-dimensional wavelet packet transform of an $N \times N$ pixel image is $O(N^2 \log_2 N)$ as compared to $O(N^2)$ for a two-dimensional wavelet transform. The filter banks used for the wavelet transform can be used for a full wavelet packet transform, with the modification that instead of only the lowest frequency subband, all the subbands are now decomposed into further subbands.

4.1.4 Special Wavelet Packets

Two of the most commonly used special wavelet packets are M-band wavelets and full-tree wavelet packets. In an M-band wavelet packet basis, only those wavelet packets ψ_j^p are used for which $p \in [M, 2M)$ and $j > (L + \ell)$, where $\ell = \log_2 M$. This defines the wavelet transform as a 1-band wavelet packet transform, in which W_j^1 is not decomposed and ψ_j^1 are used to approximate a given function. A full-tree wavelet packet basis, also known as *pseudo local cosine basis*, of depth $D = J - L$ is obtained by

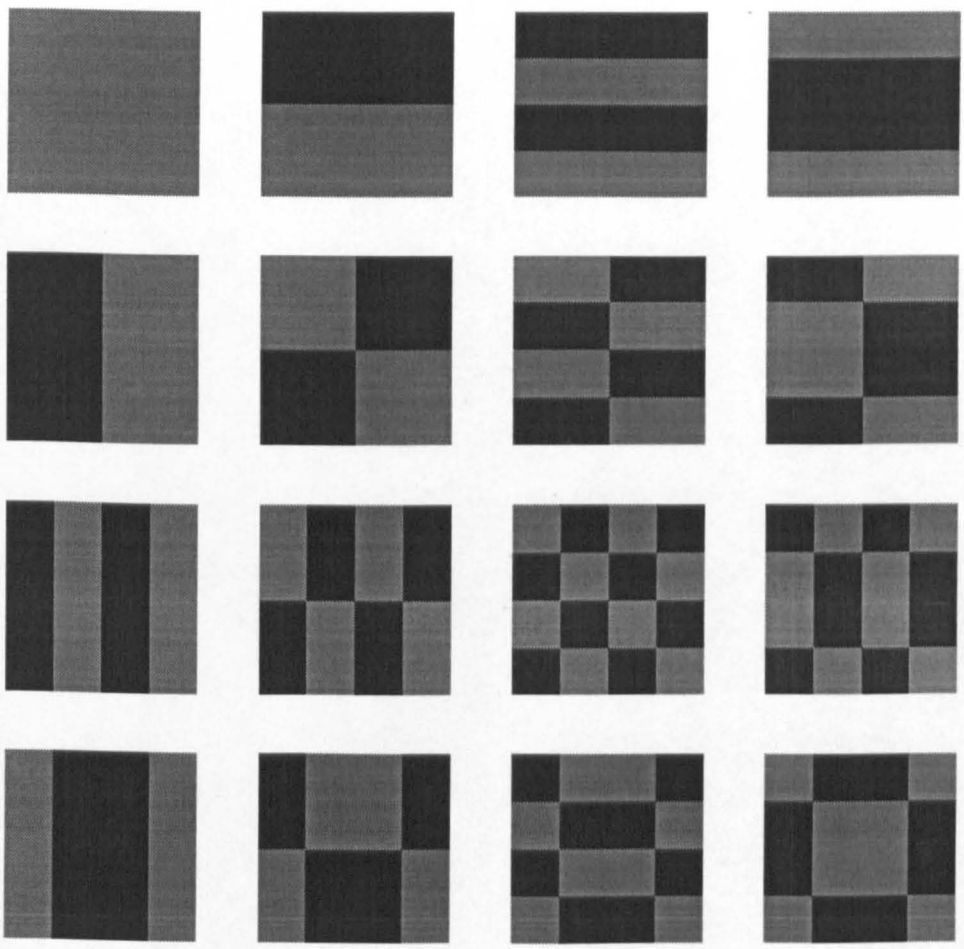


Figure 4.10: Two-dimensional Haar-Walsh wavelet packets at depth 3 of the full wavelet packet tree

Top to Bottom: *Row* 1: $\psi_2^{0,0..3}$, *Row* 2: $\psi_2^{1,0..3}$, *Row* 3: $\psi_2^{2,0..3}$, and *Row* 4: $\psi_2^{3,0..3}$.

using only the terminal nodes of a full wavelet packet tree (i.e. those ψ_j^p for which $j = J$ and $0 \leq p < 2^D$). All such ψ_J^p have a bandwidth that is equal to $\pi 2^{-J}$.

4.2 Other Adaptive Wavelet Bases

The wavelet packet transform can adaptively divide the frequency axis in order to adapt to the frequency characteristics of a given signal. It does not, however, provide a mechanism to adapt to the signal's non-stationarity, since only one unsegmented signal space is used.

This problem was addressed by Wilson et al. [96] in their development of multiresolution Fourier transform (MFT). As a method to cope with the localization limitations of short-time Fourier transform (STFT) and wavelet transform, the MFT analyzes the Fourier transform of the signal at multiple scales in order to pinpoint locally occurring signal phenomena in both time (space) and frequency. The MFT has been proven to be a general purpose tool for image analysis [13] and a very useful tool for audio and image segmentation [13, 61] in particular.

The *double tree* of Kovacevic et al. [33] provides a way to extend the frequency adaptation concept of the wavelet packet transform to include adaptation across the spatial segments of an image. A combination of spatial segmentation and wavelet packet localization in frequency (for each segment) is found in order to best suit the spatially local frequency contents of an image. A dual of this methodology would be a transform which spatially segments the frequency subbands after adaptively partitioning the frequency axis by a wavelet packet transform.

A more general scheme, named a *full adaptive tree*, accommodates all combinations of spatial and frequency splits (in no particular order) [74]. It selects the best combination

using an optimization method similar to the one discussed in Section 4.3.

Although these tools provide more generality and are better suited for some analysis purposes, we limit our study to wavelet packets due to their being computationally less expensive and therefore more suitable for compression purposes.

4.3 Selecting the Best Wavelet Packet Basis

It is clear from (4.3) that the number of possible bases N_d for a wavelet packet tree of depth d grows exponentially as d increases. A brute force approach, that tries each possible basis to find out the most suitable one, cannot be employed to represent a signal in almost all practical applications. Adaptively finding the best basis - or the best admissible wavelet packet tree - from such an enormous collection of possible bases needs a fast algorithm.

Dynamic programming offers an efficient solution to this optimization problem. According to Bellman's optimality principle (OP) [39], if $P_i : 1 \rightarrow j$ is the shortest path from 1 to j in a graph and (i, j) is the last edge of P, then $P_i : 1 \rightarrow i$ is the shortest path from 1 to i. Due to the orthonormality property of wavelet packet spaces described earlier, the signal space covered by a subtree emanating from an arbitrary node n_d^p is identical to the signal space covered by two subtrees emanating from the two immediate children of n_d^p, i.e. n_{d+1}^{2p} and n_{d+1}^{2p+1}. Based on this fact, Coifman and Wickerhauser [22] applied the OP to develop a fast algorithm which prunes the full wavelet packet tree into the best (suitable) one for a given signal. According to the OP, an optimal subtree starting from node n_d^p must include optimal subtrees starting from its child nodes n_{d+1}^{2p} and n_{d+1}^{2p+1}. Let $\mathcal{C}(f, \mathcal{B})$ denote the cost of representing f in a basis $\mathcal{B}$. The following proposition then suggests an optimal way to prune the full wavelet packet

tree to obtain the best basis for approximating (or representing) f with respect to the cost function $C(f, \mathcal{B})$.

Proposition 4.1 (Coifman & Wickerhauser)

Let $\mathcal{B}_d^p$ denote the basis of space W_j^p, $d = j - L$, corresponding to the node n_d^p, and $\mathcal{O}_j^p$ denote another basis of W_j^p associated with the best wavelet packet tree (with respect to the cost function C) emanating from the node n_d^p. Then $\mathcal{O}_j^p$ can be determined in the following recursive way:

$$\mathcal{O}_j^p = \begin{cases} \mathcal{O}_{j+1}^{2p} \cup \mathcal{O}_{j+1}^{2p+1} & if \ \left(C(f, \mathcal{O}_{j+1}^{2p}) + C(f, \mathcal{O}_{j+1}^{2p+1})\right) < C(f, \mathcal{B}_j^p) \\ \mathcal{B}_j^p & otherwise \end{cases} \tag{4.5}$$

The above proposition dictates that the cost function must be additive. In other words,

$$C\left(\mathcal{O}_{j+1}^{2p} \cup \mathcal{O}_{j+1}^{2p+1}\right) = C(f, \mathcal{O}_{j+1}^{2p}) + C(f, \mathcal{O}_{j+1}^{2p+1}). \tag{4.6}$$

The Coifman-Wickerhauser approach finds the wavelet packet basis which minimizes the overall cost C of representing f. Starting from the terminal nodes of a full wavelet packet tree of depth D, the algorithm decides whether to keep the two child nodes n_D^{2p} and n_D^{2p+1} or to keep their parent node n_{D-1}^p, for all $0 \leq p \leq 2^{D-1}$, depending on which of the following two is smaller: the sum of the costs of n_D^{2p} and n_D^{2p+1}, or the cost of n_{D-1}^p. A series of such decisions at depth $D-1$ to keep the children (*split*) or to keep the parent (*merge*) gives the optimal subtree for each node at depth $D-1$. Proceeding in a similar fashion in a breadth-first, bottom-up direction, the best basis with respect to C is obtained when we reach the root node of the tree.

The computational complexity of finding the best basis, provided the wavelet packet coefficients at all depths d, $1 \leq d \leq D$, are given, is $O(ND)$, since there are N coefficients at each scale. The overall complexity of finding the best basis for a given signal is also $O(ND)$. In cases where N is an exact power of 2, a maximum depth $D = \log_2 N$

is achievable, in which case the complexity of finding the best wavelet packet basis is $O(N \log_2 N)$.

The extension of this basis selection algorithm to images is quite straightforward. A node n_d^p can now be split into four nodes (subband images) n_{d+1}^{2p}, n_{d+1}^{2p+1}, n_{d+1}^{2p+2}, and n_{d+1}^{2p+3}. Making comparisons of the following sort

$$\mathcal{C}(f, \mathcal{O}_{j+1}^{2p}) + \mathcal{C}(f, \mathcal{O}_{j+1}^{2p+1}) + \mathcal{C}(f, \mathcal{O}_{j+1}^{2p+2}) + \mathcal{C}(f, \mathcal{O}_{j+1}^{2p+3}) < \mathcal{C}(f, \mathcal{B}_j^p)$$

to decide whether to split node n_d^p or not and proceeding in a breadth-first and bottom-up direction yields the best basis for a given image f with respect to the cost function $\mathcal{C}$. The complexity of computing the wavelet packet coefficients and finding the best basis, with respect to $\mathcal{C}$, for an image of width M and height N to depth D is $O(MND)$. For square images whose width N is (or can be made) dyadic, the complexity reduces to $O(N^2 \log_2 N)$.

4.3.1 Search Methods

The dynamic programming approach discussed above selects an optimal representation for a given signal f with respect to the cost function $\mathcal{C}$. The whole wavelet packet tree is grown first and then pruned to a smaller subtree which corresponds to a wavelet packet basis best suited to f in terms of $\mathcal{C}$.

Contrary to this bottom-up search for the optimal wavelet packet basis, Taswell [82, 83] advocated the use of top-down searches to obtain the so-called *near-best basis* in order to reduce the computational complexity of best basis selection at the loss of some performance in compression. Marpe et al. [50] employed this search method to find suboptimal wavelet packet basis constrained by an objective complexity criterion. Taswell [83] conjectured that *there appears to be no practical justification*

for using bottom-up optimizing searches instead of top-down satisficing searches for selecting bases in tree-structured wavelet transforms except for investigation and characterization of unknown signal classes. This conjecture is disproved by the following argument.

The so-called near-best basis is selected using depth-first top-down searches. The search is terminated as soon as the combined cost of child nodes is greater than the cost of their parent node. Let $C^*(n)$ denote the optimal cost of a node n (the minimum of its own cost and the sum of its children's costs) and let $C(n)$ denote the cost of a node n, equal to $C(f, \mathcal{B}^n)$, where $\mathcal{B}^n$ denotes the basis corresponding to node n. Let $\mathcal{T}^*(n)$ denote the optimal subtree emanating from a node n. According to the OP, if $C(n_d^p) > \left(C^*(n_{d+1}^{2p}) + C^*(n_{d+1}^{2p+1})\right)$, then $\mathcal{T}^*(n_d^p)$ includes both $\mathcal{T}^*(n_{d+1}^{2p})$ and $\mathcal{T}^*(n_{d+1}^{2p+1})$, and only n_d^p otherwise. In a top-down search, a comparison $C(n_d^p) > \left(C(n_{d+1}^{2p} + C(n_{d+1}^{2p+1})\right)$ is made to decide whether to split the node n_d^p into its children or not. This comparison of *flat* costs as opposed to the comparison of optimal costs (as in the CW-approach) cannot ensure finding an optimal basis due to the fact that the optimal cost $C^*(n)$ may be much smaller than its own flat cost $C(n)$ if we proceed from bottom to top. The bottom-up search method ensures, according to the OP, that there is no other basis that can represent f with a cost less than $C(r)$, where r denotes root node of the full wavelet packet tree.

For a known signal class, an optimal wavelet packet tree can be found for a specific cost function by running the bottom-up best basis selection algorithm on a representative sample of signals in that class. Once it is known that a signal belongs to a certain class, e.g. fingerprint images [10], there is no obvious need to employ a search method for selection of the best basis. Moreover, the selection of a cost function can have a drastic effect on selection of the best wavelet packet tree, as we shall see in the next section. It is the adaptability of the wavelet packet transform that gives it an edge over the fixed

wavelet transform, and the advent of powerful computing equipment encourages the use of optimal algorithms such as the bottom-up search.

4.3.2 Choice of a Cost Function

Let $\mathcal{B} = \{b_m\}_{1 \le m \le N}$ denote a basis used to represent signal f of length N, where $\{b_m\}$ $(m = 1, \ldots, N)$ denotes a set of basis vectors corresponding to N wavelet packets. Let I_M denote the set of indices of the M largest absolute wavelet packet coefficients, so that the remaining $N - M$ coefficients are set to zero in quantization. The resulting error is given by

$$\epsilon[M] = \| f \|^2 - \sum_{m \in I_M} |< f, b_m >|^2. \tag{4.7}$$

A basis $\mathcal{B}^0 = \{b_m^0\}_{1 \le m \le N}$ is said to be a *better basis* than another basis $\mathcal{B}^1 = \{b_m^1\}_{1 \le m \le N}$ if it yields smaller errors for all possible values of M. In other words,

$$\epsilon^0[M] \le \epsilon^1[M]$$

or

$$\sum_{m \in I_M^0} | < f, b_m^0 > |^2 \le \sum_{m \in I_M^1} | < f, b_m^1 > |^2 \tag{4.8}$$

for all $M = 1, \ldots, N$. For a concave function[2] Φ, the above inequality yields [47]

$$\sum_{m=1}^{N} \Phi\left(\frac{| < f, b_m^0 > |^2}{||f||^2}\right) \le \sum_{m=1}^{N} \Phi\left(\frac{| < f, b_m^1 > |^2}{||f||^2}\right)$$

or

$$C(f, \mathcal{B}^0) \le C(f, \mathcal{B}^1) \tag{4.9}$$

[2] A function f is convex (strictly convex) if f'' is non-negative (positive) everywhere. If $-f$ is convex, then f is concave.

where $\mathcal{C}(f, \mathcal{B})$ denotes a Schur concave cost function given by

$$\mathcal{C}(f, \mathcal{B}) = \sum_{m=1}^{N} \Phi\left(\frac{|< f, b_m >|^2}{||f||^2}\right). \tag{4.10}$$

When $\Phi(x) = -x\log x$ (the Shannon entropy function), the cost function $\mathcal{C}(f, \mathcal{B})$ is called the Coifman-Wickerhauser entropy (CW-entropy) [83]. This cost function was initially suggested by Coifman and Wickerhauser to take into account both rate (entropy) and distortion $\epsilon[M]$ used in compression applications. The error $\epsilon[M]$ does not, however, have a direct relation with the mean-squared-error (MSE), the most commonly used distortion measure. Another concave function $\Phi(x) = x^{p/2}$ (for $p < 2$, $x \geq 0$) corresponds to the ℓ_p-norm of wavelet packet coefficients in $\mathcal{B}$. The cost $\mathcal{C}(f, \mathcal{B})$ is given by

$$\mathcal{C}(f, \mathcal{B}) = \left(\frac{||f||_{\mathcal{B},p}}{||f||}\right)^p$$

where $||f||_{\mathcal{B},p}$ denotes the ℓ_p-norm of wavelet packet coefficients in $\mathcal{B}$.

It should be noted that the cost function $\mathcal{C}(f, \mathcal{B})$ does not necessarily need to take the form of equation (4.10). The cost function

$$\mathcal{C}(f, \mathcal{B}) = \sum_{m=1}^{N} \log |< f, b_m >| \tag{4.11}$$

is also additive and can be used to search the best basis using a bottom-up search method.

To see how the choice of a cost function affects the selection of a basis, consider Figures 4.11 and 4.12 which respectively show the geometry of wavelet packet trees selected and the compression numbers, base-2 logarithm of the accumulated energy of transform coefficients sorted by magnitude in a descending order, plotted against total coefficients minus coefficient index for the 256×256 *Barbara* image using two cost functions: CW-entropy and the $\sum \log |.|$ cost function of (4.11). The two basis

geometries are clearly very different. We shall discuss this issue in more detail in Section 4.4.

Non-additive cost functions have also been used in order to select the near-best basis [82]. Since non-additivity cannot ensure that the cost of $\left(\mathcal{B}_{d+1}^{2p} \cup \mathcal{B}_{d+1}^{2p+1}\right)$ is equal to the sum of the costs of $\mathcal{B}_{d+1}^{2p}$ and $\mathcal{B}_{d+1}^{2p+1}$, it cannot be said that the split/merge decision will result in a better basis. Another example of a non-additive cost function is the Lagrangian cost function $D+\lambda R$ where D denotes the MSE (distortion) and R denotes the first-order entropy (rate), both being non-additive [55].

When it comes to encoding the wavelet packet coefficients, the CW-entropy does not correspond to the real entropy of the wavelet packet coefficients. As a matter of fact, it can be quite the opposite: it achieves a maximum when all the coefficients have the same magnitude, whereas the Shannon entropy is zero.

4.4 How Good is the *Best* Basis?

We know from the previous section that a basis $\mathcal{B}^0$ is better than another basis $\mathcal{B}^1$ with respect to a cost function $\mathcal{C}$ if its cost $\mathcal{C}(f, \mathcal{B}^0)$ is always smaller than the cost $\mathcal{C}(f, \mathcal{B}^1)$ of representing f in $\mathcal{B}^1$. The best basis $\mathcal{B}^*$ from a collection $\mathcal{D}$ of different bases with respect to $\mathcal{C}$ is the one which satisfies the following relation,

$$\mathcal{C}(f, \mathcal{B}^*) \leq \mathcal{C}(f, \mathcal{B}) \tag{4.12}$$

$\forall \mathcal{B} \in \mathcal{D},\ \mathcal{B} \neq \mathcal{B}^*$. Selection of the best basis $\mathcal{B}^*$ from $\mathcal{D}$ ensures that there is no other basis in $\mathcal{D}$ which can represent f with a smaller cost, when using the cost function $\mathcal{C}$. It does not, however, ensure that there is no other basis from another library of bases that can represent f with a smaller cost. Neither does it guarantee that there is no other cost function $\mathcal{C}'$ which can be used to select a different basis $\mathcal{B}'$ which is better than all

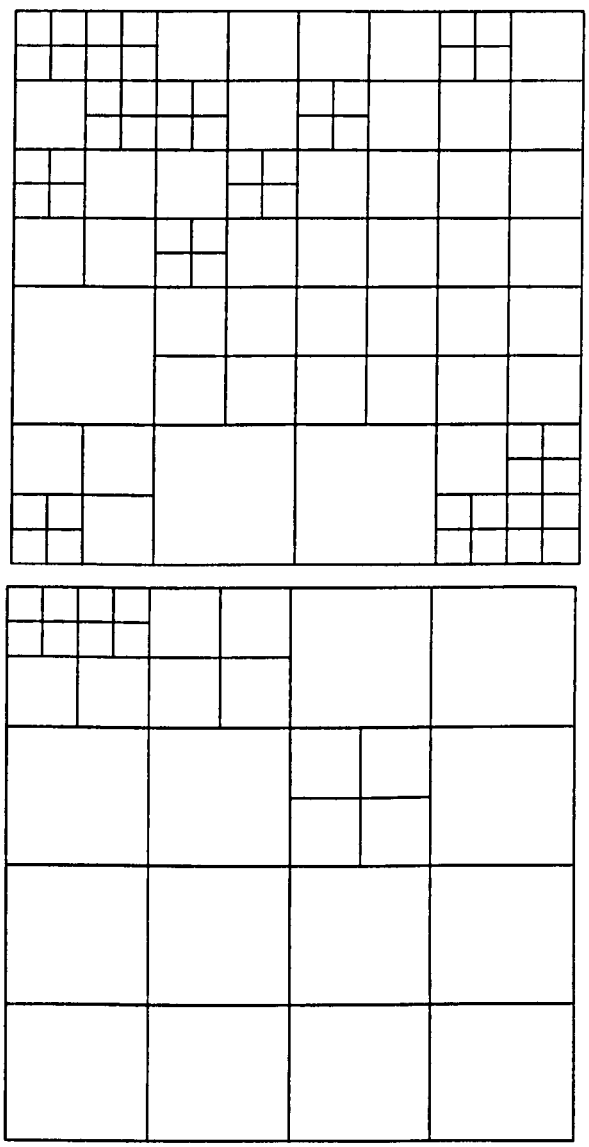


Figure 4.11: Geometries of the best bases selected for 256×256 *Barbara* image using different cost functions

Top: CW-entropy, *Bottom*: $\sum \log |.|$.

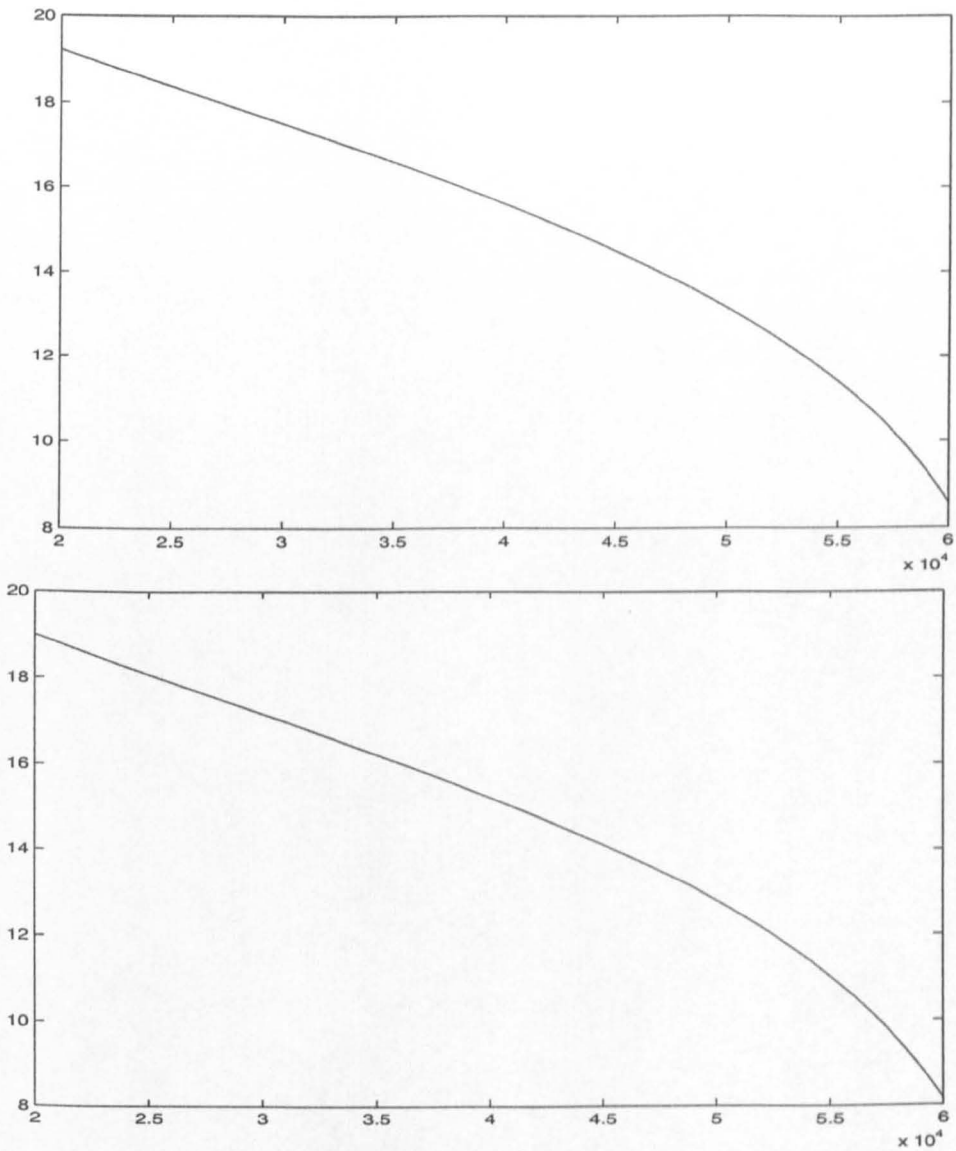


Figure 4.12: Compression numbers of the best bases selected for 256×256 *Barbara* image using different cost functions

Top: CW-entropy, *Bottom*: $\sum \log |.|$.

other bases (including $\mathcal{B}^*$) in $\mathcal{D}$, with respect to $\mathcal{C}'$. Indeed, as will be established later in this section, such a function can be found if a necessary condition is satisfied.

An *ideal basis* $\mathcal{B}^\bullet$ would be one which can represent f with just a single coefficient. In other words, one of its vectors $b_m^\bullet$ is proportional to f. Mathematically

$$b_m^\bullet = A f \tag{4.13}$$

where A is a complex-valued number. This means that f can be recovered exactly using only one basis vector $b_m^\bullet$. From (4.10), the cost of representing f in $\mathcal{B}^\bullet$ is given by

$$\mathcal{C}(f, \mathcal{B}^\bullet) = \Phi(1). \tag{4.14}$$

A *diffusing basis* $\mathcal{B}^\circ$, on the other hand, is the worst basis for representing f as it uniformly diffuses the energy of f across all the basis vectors such that

$$| < f, b_m^\circ > |^2 = \frac{||f||^2}{N}$$

for $m \in [1, N]$. The cost function $\mathcal{C}$ for this basis is given by

$$\mathcal{C}(f, \mathcal{B}^\circ) = N\Phi\left(\frac{1}{N}\right). \tag{4.15}$$

Proposition 4.2 (Mallat [47]) *Any basis $\mathcal{B}$ is worse than the ideal basis $\mathcal{B}^\bullet$ and better than the diffusing basis $\mathcal{B}^\circ$. If $\Phi(0) = 0$, then*

$$\Phi(1) \leq \mathcal{C}(f, \mathcal{B}) \leq N\Phi\left(\frac{1}{N}\right). \tag{4.16}$$

The above inequality is an important relation as it gives lower and upper bounds for representing f in any basis $\mathcal{B}$ using the cost function $\mathcal{C}$. For instance, if $\Phi(x)$ is $-x\log x$, then (4.16) tells us that $\mathcal{C}(f, \mathcal{B})$ will always lie in the interval $[0, \log N]$.

Continuing our discussion about finding a basis better than the best basis $\mathcal{B}_1$, which was found using $\mathcal{C}_1$, let us assume that there exists another cost function $\mathcal{C}_2$ which enables us to find the best basis $\mathcal{B}_2$ from the same library of bases $\mathcal{D}$. From (4.16), we have

$$\Phi_1(1) \leq \mathcal{C}_1(f, \mathcal{B}_1) \leq N\Phi_1\left(\frac{1}{N}\right) \tag{4.17}$$

and

$$\Phi_2(1) \leq \mathcal{C}_2(f, \mathcal{B}_2) \leq N\Phi_2\left(\frac{1}{N}\right) \tag{4.18}$$

where Φ_1 and Φ_2 are concave functions used for cost functions $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively, in the spirit of (4.10).

Suppose now that the cost of representing f in $\mathcal{B}_2$ with respect to $\mathcal{C}_2$ is smaller than the cost of representing f in $\mathcal{B}_1$ with respect to $\mathcal{C}_1$,

$$\mathcal{C}_2(f, \mathcal{B}_2) \leq \mathcal{C}_1(f, \mathcal{B}_1). \tag{4.19}$$

Combining (4.19) with (4.17) and (4.18), we have

$$\frac{1}{N}\Phi_2(1) \leq \Phi_1\left(\frac{1}{N}\right). \tag{4.20}$$

This implies that in order for $\mathcal{C}_2$ to select a best basis that is better than the best basis selected with $\mathcal{C}_1$, (4.20) is a necessary condition. On the other hand, if

$$\Phi_1(1) > N\Phi_2\left(\frac{1}{N}\right), \tag{4.21}$$

then we can say with certainty that $\mathcal{C}_1(f, \mathcal{B}_1)$ is always larger than $\mathcal{C}_2(f, \mathcal{B}_2)$. The above inequality tells us that using $\mathcal{C}_1$, the cost of an ideal basis is more than the cost of a diffusing basis using $\mathcal{C}_2$.

Inequalities (4.20) and (4.21) are necessary and sufficient conditions respectively for the cost of representing f in $\mathcal{B}_2$ using $\mathcal{C}_2$ being smaller than that of representing f in

$\mathcal{B}_1$ using $\mathcal{C}_1$. These can also help us decide which Φ to choose from a list of concave functions. For instance, it can be shown that $\Phi(x) = \log x$ is better than all of $-x\log x$, $x^{p/2}$, and $\sqrt{x}$ (refer to Appendix B for a proof). This can also be observed when looking at the two spectrograms in Figure 4.13, which show time-frequency tilings of the wavelet packet bases selected using two concave functions: $-x\log x$ and $\log x$.

An ideal basis is the best possible representation for a given signal from a compression standpoint. As a new best basis is found which is better than the previous best basis, it can be termed as a step towards the ideal basis. When there exists no better basis than the best basis in a library of bases irrespective of the cost function, a limit is reached for that library. Other libraries need to be investigated, to get closer to the ideal basis.

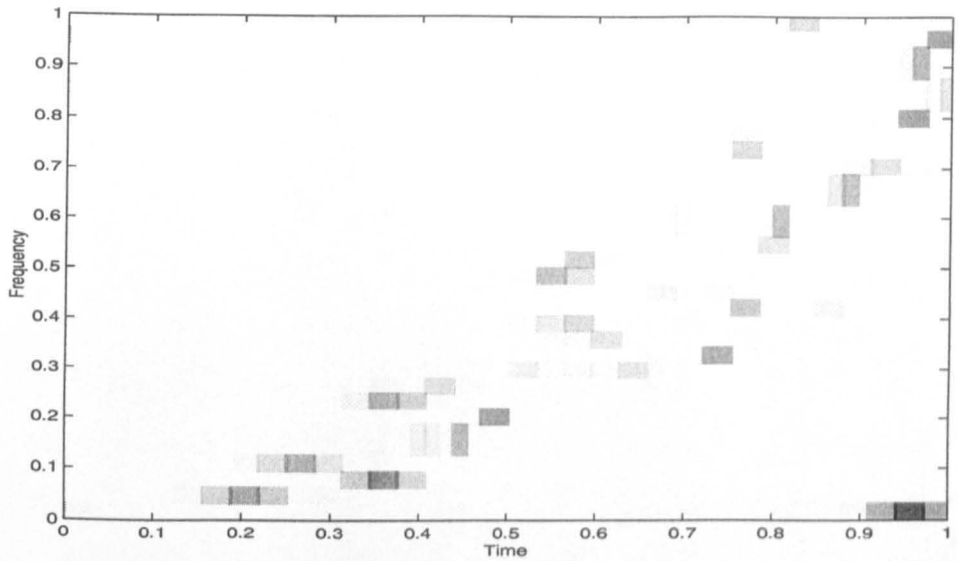
4.5 A New Paradigm for Basis Selection

An important issue that needs to be taken into account while selecting the best basis or when comparing two selected bases is that of the strategy used to encode the basis coefficients. Consider, for instance, the most commonly used encoding strategy which employs some sort of quantization $\mathcal{Q}$ (to suppress unimportant coefficients or some parts of them) followed by an entropy coding method. The encoding error $(\epsilon[M])_{\mathcal{Q}}$ is now given by

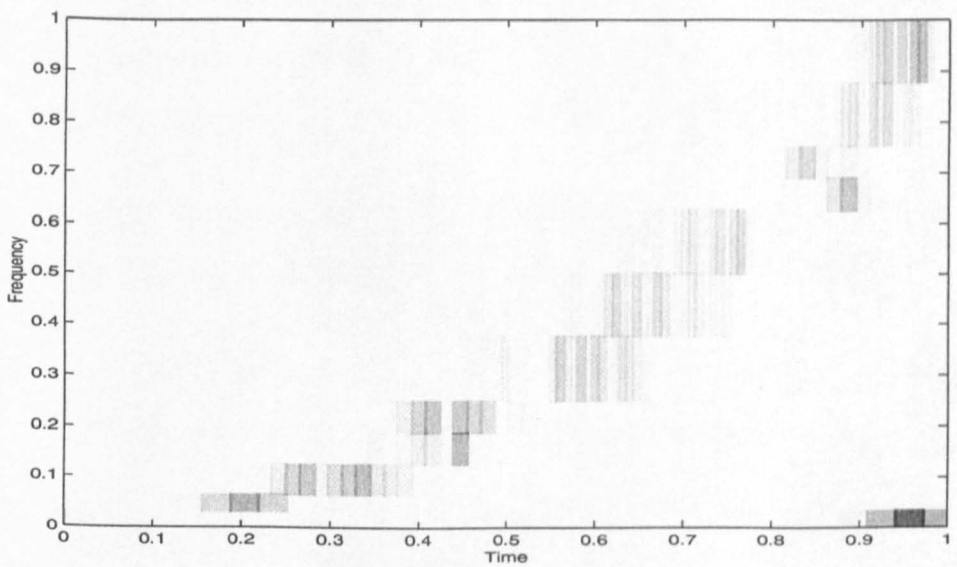
$$(\epsilon[M])_{\mathcal{Q}} = ||f||^2 - \sum_{m \in I_M} |\mathcal{Q}(< f, b_m >)|^2$$

for $M = 1, \ldots, N$. Suppose two different quantizers $\mathcal{Q}^\alpha$ and $\mathcal{Q}^\gamma$, followed by an entropy coder, are used to encode the coefficients of same wavelet packet basis $\mathcal{B} = \{b_m\}_{1 \leq m \leq N}$. Let $(\epsilon[M])_\alpha$ and $(\epsilon[M])_\gamma$ denote the encoding errors due to quantizers $\mathcal{Q}^\alpha$ and $\mathcal{Q}^\gamma$ respectively for encoding the coefficients of $\mathcal{B}$. In general, there exists $M \in [1, N]$ such that

$$(\epsilon[M])_\alpha \neq (\epsilon[M])_\gamma.$$



(a)



(b)

Figure 4.13: Spectrograms of quadratic chirp signal using best wavelet packet basis selected by different cost functions

(*a*): CW-entropy and (*b*): $\Phi(x) = \log x$

Now consider two different bases $\mathcal{B}^0$ and $\mathcal{B}^1$ used to represent f. Suppose the quantizers $\mathcal{Q}^\alpha$ and $\mathcal{Q}^\gamma$ are used to encode the coefficients of $\mathcal{B}^0$ and $\mathcal{B}^1$ (respectively). If there exist $M_1, M_2 \in [1, N]$ such that

$$(\epsilon^0[M_1])_\alpha < (\epsilon^1[M_1])_\gamma$$

and

$$(\epsilon^0[M_2])_\alpha > (\epsilon^1[M_2])_\gamma$$

then neither $\mathcal{B}^0$ nor $\mathcal{B}^1$ is a better basis than the other, according to (4.8). In other words, if it happens that the basis coefficients of $\mathcal{B}^0$ when encoded by $\mathcal{Q}^\alpha$ cannot always produce a lower error than the coefficients of $\mathcal{B}^1$ encoded by $\mathcal{Q}^\gamma$, neither of $\mathcal{B}^0$ and $\mathcal{B}^1$ is a better basis than the other. Consider the geometries of two bases for 256×256 *Barbara* image in Figure 4.11. While it is clear that the second basis (selected using $\sum \log |.|$ as the cost function) can yield better performance than the first basis when using a zerotree quantization strategy, it cannot guarantee better performance for a simple thresholding and bit-plane encoding method such as that of [55].

To emphasize the crucial role of the quantization strategy at the time of basis selection, let us redefine the best basis as follows.

Definition: A basis $\mathcal{B}^i$ belonging to a collection of bases $\mathcal{D} = \{\mathcal{B}^k\}_{k=1,2,\ldots}$ is the best basis with respect to a cost function $\mathcal{C}$ and a quantization strategy $\mathcal{Q}$ if

$$\mathcal{C}(f, \mathcal{B}^i) \leq \mathcal{C}(f, \mathcal{B}^j) \tag{4.22}$$

and

$$(\epsilon^i[M])_\mathcal{Q} \leq (\epsilon^j[M])_\mathcal{Q} \tag{4.23}$$

for all $j \neq i$ and $M \in [1, N]$.

4.6 Chapter Summary

In this chapter, the importance of adapting the underlying wavelet basis according to contents of the given image was emphasized. The use of a wavelet packet transform whose time-frequency tiling is adapted to the given image was advocated. Issues related to the selection of the *best* basis among the library of available wavelet packet bases were discussed. It was shown that the choice of a cost function can have a considerable effect on selection of the *best* basis, in turn, producing different coding results using the same quantization method. A new paradigm was presented for wavelet packet basis selection which aims to unite the basis selection process with quantization strategy to achieve better compression performance.

Chapter 5

Progressive Wavelet Packet Image Coding

The discussion so far has established that the wavelet transform is a useful tool for encoding smooth images with well-defined boundaries, whereas the wavelet packet transform is a better option for relatively complex images. The symmetry that exists among wavelet subbands provides more efficient image compression in terms of coding gains. Practical coding schemes such as [11, 71, 73, 101] exploit these symmetry properties and have successfully exhibited their superiority over block transform coding methods. The artifacts caused by wavelet coding are known to be psychovisually tuned and thus are not so easily detectable even at medium bit rates [47]. These methods, however, have a problem at low bit rates: while they perform well on images containing smooth regions and edges, they perform poorly on images with oscillatory patterns, such as the *Barbara* image. The quantization of many low energy coefficients belonging to the high frequency subbands causes artificial smooth regions (*smearing*) in the areas of image that contain rapid variations of intensity. Recent work [56] on multi-layered representation of images for coding purposes uses the wavelet transform for encoding only the edges and the smooth contents of the image. Results from various image coding schemes based on wavelet packets [55, 103] show that such methods

are particularly good in encoding images with oscillatory patterns – one form of *texture.*

Zerotree quantization, as discussed earlier in Chapter 3, is an effective way of exploiting the self-similarities among high frequency subbands at various resolutions. The main thrust of this quantization strategy is in the prediction of the significance of corresponding wavelet coefficients in higher frequency subbands at the finer scales, by exploiting parent-offspring dependences. This prediction works well, in terms of efficiently encoding the wavelet coefficients, due to the statistical characteristics of subbands at various resolutions and is related to the scale-invariance of edges in high frequency subbands of similar orientation.

While wavelet packet bases can be adapted to a given image, one apparently loses the parent-offspring dependences, the *zerotrees* or the *spatial orientation trees.* In this chapter, the following questions are addressed:

(*a*) Can the zerotree quantization strategy be applied to the wavelet packet transformed images?

(*b*) If so, how would the spatial orientation trees, or zerotrees, be defined in order to predict the insignificance of corresponding wavelet packets at a finer scale, given a wavelet packet coefficient at a coarser scale?

The validity of the first of these questions arises from the fact that there is less self-similarity among the wavelet packet subbands than among the wavelet subbands. We first provide an answer to the latter question, assuming that zerotree quantization can be applied to the wavelet packet decomposition of an image. A generalized zerotree structure, termed the *compatible zerotree* structure, is presented, which provides a quantization framework for encoding the wavelet packet coefficients. The answer to

question (*a*) becomes clear from the coding results and the successful testing of this structure.

This chapter is organized as follows. In the next section, an extension of wavelet zerotrees is presented for the wavelet packet transform. The compatible zerotree structure and related algorithms are described in Section 5.2. An analysis of general zerotree quantization based on Markov chains is presented in Section 5.3. The new basis selection paradigm, presented in the previous chapter, is unified with the zerotree quantization strategy to develop an image coder which is explained in Section 5.4. The chapter concludes with a presentation and discussion of experimental results of applying this coder to a diverse collection of test images.

5.1 Wavelet Packets Meet Zerotrees

Wavelet based zerotree image coding methods [73, 71, 101] have proved their superiority over others in terms of both computational complexity and compression performance. Moreover, embedded (progressive) transmission and reconstruction, which is required in some applications, is quite straightforward[1] using a zerotree method. The possibility of wedding the wavelet packet transform, which does not have as simple a dyadic decomposition as the wavelet transform, with zerotree quantization is, therefore, worth exploring.

Attempts have been made to combine wavelet packets with zerotrees to efficiently encode complex images [103, 14]. The rearrangement algorithm of Cho and Ra [14] puts together the coefficients of four split subbands to merge them back into a single subband, as if no split was made. This rearrangement is iterated until a wavelet-like sub-

[1] due to the facts that the significance map sends information about newly found significant coefficients and their values can be further refined subject to the bit budget

band decomposition is obtained. The SPIHT coder is then applied to this wavelet-like decomposition. However, this algorithm faces a consistency problem that in a post-rearrangement scenario, four child coefficients of a coefficient in a lower frequency subband do not necessarily belong to the same spatial location.

This issue of wavelet packet zerotrees was also addressed partially by Xiong et al. [103]. In their work, however, the subband decomposition was restricted not to have a basis causing *parenting conflict*, a problem explained in Section 5.2.1. This may result in the selection of a suboptimal basis.

Another approach towards solving this problem is as follows. Suppose the best basis has been selected, using any of the methods described in the previous chapter, and zerotree quantization is to be used to encode the wavelet packet transform coefficients. The real issue is how to organize spatial orientation trees so as to exploit the self-similarities, if any, among the subbands. The wavelet packet subbands do not, apparently, yield parent-offspring relationships such as those in the wavelet subbands. Let us try to extend the concept of zerotrees from the fixed wavelet scenario to this general wavelet packet case. Consider a square image of width $N = 2^{-L}$ belonging to the space V_{L^2} or $W_L^{0,0}$. A 2-level wavelet decomposition of this image can also be shown by a quadtree of depth 2 (Figure 5.1) each of whose nodes' label is related to the subspace associated with its corresponding subband. The node $n_L^{0,1}$, for instance, corresponds to the subspace $W_L^{0,1}$ and the subband LH_1, and so on. Each coefficient of $n_{L+2}^{0,1}$ (LH_2) is associated with four coefficients belonging to $n_{L+1}^{0,1}$ at a similar spatial location. In other words, it can be said that $n_{L+2}^{0,1}$ is the parent node of $n_{L+1}^{0,1}$ in the sense of zerotrees. A frequency split of all subbands in this decomposition, shown in Figure 5.2, supports this conclusion too.

Consider the 2-level wavelet packet decomposition and its subspace tree shown in

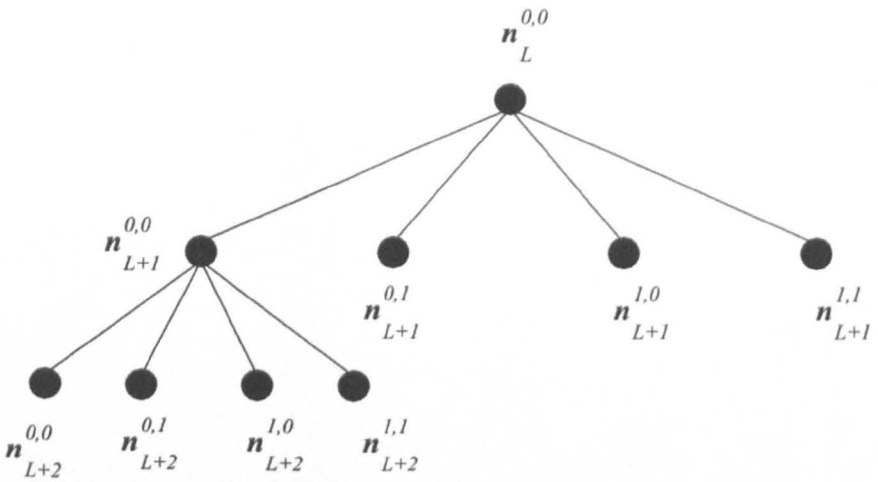


Figure 5.1: Subspace tree for a 2-level wavelet decomposition

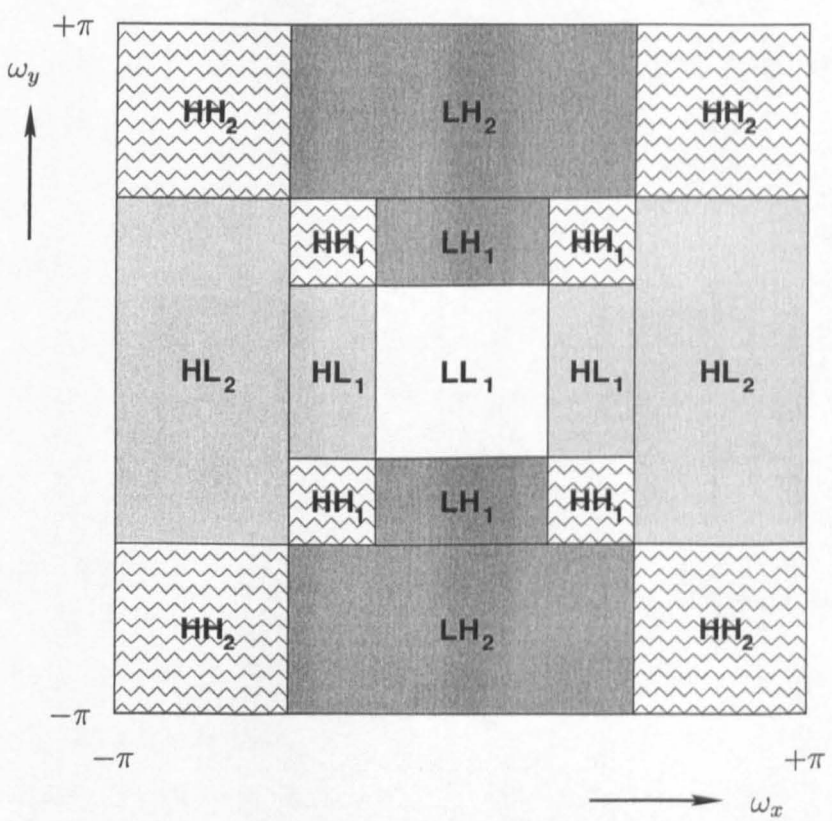


Figure 5.2: Split of the two-dimensional spatial frequency plane due to a 2-level wavelet decomposition

Figure 5.3. Each coefficient of $n_{L+2}^{1,0}$ can be spatially associated with four coefficients in $n_{L+2}^{2,0}$, $n_{L+2}^{2,1}$, $n_{L+2}^{3,0}$, and $n_{L+2}^{3,1}$. In other words, the node $n_{L+2}^{1,0}$, which was the parent node of $n_{L+1}^{1,0}$ in a wavelet tree, is the parent of all of the four children of $n_{L+1}^{1,0}$ (the node with similar orientation, or same superscripts, at previous level).

Consider now another 3-level wavelet packet decomposition and the corresponding subspace tree shown in Figure 5.4. This is just like a wavelet decomposition, except that the nodes $n_{L+2}^{1,1}$ and $n_{L+1}^{1,1}$ have been further split. The orientations of all children of $n_{L+2}^{1,1}$ are the same as those of all of the children of $n_{L+1}^{1,1}$. Therefore, each coefficient of $n_{L+3}^{2,3}$, for instance, can be associated with four coefficients of $n_{L+2}^{2,3}$ at a similar spatial location, due to their compatibility of orientation.

These simple lessons from wavelet zerotree organization are utilized and a set of rules to construct the zerotree structure for an arbitrary wavelet packet geometry is presented in the next section.

5.2 Compatible Zerotree Quantization

Due to the dyadic nature of wavelet subbands, the organization of parent-child relationships is quite straightforward. As shown in Figure 5.5, these dependences are generated in such a manner that the whole subband decomposition can be divided into three different types of subbands - horizontal, $\mathcal{H}$, vertical $\mathcal{V}$, and diagonal $\mathcal{D}$.

The subbands in a wavelet packet decomposition can also be classified into three main categories, similar to three families of subbands in a wavelet decomposition. We use the term *compatible zerotrees* to denote the trees of subbands having similar global orientation, i.e. belonging to the same family. The adjective *compatible* originates from the fact that these trees are generated taking into account both scale and orienta-

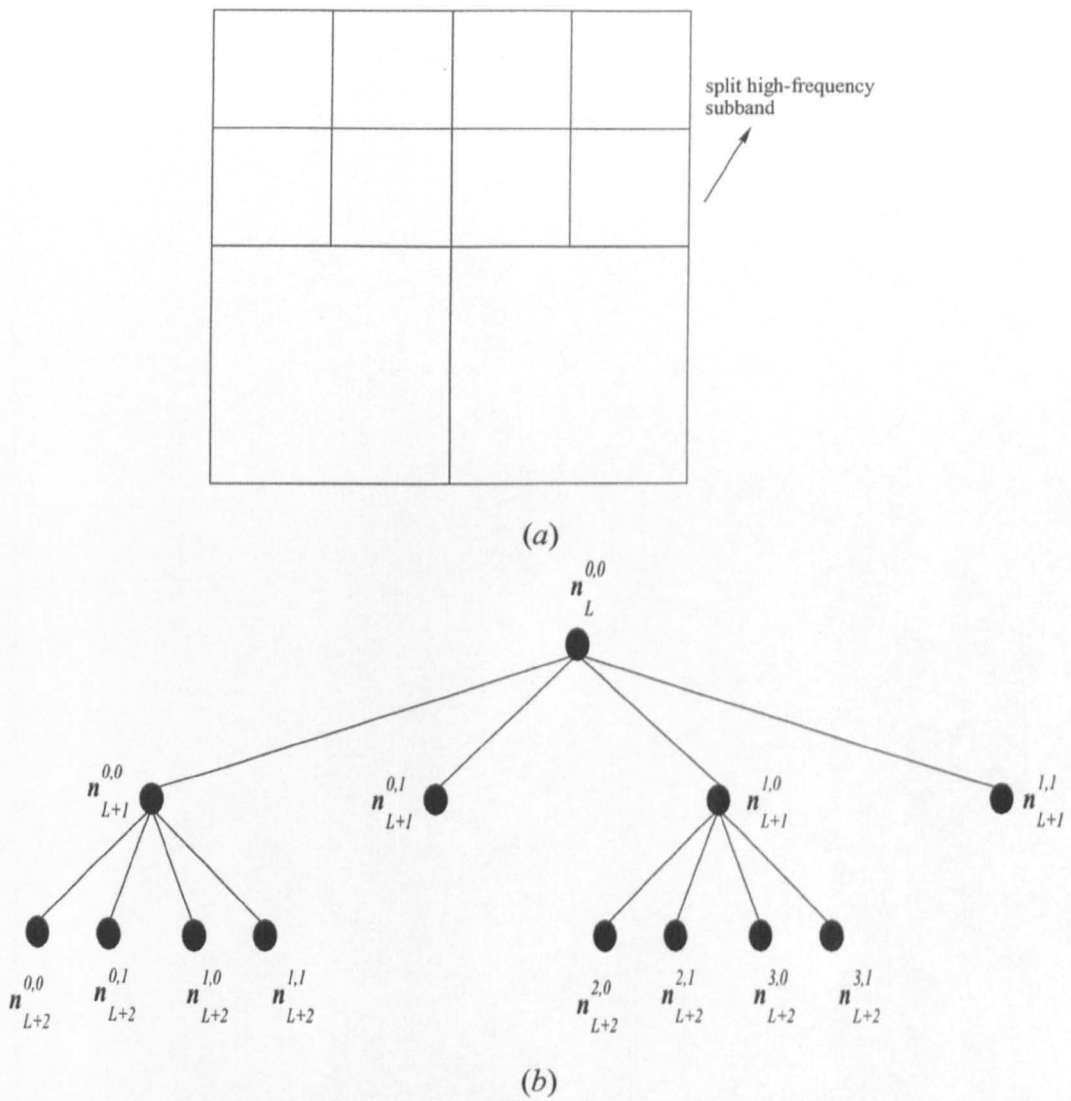


Figure 5.3: Subspace tree for a simple 2-level wavelet packet decomposition

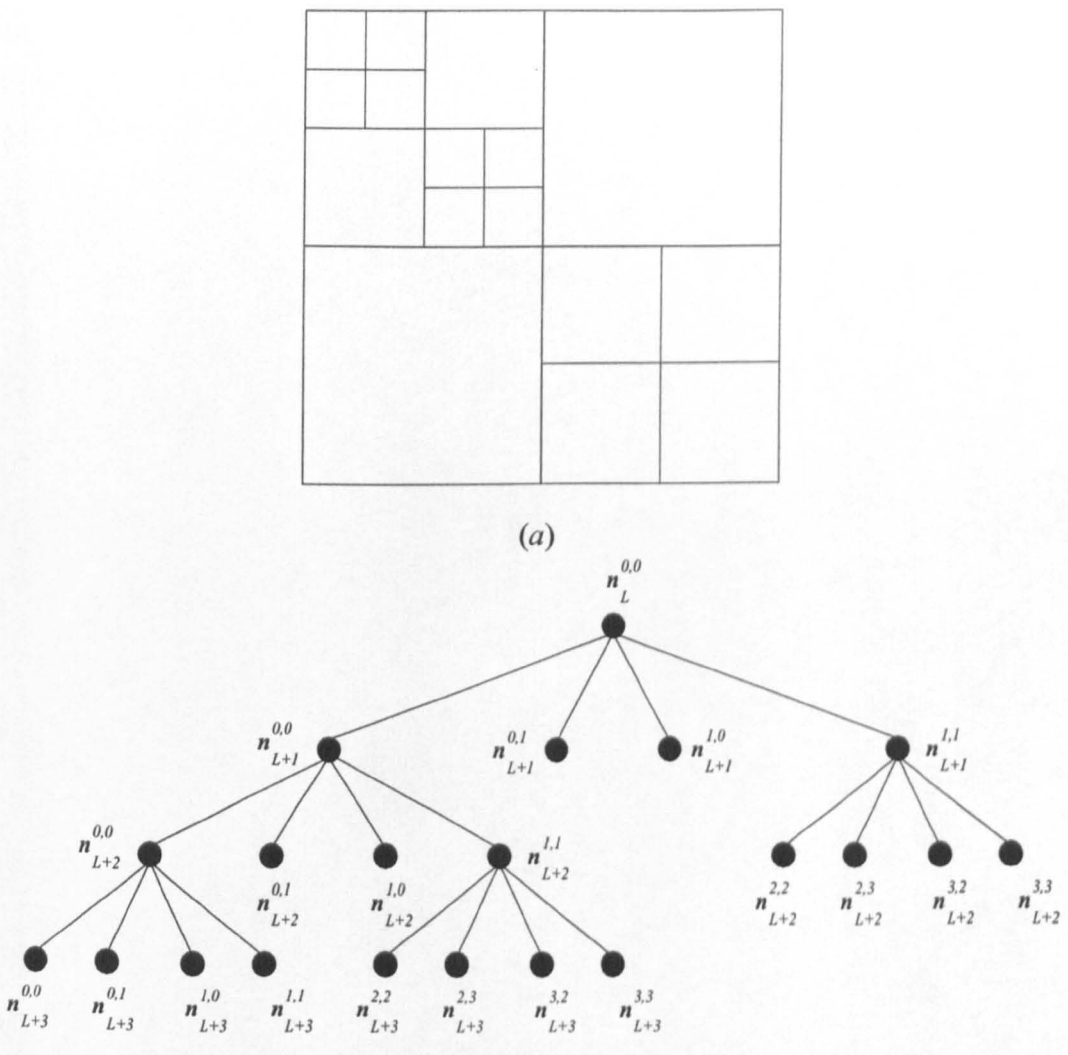


Figure 5.4: Subspace tree for a 3-level wavelet packet decomposition

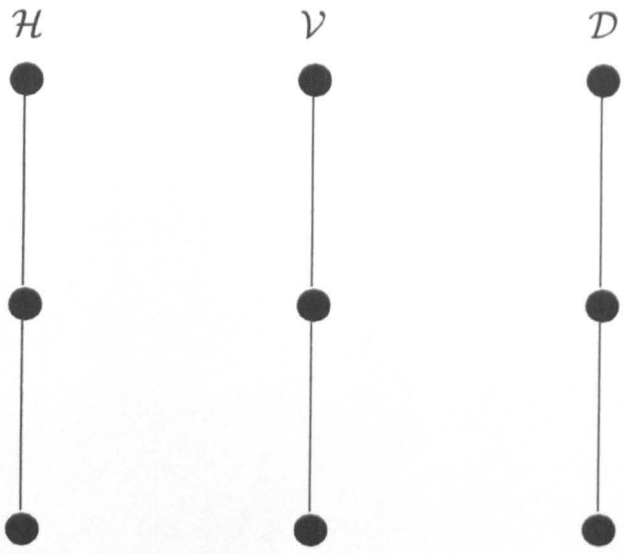


Figure 5.5: Compatible zerotrees for the subband families $\mathcal{H}$, $\mathcal{V}$, and $\mathcal{D}$ of a 3-level wavelet decomposition

tion compatibility, as will be obvious from the rules for their construction. It is to be noted that the compatible zerotrees differ from the spatial orientation trees of [71], in the sense that the nodes of a compatible zerotree represent a full subband rather than one or more transform coefficients. For instance, the compatible zerotrees associated with the three families $\mathcal{H}$, $\mathcal{V}$, and $\mathcal{D}$ of a 3-level wavelet decomposition are shown in Figure 5.5.

In a wavelet packet decomposition, the fact that the subbands of all three families $\mathcal{H}$, $\mathcal{V}$, and $\mathcal{D}$ can be further decomposed, up to the coarsest scale, makes the construction of such a tree structure non-trivial. In the next section, we present a solution to the problem when a child node in the compatible zerotree is at a coarser scale than the parent node. The set of rules required to construct compatible zerotrees, an algorithm for the construction of compatible zerotrees using these rules and the detection of zerotree root nodes in subbands belonging to such tree structures are provided in Section 5.2.2.

5.2.1 Parenting Conflict

As mentioned earlier, the high frequency subbands in a wavelet packet decomposition can be further decomposed in order to adapt the basis to the image contents. In a pre-decomposed tree of one of these families of subbands ($\mathcal{H}$, $\mathcal{V}$, and $\mathcal{D}$), if a child node is decomposed into only four subbands at the next coarser scale, the parent-offspring relations are easy to establish, as shown in Figure 5.6(*a*). However, if any of these four child nodes is decomposed any further, as shown in Figure 5.6(*b*), the resulting subbands C_1, C_2, C_3, and C_4 would be at a coarser scale than the parent node (subband) LH_2. This results in the association of each coefficient of such a child node to multiple parent coefficients, in the parent node, giving rise to a *parenting conflict.* In other words, there are four candidate coefficients in LH_2 claiming the parenthood of each of the four corresponding coefficients belonging to the child nodes C_i $(i = 1, \ldots, 4)$. In [103], the best basis was selected in such a way that whenever a conflict like this arises, the four children are merged so as to resolve the conflict. This suboptimal approach constrains selection of the basis at the cost of the loss of freedom in adapting the wavelet packet basis to the contents of a given signal. In order to resolve the parenting conflict, we suggest the following solution. Due to their orientation and scale compatibility with the subband LH_3, these child nodes C_i $(i = 1, \ldots, 4)$ can be moved up in the tree so that they are linked directly to LH_3, the root node of the compatible zerotree associated with the family $\mathcal{V}$ of subbands. This allows us to generate the compatible zerotree structure with no restriction on selection of the wavelet packet basis.

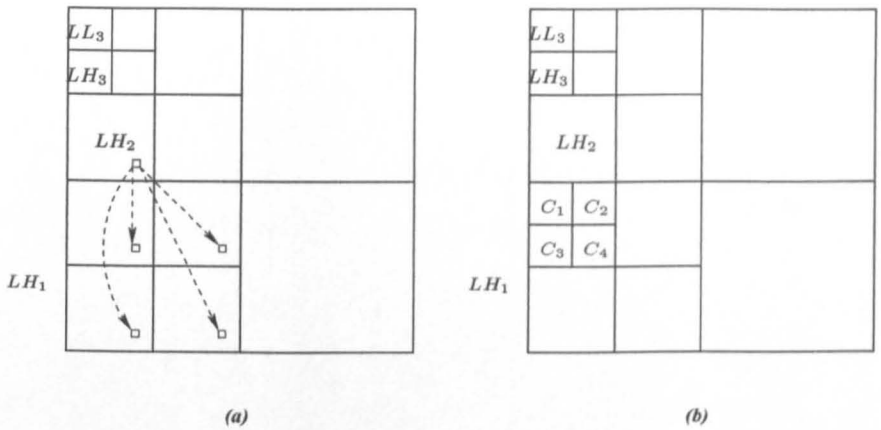


Figure 5.6: Parenting conflict in a wavelet packet decomposition
(a) Wavelet packet subband decomposition with no parenting conflict (b) Wavelet packet subband decomposition with parenting conflict arising from the split of one of the child nodes of the vertical high frequency subband LH_1

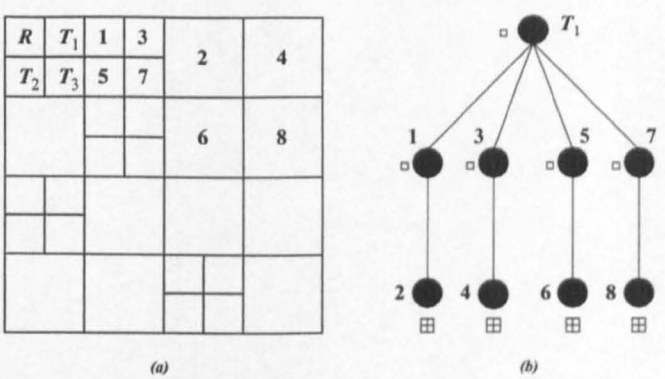


Figure 5.7: (a) Sample geometry of a 3-level wavelet packet decomposition, and (b) the parent-offspring dependences for compatible zerotree originating from T_1

5.2.2 Rules for Generating the Compatible Zerotrees

Once constructed for a given wavelet packet basis, the compatible zerotrees can be used to find the coefficients which are zerotree roots for various thresholds. In this section, a set of rules is defined so that the overall zerotree structure can be constructed for an arbitrary wavelet packet basis. The only assumption made is that the selected basis has the lowest frequency subband at the coarsest scale. This is a reasonable assumption, which is based upon the fact that a significant amount of the signal energy is concentrated in the lowest frequency subband, which is most likely not to be merged by any of the tree pruning algorithms.

Consider the sample segmentation shown in Figure 5.7(a). Let R denote the node representing the lowest frequency subband, situated in the top-left corner of a conventional subband decomposition. Then R represents the root node of an overall compatible zerotree with T_1, T_2, and T_3 - which represent the coarsest scale high frequency subbands - being its immediate children, as shown in Figure 5.7(a). These child nodes are themselves roots of three compatible zerotrees corresponding to the families $\mathcal{H}$, $\mathcal{V}$, and $\mathcal{D}$ respectively. These compatible zerotrees are separately generated, only once for a given wavelet packet basis, in a recursive manner, using the rules described below. In the description of these rules, "node X is followed by node Y" refers to the situation where node X is at a higher level (or coarser scale of pre-decomposed high frequency subbands, as in a wavelet decomposition) than the node Y in the hierarchy of subbands in a given family tree.

a) If a node P at a coarser scale is followed only by a node C at the next finer scale (as in a wavelet transform), the node P is declared as a parent of C.

b) If a node P is followed by four nodes C_1, C_2, C_3, and C_4 (at the same scale), then P is declared to be the parent of all these four nodes.

c) If four subbands P_1, P_2, P_3, and P_4 at a coarser scale are followed by four subbands C_1, C_2, C_3, and C_4 at the next finer scale, then node P_i is declared to be the parent of node C_i (for $i = 1, 2, 3, 4$).

d) If a node P is at a finer scale than four of its children, say C_1, C_2, C_3, and C_4, then P is disregarded as being the parent of all these nodes and all of them are moved in the tree under a node at the same or a coarser scale.

In a compatible zerotree, the parent-offspring relationships are established by looking at the scales of subband nodes in the tree. Unlike zerotrees for wavelet subbands, where each parent subband node is followed by exactly one child subband of similar orientation at the next finer scale and thus each coefficient of the parent node is associated with four coefficients of the child node at some spatial location, an intermediate node in a compatible zerotree, in general, can have more than one child node. What determines the parent-offspring relationships between their coefficients is the difference in scale of the parent and child nodes. For instance, if both parent and the child nodes are at the same scale, each coefficient of the parent node will have exactly one offspring coefficient at same spatial location in the child node, whereas if the parent node is at the immediate coarser scale than the child node, then each coefficient of the parent node is associated with four coefficients at the same spatial location in the child node.

The construction of compatible zerotrees proceeds in two steps. In the first step, the primary compatible zerotrees corresponding to the subband families $\mathcal{H}$, $\mathcal{V}$, and $\mathcal{D}$ are constructed using rules *a*, *b*, and *c*. In the second step, the overall tree is reorganized using rule *d*, in order to resolve any parenting conflicts. Let us consider the segmentation shown in Figure 5.7(*a*) to explain these rules. The primary compatible zerotrees T_1, T_2, and T_3 are generated, as shown in Figure 5.8, in the first step. The overall tree is re-organized (as shown in Figure 5.9) using rule *d* in order to resolve the parenting

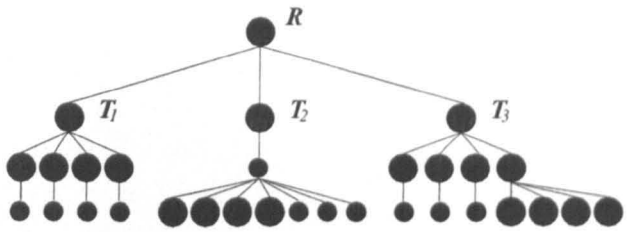


Figure 5.8: The overall compatible zerotree structure comprising of the three primary compatible zerotrees T_1, T_2, and T_3; each node represents a whole wavelet packet subband, and the node radius corresponds to the support of wavelet packets at that transform level.

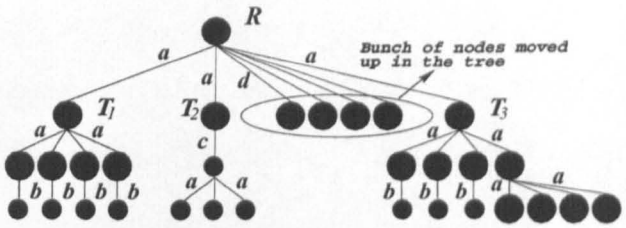


Figure 5.9: The overall compatible zerotree structure, after re-organization to resolve the parenting conflicts; the edge labels depict the rules used to generate the links between parent and the child nodes.

conflict under node P. This rule first identifies nodes with the parenting conflicts and when such nodes are found, the whole bunch (consisting of C_1, C_2, C_3, and C_4) is plucked from its current position in the tree. The algorithm then climbs up the tree to look for an appropriate node, a compatible ancestor at the nearest scale and having the nearest orientation, and glues the bunch under this newly found compatible parent. The parent-offspring dependences for the primary compatible zerotree T_1 are shown in Figure 5.7(*b*).

Given an arbitrarily segmented wavelet packet basis, compatible zerotrees are constructed using the algorithm described above. In order to be able to make predictions for wavelet packet coefficients at finer scale subbands, the *compatible zerotree hypothesis* is defined as follows. *If a wavelet packet coefficient of a node from the compatible zerotree is insignificant, it is more likely that the wavelet packet coefficients at similar*

spatial locations in all the descendant nodes of the same compatible zerotree will be insignificant as well.

Compatible zerotrees provide us a convenient way of making predictions about the insignificance of corresponding coefficients in the child nodes, given the significance of a coefficient belonging to the parent node. It is to be noted here that the compatible zerotrees are used for the detection of coefficients which are roots of the zerotrees. The encoding of zerotree codewords and information related to the significance of coefficients with respect to a threshold value follows the detection. Details of the algorithms used for the detection of zerotree root coefficients in each of the three primary compatible zerotrees and for the encoding are provided in [65, 64].

5.2.3 Testing the Compatible Zerotree Hypothesis

The success of the compatible zerotree hypothesis for wavelet packets, as defined in Section 5.2.2, needs to be tested, and we employ two empirical ways of doing so. First, the amplitude of transform coefficients is plotted for all the subbands organized both in the ordinary increasing frequency order and in the compatible zerotrees. Consider the plots of wavelet coefficient amplitude against the coefficient indices for a 5-level wavelet decomposition of the 512×512 *Barbara* image, as shown in Figures 5.10 and 5.11. The amplitude axis in both plots was restricted to ± 1000 to facilitate the display of smaller coefficients. While the subbands were arranged in an increasing frequency order for the plot in Figure 5.10, the plot in Figure 5.11 refers to the subbands as organized in the compatible zerotrees. Similar plots of wavelet packet coefficients' amplitudes against the coefficient indices for the same image using the basis geometry shown in Figure 5.12(*a*) (the corresponding compatible zerotrees are also shown in the Figure 5.12(*b*) with a numbering generated in a recursive manner) are given in

Figures 5.13 and 5.14. In all plots corresponding to the subbands organized in the compatible zerotrees, three families of primary compatible zerotrees T_1, T_2, and T_3 are clearly visible showing that the new arrangement is successful in isolating the coefficients of similar orientation in a hierarchy to be used for prediction in a top-down fashion. Zoomed plots of these families, as shown in Figure 5.15, exhibit more sub-families organized within these compatible zerotrees.

An alternate approach to test the success of compatible zerotree hypothesis is to compute the conditional probability $p(x|y)$ of the significance of a child coefficient x given the significance of its parent coefficient y for a specific threshold value. We plotted both joint and the conditional histograms to verify how successfully the prediction for insignificance of a coefficient in a child subband can be made given the insignificance of its parent coefficient. The joint and conditional histograms for some parent subbands and their immediate children using the subband decomposition of Figure 5.12(a) for *Barbara* are shown in Figures 5.16–5.19. The concentration of these histograms in the low significance range of child coefficients throughout the range of parent coefficients is encouraging evidence for the organization of these subbands in a compatible tree order.

5.3 Analysis of the Zerotree Quantization

If X and Y denote two random variables representing the significance of child and parent coefficients respectively, then the following relation

$$H(X|Y) = H(X) - I(X;Y),$$

tells us that encoding the conditioned significance is more efficient than just encoding the individual coefficients' significance, if the mutual information $I(X;Y)$ is positive.

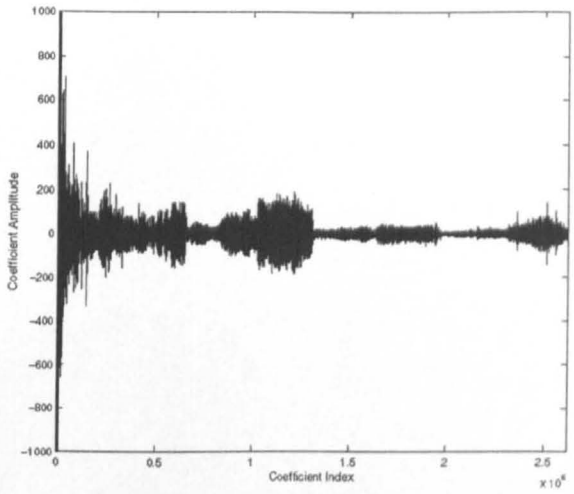


Figure 5.10: Plot of wavelet coefficients' amplitude versus coefficient indices for 5-level wavelet decomposition of 512×512 *Barbara* image; the subbands were arranged in an increasing frequency order.

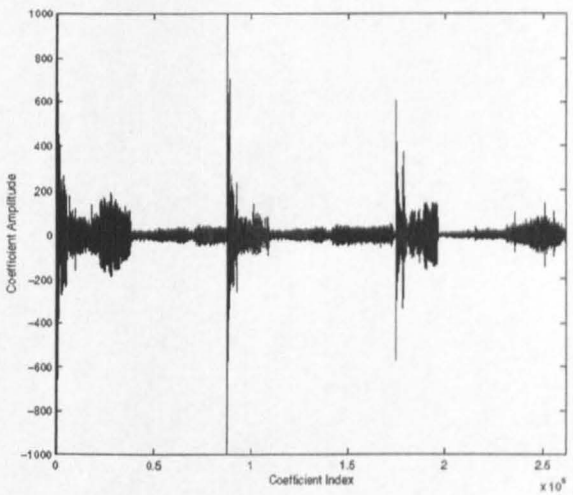


Figure 5.11: Plot of wavelet coefficients' amplitude versus coefficient indices for 5-level wavelet decomposition of 512×512 *Barbara* image; the subbands were organized in a compatible zerotree order

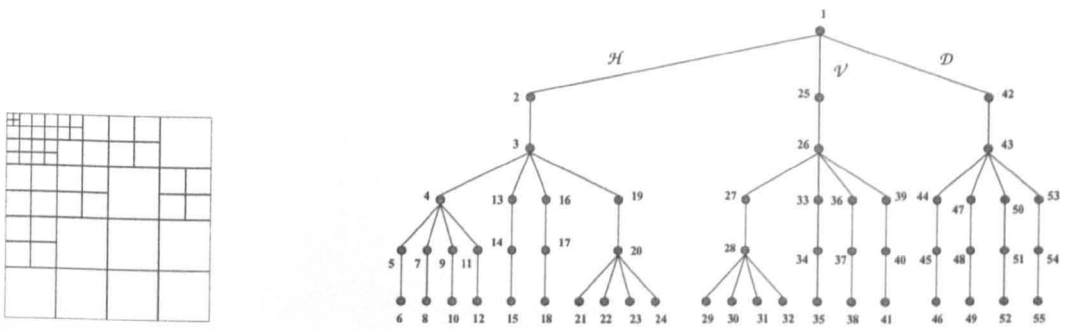


Figure 5.12: (*a*) Wavelet packet decomposition used for illustration of the compatible zerotree hypothesis for wavelet packets, (*b*) Compatible trees for the decomposition in (*a*)

The performance of zerotree encoding depends largely upon the value of $I(X;Y)$ for a given basis. The larger the value of this measure, the more efficient the encoding is. Given a basis $\mathcal{B}$ for an image, this value determines how *friendly* the basis $\mathcal{B}$ would be to the zerotree quantization method, in the spirit of the new basis selection paradigm described in the previous chapter.

One way of taking into account the mutual information between the parent and children subbands in a compatible zerotree organization, described above, is to estimate the cost of zerotree quantization without actually encoding the wavelet packet coefficients (in order to speed up the otherwise very slow basis selection process). The cost $\mathcal{C}(f,\mathcal{B})$ of encoding the coefficients of a D-level wavelet packet basis $\mathcal{B}$ can be written as

$$\mathcal{C}(f,\mathcal{B}) = \sum_{i=0}^{n-1} [\mathcal{C}_{sm}(T_i) + \mathcal{C}_{re}(T_i)]$$

where $T_i = T_0/2^i$ is the threshold value used at the ith iteration, n denotes the number of stages of encoding, and $\mathcal{C}_{sm}$ and $\mathcal{C}_{re}$ denote the costs of encoding the significance map and the refinement information. It was observed that the symbol used to encode the refinement information using Shapiro's method is nearly random, which leads to the conclusion that it would suffice to estimate the first term in the above expression.

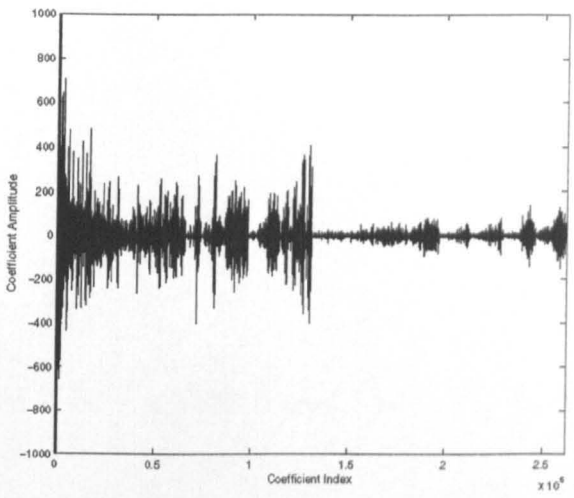


Figure 5.13: Plot of wavelet packet coefficients' amplitude versus coefficient indices for the wavelet packet decomposition shown in Figure 5.12 (*a*) of the 512×512 *Barbara* image; the subbands were arranged in an increasing frequency order

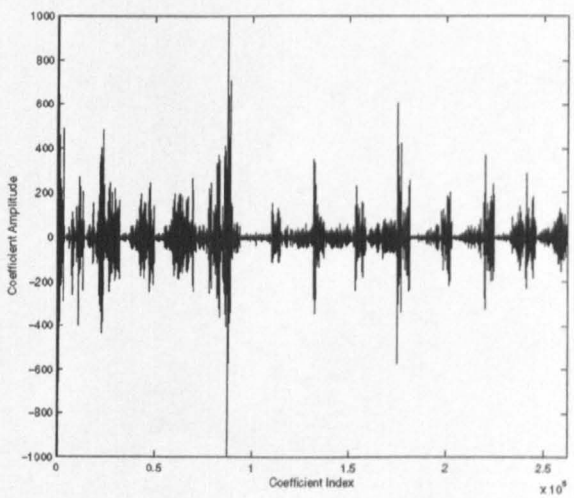


Figure 5.14: Plot of wavelet packet coefficients' amplitude versus coefficient indices for the wavelet packet decomposition shown in Figure 5.12 (*a*) of the 512×512 *Barbara* image; the subbands were organized in a compatible zerotree order

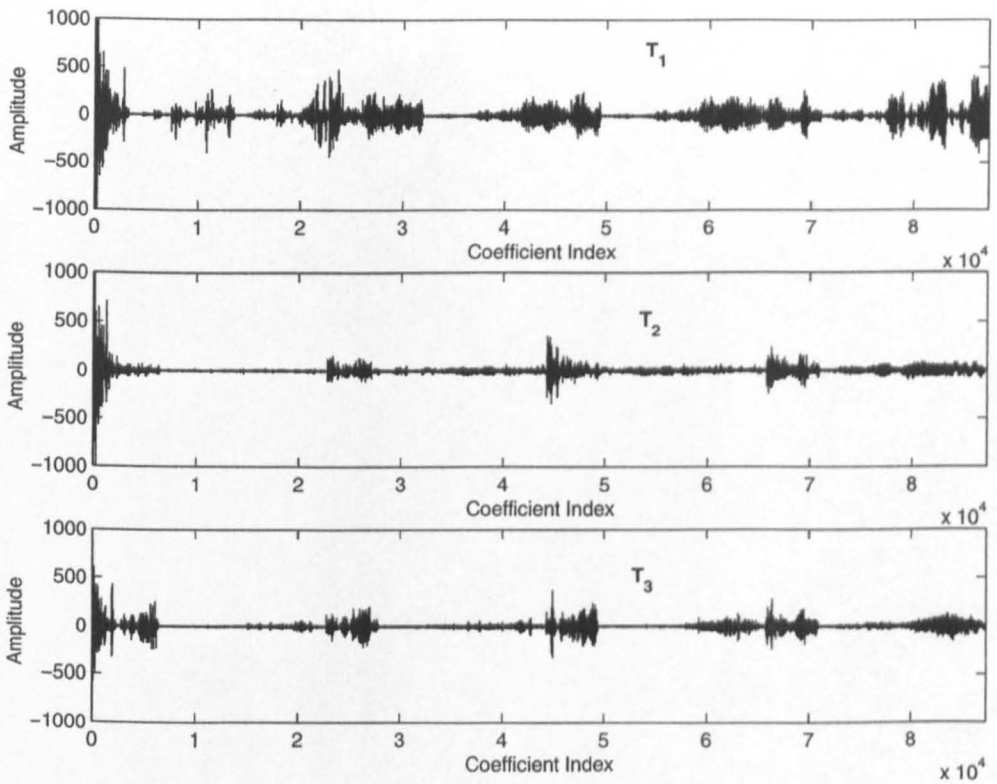


Figure 5.15: Zoomed sections of Figure 5.14
The families T_1, T_2, and T_3 showing more sub-families within these compatible zerotrees

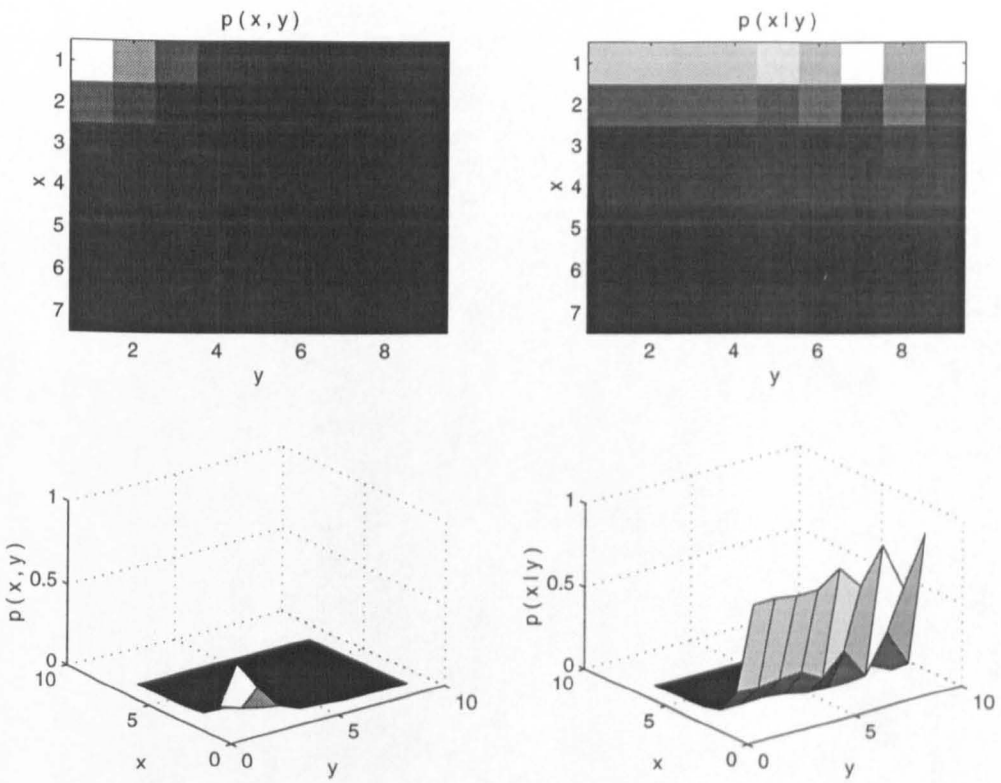


Figure 5.16: Joint and conditional histograms for subband numbered 3 and its immediate children (binsize=80)

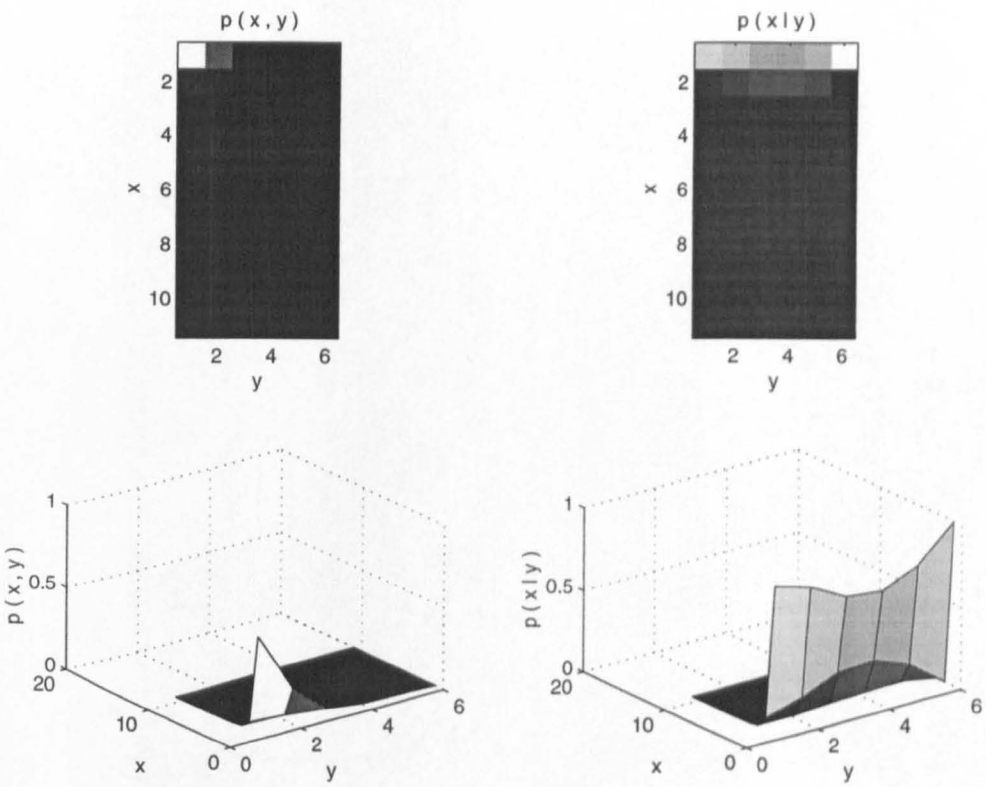


Figure 5.17: Joint and conditional histograms for subband numbered 4 and its immediate children (binsize=50)

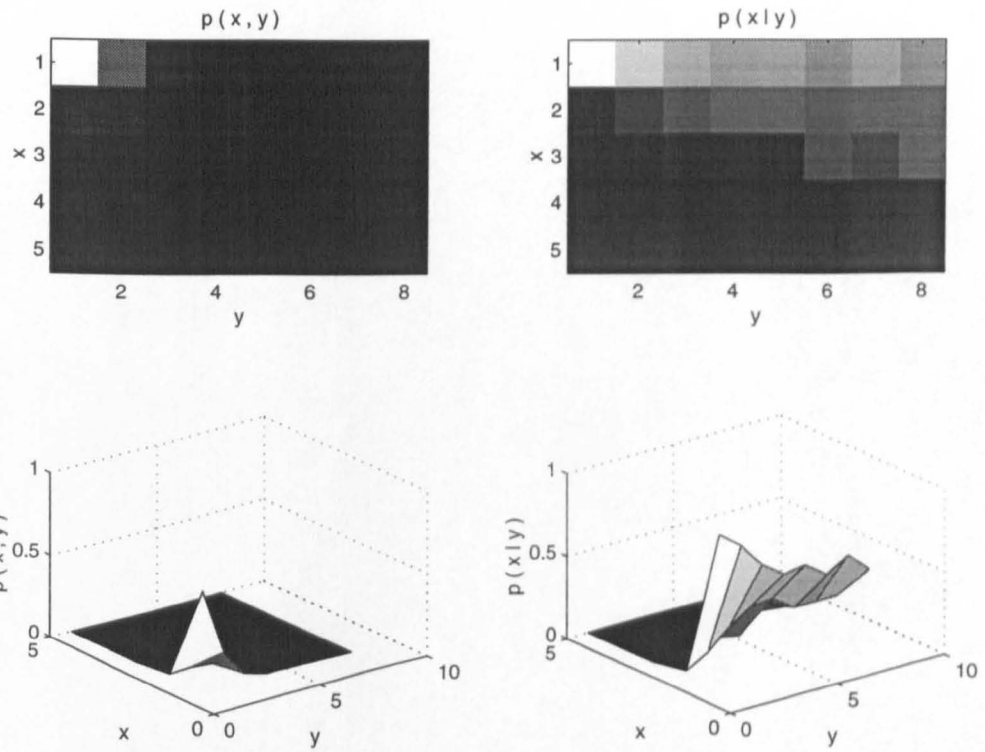


Figure 5.18: Joint and conditional histograms for subband numbered 26 and its immediate children (binsize=100)

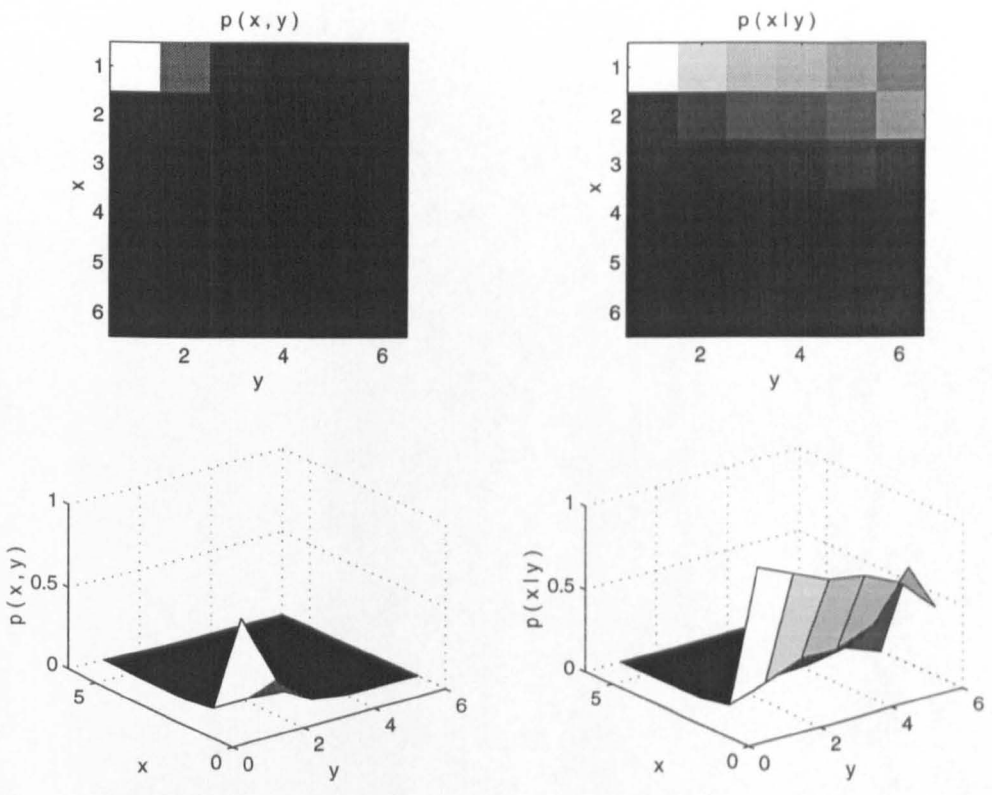


Figure 5.19: Joint and conditional histograms for subband numbered 43 and its immediate children (binsize=80)

The cost C_{sm} can be estimated by computing the entropy of a discrete random variable whose value is drawn from the set of codewords used to encode the significance map. These codewords include two symbols (0 and 1) to represent whether a coefficient is significant or not, and a zerotree symbol whose probability can be computed as follows.

Due to the fact that the significance of child coefficients in a subband is related only to the significance of the parent coefficient, the subbands (nodes) belonging to each family of the compatible zerotrees can be modelled as a non-homogeneous Markov chain (MC) with time replaced by node depth in the tree, i.e. the root compatible zerotree. Let X_j denote a random variable corresponding to the coefficients, of all nodes at tree depth j, that are the children of coefficients belonging to the corresponding parent nodes at tree depth $j-1$. The sample space for these random variables $X_1, X_2, \ldots, X_D$, where D denotes depth of the tree or the number of transform levels, is $\{0, 1\}$, where a value of 0 denotes that the coefficient is insignificant with respect to a threshold and 1 denotes its magnitude being larger than the threshold.

Let $P_k(0)$ denote $Pr(X_k = 0)$, the probability of a coefficient belonging to subband nodes at tree depth k being insignificant, and $P_{j,i}(0|0)$ denote the probability of all child coefficients at depth j to be insignificant given all of their corresponding parent coefficients at the previous depth i are insignificant[2]. Let $P_k(0)$ denote the joint probability of all the coefficients originating from nodes at tree depth k and all their child coefficients being insignificant. In other words, it denotes the probability of a zerotree of length $D-k$, which consists of $(4^{D-k+1}-1)$ coefficients in a wavelet zerotree. According to the multidimensional pmf theorem of Markov chains [87], $P_k(0)$ is given

[2]It was observed that given a coefficient and all its child coefficients are insignificant, it is very likely that its siblings and all their children are insignificant too. The probability $P_{j,i}(0|0)$ can, therefore, be approximated by the probability of all four child coefficients at depth j being insignificant given their parent coefficient at depth i is insignificant.

by

$$P_k(0) = P_k(0)P_{k+1,k}(0|0)P_{k+2,k+1}(0|0)\cdots P_{D,D-1}(0|0) \tag{5.1}$$

or

$$P_k(0) = P_k(0)\prod_{i=k}^{D-1} P_{i+1,i}(0|0) \tag{5.2}$$

Zerotrees of length l, however, contain all zerotrees of length $l-1$ and less. A recursive update of the following sort is, therefore, required to adjust the number of zerotrees of different lengths.

$$P_i(0) \longleftarrow P_i(0) - P_{i-1}(0) \tag{5.3}$$

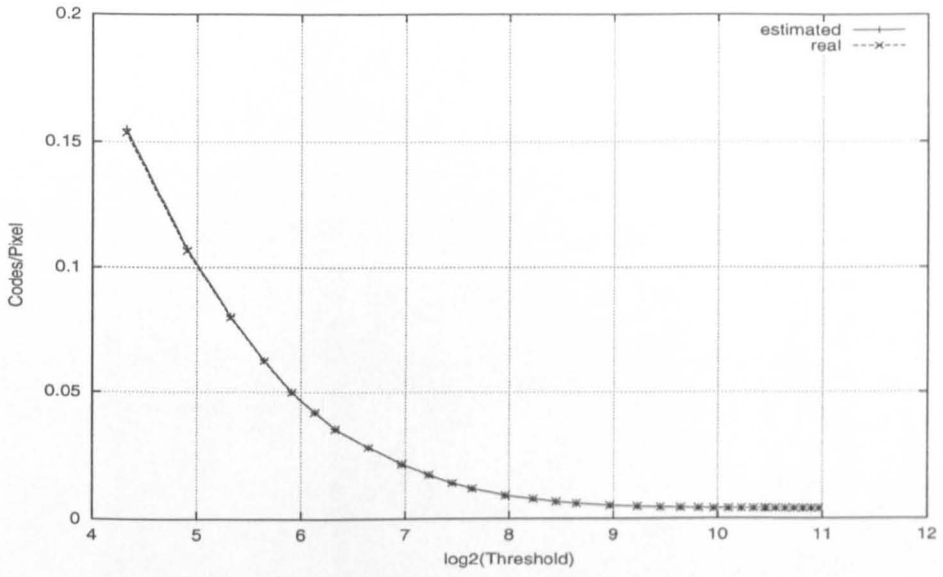
for $i = D-1, D-2, \ldots, 2$. The cost C_{sm} can now be computed as follows

$$C_{sm} = H_s - \sum_{k=1}^{D} P_k(0)\log P_k(0) \tag{5.4}$$

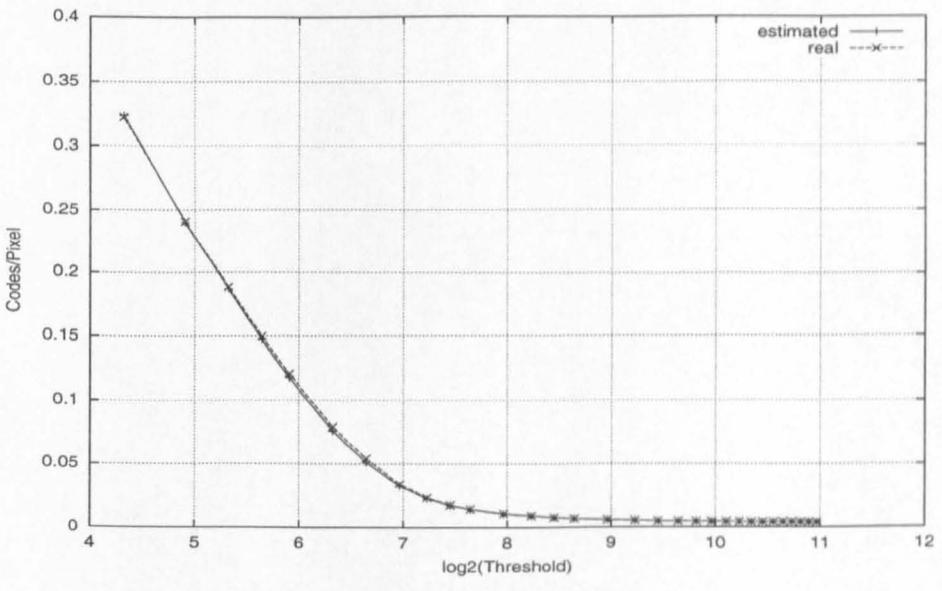
where

$$H_s = -\sum_{k=1}^{D} (P_k(0)\log P_k(0) + P_k(1)\log P_k(1)) \tag{5.5}$$

gives the entropy of the one and zero codewords identifying respectively the position of significant coefficients and those insignificant coefficients which are not contained in any of the zerotrees. Figure 5.20 shows the graph of the cost of encoding the significance map (both estimated and real) in terms of number of codewords per pixel plotted against various threshold values for both *Lena* and *Barbara* images. From these graphs, it is clear that this estimation works better for the former image than for the latter, due to its being more complex. The MC based computation of the cost of encoding the significance map proves to be a good estimate, particularly at large threshold values, which correspond to low bit rates.



(*a*)



(*b*)

Figure 5.20: Cost of encoding the significance map vs. threshold value
(*a*) Lena and (*b*) Barbara

5.4 The Coder Algorithm

As discussed earlier in the previous chapter, the use of a bottom-up search method along with a cost function that takes into account the quantization strategy ensures the selection of a best basis for compressing a given image using that particular quantization method. Based on the cost estimate described in the previous section, a simple heuristic can be devised to select what can be termed as *zerotree friendly* wavelet packet basis. The algorithm for selection of the best basis unifying the new paradigm, presented in the previous chapter, and the cost estimate described in the previous section is given in Figure 5.21.

> Let the given image f be of size $N \times N$. Let $\mathcal{B}_j^{p,q}$ be the wavelet packet basis associated with the subband node $n_j^{p,q}$ undecomposed and $\mathcal{B}_{j+1}^{p,q}$ be the wavelet packet basis associated with the four child subbands of $n_j^{p,q}$ (i.e. it is split into four nodes which are: $n_{j+1}^{2p,2q}$, $n_{j+1}^{2p+1,2q}$, $n_{j+1}^{2p,2q+1}$, and $n_{j+1}^{2p+1,2q+1}$). The best wavelet packet basis for zerotree quantization up to a depth D is selected as follows.
>
> 1. Compute the D-level full-tree wavelet packet decomposition of the given image, by splitting all the intermediate nodes (subbands).
> 2. Initialize $d \longleftarrow D - 1$.
> 3. For $0 \leq p \leq 2^d - 1$, $0 \leq q \leq 2^d - 1$, do the following:
>   a) Compute $\mathcal{C}(f, \mathcal{B}_j^{p,q})$ and $\mathcal{C}(f, \mathcal{B}_{j+1}^{p,q})$ using equations (5.2)-(5.5).
>   b) If $\mathcal{C}(f, \mathcal{B}_j^{p,q}) > \mathcal{C}(f, \mathcal{B}_{j+1}^{p,q})$, then keep the four child subbands at depth $d+1$, otherwise merge them to get n_j^p.
> 4. Assign $d \longleftarrow d - 1$.
> 5. If $d < 0$, then stop, otherwise go to step 3.
>
> Figure 5.21: Best basis selection algorithm for zerotree quantization

Given a basis, the compatible zerotrees can be generated using rules of Section 5.2.2. Once the compatible zerotrees have been generated, based upon knowledge of the best basis, the coding is performed by successively encoding the significance information about the coefficients, as they appear in the subbands in increasing frequency

order, and the refinement information until the bit budget has expired or the encoded bitstream terminates, whichever happens earlier. A detailed flowchart of the coder algorithm is given in Figure 5.22 (where the basis $\mathcal{B}$ is selected and its corresponding transform coefficients are computed using the algorithm described in Figure 5.21). This algorithm requires only one list of detected coefficients (LDC) to be maintained, similar to [66], as opposed to two (or three) lists kept by the EZW (or SPIHT). The decoder first reads the geometry of the best basis and generates the compatible zerotrees. It then proceeds by entropy decoding and interpreting the codewords until the bit budget is expired. The decoder is also capable of generating an approximation to the original image at any given bit rate or quality, as long as the minimum required bits are available for doing so.

5.5 Experimental Results and Discussion

The idea of compatible zerotree quantization was combined with the wavelet packet basis for progressive image coding and its performance was tested on four standard 8-bit greyscale 512×512 images - *Lena*, *Goldhill*, *Barbara*, and *Fingerprints* - using both a wavelet basis and a zerotree friendly wavelet packet basis. The latter of these bases was selected using the encoding cost discussed earlier in Section 5.3. For all the experiments, the factorized 9-7 biorthogonal filters [55] were used for efficiently computing the wavelet packet transform. Results for the performance of both variations of the CZQ coder - that is, with the wavelet basis CZQ-$\mathcal{WV}$ and with the wavelet packet basis CZQ-$\mathcal{WP}$ - for all the test images are presented in Tables 5.1–5.4. The measure used to describe the performance of each coder is the PSNR. Although it is well known that PSNR is not a representative measure of the performance from a perceptual viewpoint [37], it is still widely used for comparing the coding performances

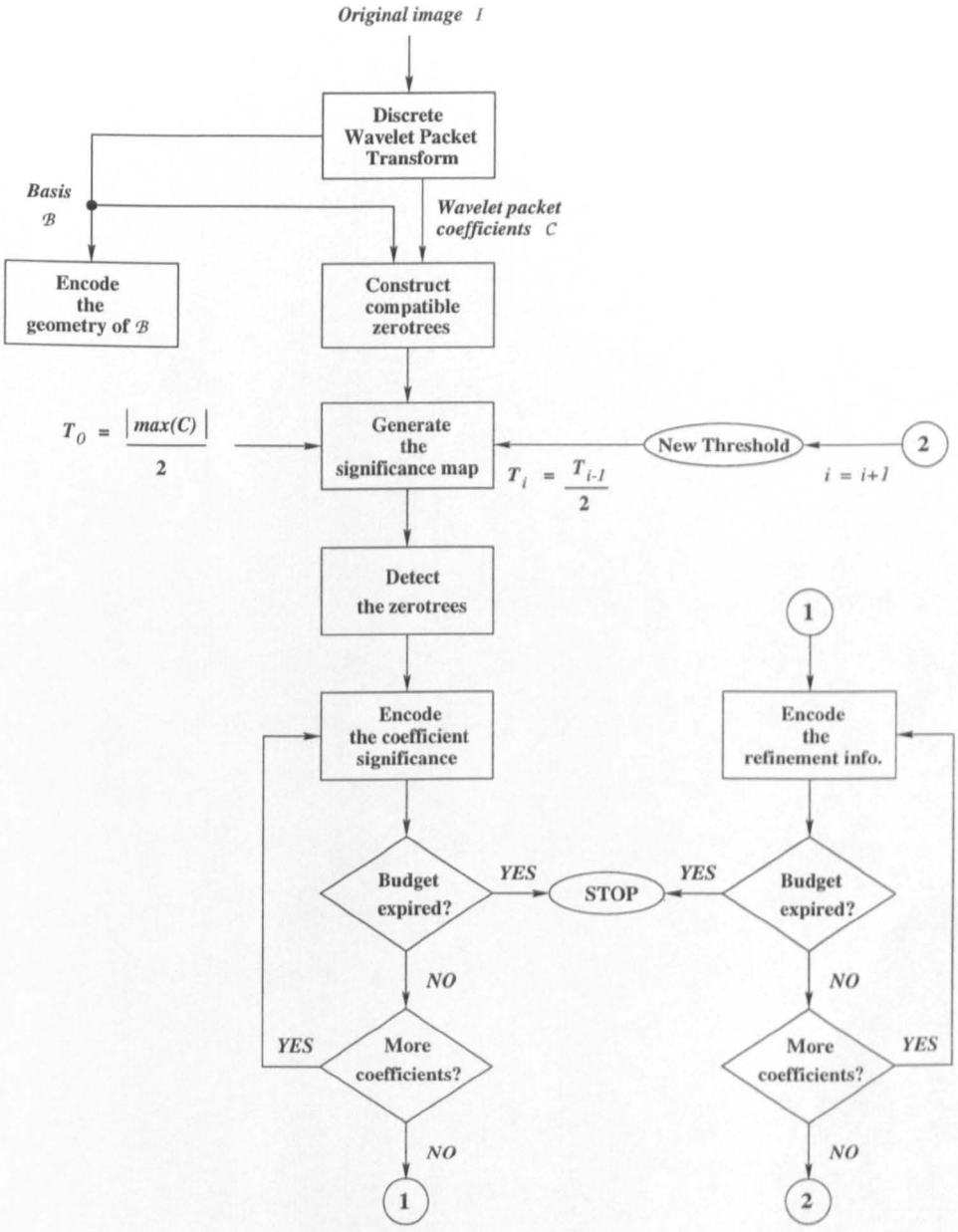


Figure 5.22: Flowchart of the progressive wavelet packet image coding algorithm with the compatible zerotree quantization

in quantitative terms.

Bit rate (bpp.)	Compression Ratio (:1)	PSNR (dB)	
		CZQ-$\mathcal{WV}$	CZQ-$\mathcal{WP}$
1.0	8	40.06	40.10
0.8	10	38.74	38.85
0.7	11.43	38.35	38.47
0.6	13.33	37.96	38.08
0.5	16	37.50	37.55
0.4	20	35.86	36.02
0.3	26.67	34.89	35.06
0.25	32	34.49	34.56
0.2	40	32.89	32.90
0.1	80	29.90	29.95
0.08	100	28.89	28.93

Table 5.1: CZQ compression results for 512×512 *Lena* image

While being capable of progressively reconstructing the encoded image and being relatively faster than other wavelet packet coders [55], the CZQ-$\mathcal{WP}$ coder performs comparably well. The coding gains achieved by it on top of CZQ-$\mathcal{WV}$ (nearly 0.1dB for *Lena*, 0.1–0.25dB for *Goldhill*, 0.6–1.5dB for *Barbara*, and 0.4–0.7dB for *Fingerprints*) empirically demonstrate the success of the compatible zerotree hypothesis. Figure 5.23 shows the geometries of 2-d wavelet packet bases selected for all test images. As expected, the basis selected for *Lena* closely resembles a wavelet basis, due to its being a smooth image.

A closer look at the reconstructed images by CZQ-$\mathcal{WP}$ and SPIHT at 0.25 bpp. reveals that CZQ-$\mathcal{WP}$ yields better visual quality than SPIHT. Note, for instance, the quality of a portion (table cloth) of the reconstructed *Barbara* image encoded by CZQ-$\mathcal{WP}$ and SPIHT as shown in Figure 5.24, and the quality of a portion (central spiral) of the reconstructed *Fingerprints* image encoded by CZQ-$\mathcal{WP}$ and SPIHT as shown in Figure 5.25. It is worth noting, however, that the effectiveness of zerotree quantization

Bit rate (bpp.)	Compression Ratio (:1)	PSNR (dB)	
		CZQ-$\mathcal{WV}$	CZQ-$\mathcal{WP}$
1.0	8	34.84	35.01
0.8	10	34.16	34.30
0.7	11.43	33.81	33.91
0.6	13.33	32.63	32.83
0.5	16	31.69	31.94
0.4	20	31.02	31.25
0.3	26.67	30.36	30.54
0.25	32	29.75	29.88
0.2	40	28.81	28.98
0.1	80	27.35	27.48
0.08	100	26.60	26.69

Table 5.2: CZQ compression results for 512×512 *Goldhill* image

depends strongly upon the scale-invariance of edges among subbands of similar global orientation.

5.6 Chapter Summary

In this chapter, a general zerotree structure, termed the *compatible zerotree* structure, for an arbitrary wavelet packet basis was presented. A solution to the parenting conflict– a problem that occurs when a parent node in a wavelet packet subband hierarchical tree has fine resolution than one or more of its child nodes – was suggested. A set of rules was defined to construct the *compatible zerotrees* for any given wavelet packet decomposition. The wavelet packet subbands were modeled as a Markov chain in order to efficiently estimate the cost of the compatible zerotree quantization. Finally, a progressive wavelet packet image coding algorithm which combines this cost estimator with the new compatible zerotree quantization was presented. Experimental results using this new wavelet packet coder on a set of test images from diverse classes showed useful coding gains over its wavelet counterpart.

Bit rate (bpp.)	Compression Ratio (:1)	PSNR (dB)	
		CZQ-$\mathcal{WV}$	CZQ-$\mathcal{WP}$
1.0	8	35.14	36.15
0.8	10	32.64	33.91
0.7	11.43	32.04	33.21
0.6	13.33	31.50	32.56
0.5	16	30.28	31.60
0.4	20	28.41	29.85
0.3	26.67	27.50	28.77
0.25	32	27.12	28.12
0.2	40	25.25	26.64
0.1	80	23.64	24.27

Table 5.3: CZQ compression results for 512×512 *Barbara* image

Bit rate (bpp.)	Compression Ratio (:1)	PSNR (dB)	
		CZQ-$\mathcal{WV}$	CZQ-$\mathcal{WP}$
1.0	8	35.24	35.82
0.8	10	32.71	33.38
0.7	11.43	31.84	32.45
0.6	13.33	31.12	31.74
0.5	16	30.65	31.15
0.4	20	28.39	29.09
0.3	26.67	27.00	27.64
0.25	32	26.53	27.07
0.2	40	25.08	25.64
0.1	80	22.84	23.24

Table 5.4: CZQ compression results for 512×512 *Fingerprints* image

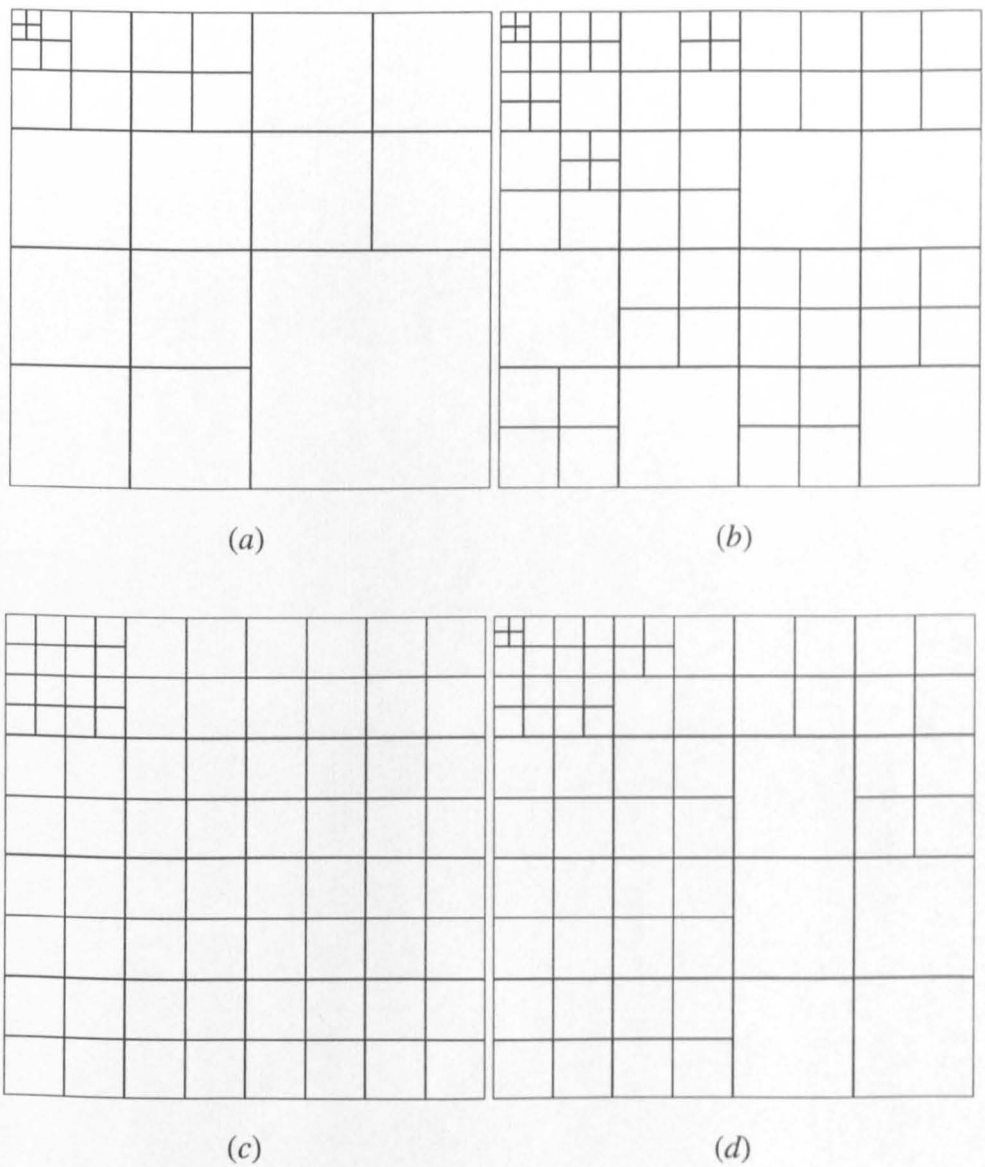
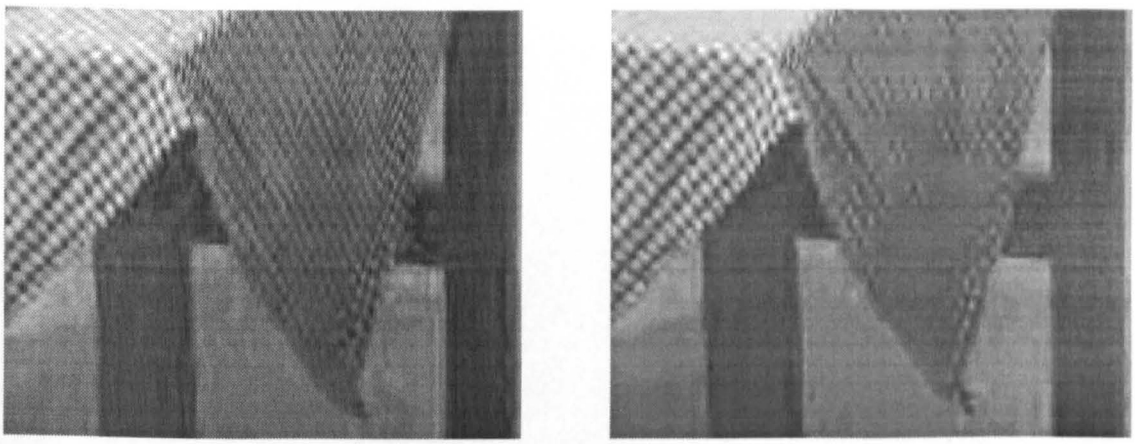
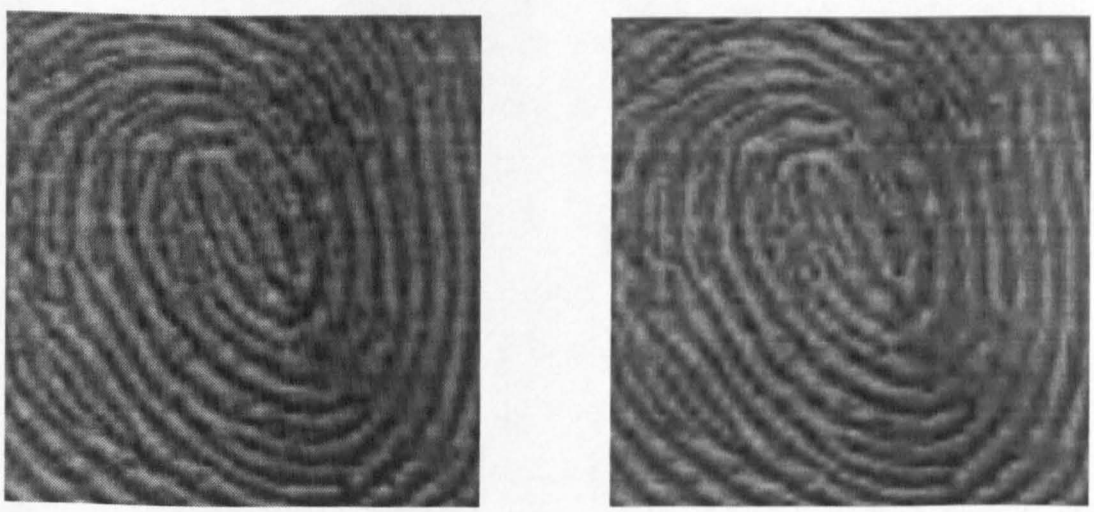


Figure 5.23: Selected 2-*d* wavelet packet bases used for encoding
(*a*) *Lena*, (*b*) *Goldhill*, (*c*) *Barbara*, and (*d*) *Fingerprints*



(a) (b)

Figure 5.24: Portion of *Barbara* (table cloth) encoded at 0.25 bpp.
(*a*) CZQ-$\mathcal{WP}$ and (*b*) SPIHT



(a) (b)

Figure 5.25: Portion of *Fingerprints* (central spiral) encoded at 0.25 bpp.
(*a*) CZQ-$\mathcal{WP}$ and (*b*) SPIHT

Chapter 6

Conclusions and Future Directions

In this thesis, we have discussed various approaches to the problem of adaptive wavelet transform based image coding and some issues involved therein.

6.1 Summary and Conclusions

In Chapter 2, issues related to the image representation for compression were discussed. After a review of the possible options, it was established that the wavelet transform is a suitable candidate for image representation in coding applications. An explanation of the wavelet subband decomposition of images with the help of filter banks was presented. Components of the transform coding framework other than the wavelet transform, i.e. quantization and entropy coding, were briefly discussed. A new improvement to dynamic, dictionary based entropy coding, based on an optimal parsing strategy was described. The idea of one-step lookahead parsing for dynamic dictionary based data compression methods satisfying the prefix property was applied to the LZW compression scheme. The experimental results show that this scheme produces a compressed file which is always smaller than the LZW compressed file and achieves an improvement of up to 31% in terms of the size of compressed file.

In Chapter 3, some important properties of the wavelet transform were studied from a compression viewpoint. It was argued that the decay of wavelet coefficients is directly related to compression performance at various bit rates. It was observed that the arithmetic coder almost always achieves better performance, in this application, than dictionary based entropy coding with flexible parsing. The PSNR results of a simple wavelet-thresholding image coder showed that better potential gains could be obtained by employing a more sophisticated quantization strategy. The idea of zerotree quantization was explained, and an analysis of both the EZW [73] and the SPIHT [71] algorithms was presented. The zerotree hypothesis of Shapiro was extended to include the lowest frequency subbands. After its experimental validation, a novel augmented zerotree wavelet image coding algorithm was presented, whose compression performance is comparable to the SPIHT coder. The coding results also demonstrated the fact that the wavelet transform based image coders do not perform as well on relatively complex images as on smooth images.

The importance of adapting the underlying wavelet basis according to contents of the given image was emphasized in Chapter 4. The use of the wavelet packet transform, whose time-frequency tiling is adapted to the given image, was advocated. Issues related to the selection of the *best* basis among the library of available wavelet packet bases were discussed. Upper bounds were given for the number of possible wavelet packet bases in case of both 1-d signals and images. It was proved that bottom-up search methods using additive cost functions are optimal for wavelet packet basis selection. A similar argument also disproves Taswell's conjecture about the so-called near-best basis. It was shown that the choice of a cost function can have a drastic effect on selection of the *best* basis, in turn producing different coding results using the same quantization method. In order to assess which of the two *best* bases selected using two different cost functions is better than the other, necessary and sufficient conditions

were presented. There are, however, two assumptions for the type of cost functions that these conditions apply to. The cost functions have to be concave and they should be additive with respect to the child nodes in a wavelet packet tree. A new paradigm was presented for wavelet packet basis selection which aims to unite the basis selection process with quantization strategy to achieve better compression performance. The motivation for using such a paradigm in a wavelet packet transform coding framework is the fact that the cost function used to select the best wavelet packet basis ought to have a direct relation with the quantization strategy being used. The basis selected by using Coifman-Wickerhauser entropy as a cost function, for instance, is very likely not to be zerotree friendly. Therefore, encoding the coefficients of such a basis using zerotree quantization will not be a good idea.

In Chapter 5, a general zerotree structure, termed the *compatible zerotree* structure, for an arbitrary wavelet packet basis was presented. A solution to the parenting conflict – a problem that occurs when a parent node in a wavelet packet subband hierarchical tree has higher resolution than one or more of its child nodes – was suggested. A set of rules was defined to construct *compatible zerotrees* for any given wavelet packet decomposition. The wavelet packet subbands were modelled as a Markov chain, in order to estimate the cost of the compatible zerotree quantization. This analysis can also be used to estimate the coding cost in case of the fixed wavelet zerotree coding. Finally, a progressive wavelet packet image coding algorithm which applies the new basis selection paradigm, combining this cost estimator with the new compatible zerotree quantization, was presented. Experimental results using this new wavelet packet coder on a set of test images from diverse classes of images showed significant coding gains over the simpler wavelet coder.

6.2 Limitations

All the experiments for various image coding algorithms presented in this thesis were conducted on greyscale images that were square (typically having a 512×512 resolution). A simple extension of these algorithms to non-square images can be easily obtained by making sure, for instance, that the number of rows and the number of columns of the image is divisible by 2^D where D is the number of transform levels. This can be done by a smooth extension, for example, of the image which extends the image in one or both directions, as required, by repeating the last pixel values from each row and column. These algorithms can also be extended for compressing colour images by transforming the image from an RGB-space into the YUV components. The luminance component Y can be treated similarly to a greyscale image, while the chrominance components U and V can be encoded separately.

The complexity of wavelet packet transform for an image of size $M \times N$ is $O(MND)$ as compared to $O(MN)$ for a wavelet transform, where D is the number of transform levels or the depth of full wavelet packet tree. Moreover, the new basis selection algorithm needs to compute the estimated cost of quantization $(4^{D-1} - 1)/3$ times. For large images and high numbers of transform levels, this can turn out to be computationally expensive. However, this may be well achievable as computational capabilities increase.

The only assumption made for the organization of wavelet packet subbands into a compatible zerotree is that the lowest frequency subband has the coarsest resolution. Although this assumption is based upon the fact that a significant amount of the signal energy is concentrated in the lowest frequency subband, which is most likely not to be merged, the possibility of the lowest frequency subband not having the coarsest resolution cannot be completely ruled out.

The wavelet coders presented in this thesis, like other wavelet coders, may at low bit rates cause *ringing artifacts* around the edges. This is mainly due to the quantization of relatively large high frequency wavelet coefficients that correspond to edges in the original image. The combining of low frequency coefficients with suppressed high frequency coefficients, when taking an inverse wavelet transform, produces such artifacts around the edges, instead of sharp edges.

The artifacts caused by the wavelet packet coder presented in Chapter 5 at low bit rates are of a slightly different nature. When high frequency coefficients belonging to a coarse wavelet packet subband, corresponding to wavelet packets of large support, are quantized, they may affect a larger neighbourhood than that caused by wavelets. This may result in introducing *artificial texture* artifacts in smooth areas of the image at low bit rates.

6.3 Future Directions

The rate-distortion derivations for wavelet transform coefficients, given in Chapter 3, can be applied to estimate the optimal step size for a given rate or distortion value. Moreover, based on the assumption that the wavelet or wavelet packets coefficients within a subband follow a Laplacian distribution, an efficient near optimal entropy constrained scalar quantizer [79] can be used to efficiently encode and reconstruct these coefficients.

A more sophisticated statistical correlation between the wavelet coefficients such as [11, 101] can be employed for entropy coding of the coefficients in order to take into account both intra-band and inter-band dependences.

The new paradigm for basis selection, presented in Chapter 4, can be applied to the co-

sine packet basis selection as well in order to select a better basis specifically designed for coding purposes. In this thesis, this paradigm was applied to wavelet packet basis selection for zerotree quantization only. Since zerotree quantization is not necessarily an optimal strategy for quantizing the wavelet packet subbands, it can be applied to incorporate other quantization methods into the wavelet packet transform coding framework.

6.4 Concluding Remarks

The problem of image coding using an adaptive wavelet transform was addressed in this thesis. Although there is still room for improvement, the coding results presented are among the best published to date.

Appendix A

Rate-Distortion Characteristics of Wavelet Transform

Assuming that the random variable representing the high frequency wavelet transform coefficients follows a memoryless Laplacian source with a zero mean, a relationship can be derived between the distortion (mean squared error) and rate (first-order entropy) in a uniform-quantizer-followed-by-entropy-coding setting. Let $p(x)$ denote the probability of occurrence, also known as the probability density function (pdf), of the source value x, given by

$$p(x) = \frac{\lambda}{2} e^{-\lambda|x|} \tag{A.1}$$

where λ is the distribution parameter related directly to the source variance σ^2 by $\lambda = \sqrt{2/\sigma^2}$. In order to investigate the distortion-rate function for this distribution, let us suppose that the uniform step size Δ produces $2n + 1$ equally spaced points $(x_{-n}, \ldots, x_{-1}, x_0, x_1, \ldots, x_n)$; $x_i = i\Delta$, yielding the distortion $D(\Delta)$ and requiring $R(\Delta)$ bits/sample, at the least, for encoding. The quantities $R(\Delta)$ and $D(\Delta)$ can thus be expressed as follows,

$$R(\Delta) = -\sum_i P(x_i) \log_2 P(x_i) \tag{A.2}$$

and

$$D(\Delta) = \sum_i d_i, \tag{A.3}$$

where

$$P(x_i) = \int_{a_i}^{b_i} p(x)dx \tag{A.4}$$

is the marginal pdf for the quantized source data, and

$$d_i = \int_{a_i}^{b_i} p(x)(x - x_i)^2 dx \tag{A.5}$$

is the distortion due to mapping of the source data values $x \in (a_i, b_i)$ to x_i, with $a_i = x_i - \frac{\Delta}{2}$ and $b_i = x_i + \frac{\Delta}{2}$. The new pdf $P(x_i)$ for quantized coefficients given in equation (A.4) can be simplified as follows,

$$P(x_i) = \frac{2\sinh\frac{\lambda\Delta}{2}}{\lambda} \cdot p(x_i) \tag{A.6}$$

The above equation implies that $P(x_i)$ is a constant multiple of $p(x_i)$, with the value of constant increasing monotonically with the step size Δ. The obvious fact that $p(x_i) = \Delta P(x_i)$ for small values of Δ is also clearly decipherable from this equation. Now putting the value of $p(x)$ from (A.1) in (A.5), we obtain

$$d_i = \frac{\lambda}{2}\int_{a_i}^{b_i}(x - x_i)^2 e^{-\lambda|x|}dx;\ \forall i,\ i{\neq}0$$

and

$$d_0 = \lambda\int_0^{\frac{\Delta}{2}} x^2 e^{-\lambda x}dx.$$

Integration of the right hand sides of above equations yields the following expression for d_i and d_0

$$d_i = \left\{\frac{\Delta^2}{4} + \frac{2}{\lambda^2} - \frac{\Delta}{\lambda}\coth\frac{\lambda\Delta}{2}\right\} P(x_i);\ \forall i,\ i{\neq}0 \tag{A.7}$$

and

$$d_0 = \frac{2}{\lambda^2} - \left(\frac{\Delta^2}{4} + \frac{2}{\lambda^2} + \frac{\Delta}{\lambda}\right) e^{-\frac{\lambda\Delta}{2}}. \tag{A.8}$$

We also know that $P(x_0) = P(0) = 2\int_0^{\Delta/2} p(x)dx = 1 - e^{-\frac{\lambda\Delta}{2}}$ and so

$$\sum_{i;\, i\neq 0} P(x_i) = 1 - P(0) = e^{-\frac{\lambda\Delta}{2}}.$$

Replacing d_i in (A.3) with the expressions in (A.7) and (A.8), we obtain the following expression for total distortion

$$\boxed{D(\Delta) = \frac{2}{\lambda^2} - \frac{\Delta}{\lambda}\left(1 + \coth\frac{\lambda\Delta}{2}\right)e^{-\frac{\lambda\Delta}{2}}} \tag{A.9}$$

The above equation confirms the fact that the distortion $D(\Delta)$ approaches the source variance $\sigma^2 = 2/\lambda^2$ as the step Δ increases. The rate $R(\Delta)$ can be re-written from (A.2) as

$$R(\Delta) = -P(0)\log_2 P(0) - \sum_{i;\, i\neq 0} \log_2\left(\sinh\frac{\lambda\Delta}{2}e^{-\lambda|x_i|}\right)P(x_i)$$

or

$$\boxed{R(\Delta) = -P(0)\log_2 P(0) - e^{-\frac{\lambda\Delta}{2}}\log_2\sinh\frac{\lambda\Delta}{2} + \frac{\lambda}{\log 2}S} \tag{A.10}$$

where $P(0) = 1 - e^{-\lambda\Delta/2}$ and S is given by

$$S = \sum_i |x_i| \cdot P(x_i) = 2\Delta \cdot \sinh\frac{\lambda\Delta}{2}\sum_{i=0}^{n} ie^{-i\lambda\Delta} \tag{A.11}$$

From (A.10), it is clear that as the step size Δ increases, both $e^{-\lambda\Delta/2}$ and S approach zero and thus $R(\Delta)$ has more contribution to its value from the source values $x \in [-\frac{\Delta}{2}, \frac{\Delta}{2}]$, which are quantized to zero, than for smaller values of Δ. The typical curves for $D(\Delta)$ and $R(\Delta)$ when plotted against Δ are shown in Figure A.1. As is clear from the corresponding expressions, $R(\Delta) \to 0$ and $D(\Delta) \to \sigma^2$ as the step size Δ becomes very large.

Equations (A.9) and (A.10) provide us a fast way of approximating the ($R(\Delta)$,$D(\Delta)$) points of the rate-distortion curve for data which we assume to follow a Laplacian distribution. The only parameter that we need to know about the data is its variance σ^2 (with λ given by $\lambda = \sqrt{2/\sigma^2}$).

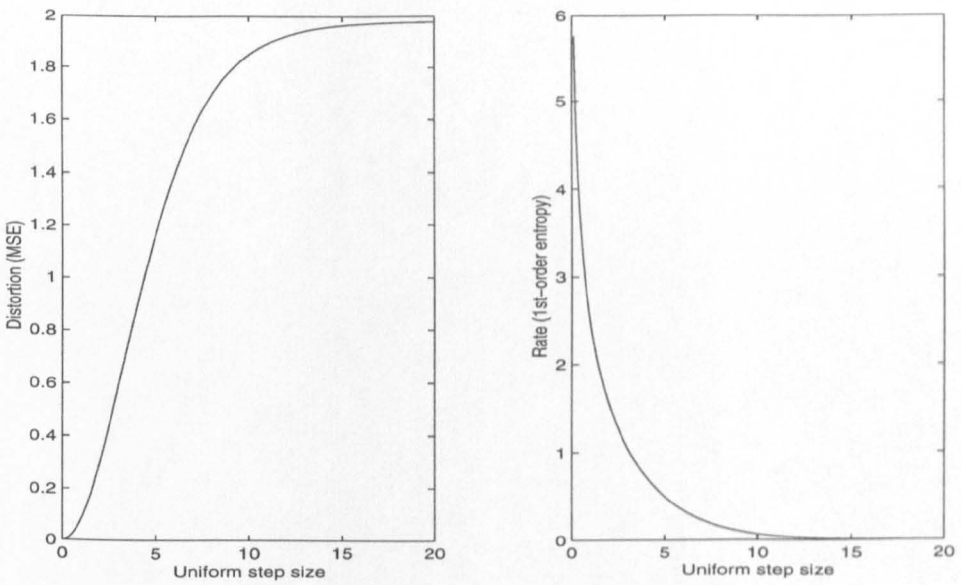


Figure A.1: Typical plots of $D(\Delta)$ and $R(\Delta)$ versus Δ

Appendix B

Comparison of the Cost Functions

1. $\log x$ *vs.* $-x \log x$: Let $\Phi_1(x) = \log x$ and $\Phi_2(x) = -x \log x$. Then $\Phi_1(1) = 0$, $N\Phi_1(1/N) = -N \log N$ and $\Phi_2(1) = 0$, $N\Phi_2(1/N) = \log N$. Clearly

$$\Phi_1(1) < N\Phi_2(1/N) \tag{B.1}$$

and

$$\Phi_2(1) > N\Phi_1(1/N) \tag{B.2}$$

for $N > 1$.

2. $\log x$ *vs.* $x^{p/2}$: Let $\Phi_1(x) = \log x$ and $\Phi_2(x) = x^{p/2}$. Then $\Phi_2(1) = 1$, $N\Phi_2(1/N) = N^{1-\frac{p}{2}}$. Clearly

$$\Phi_1(1) < N\Phi_2(1/N) \tag{B.3}$$

and

$$\Phi_2(1) > N\Phi_1(1/N) \tag{B.4}$$

for $N > 1, p < 2$.

3. $\log x$ *vs.* $\sqrt{x}$: Let $\Phi_1(x) = \log x$ and $\Phi_2(x) = \sqrt{x}$. Then $\Phi_2(1) = 1$, $N\Phi_2(1/N) = \sqrt{N}$. Clearly

$$\Phi_1(1) < N\Phi_2(1/N) \tag{B.5}$$

and

$$\Phi_2(1) > N\Phi_1(1/N) \tag{B.6}$$

for $N > 1$.

Appendix C

Paper Presented at IEEE DCC'99

The Effect of Flexible Parsing for Dynamic Dictionary Based Data Compression *

Yossi Matias† Nasir Rajpoot‡ Süleyman Cenk Şahinalp§

Abstract

We report on the performance evaluation of greedy parsing with a single step lookahead, denoted as flexible parsing. We also introduce a new fingerprint based data structure which enables efficient, linear time implementation.

1 Introduction

The most common compression algorithms are based on maintaining a dynamic *dictionary* of strings that are called *phrases*, and replacing substrings of an input text with pointers to identical phrases in the dictionary. Dictionary based compression algorithms of particular interest are the LZ78 method [ZL78], its LZW variant [Wel84], and the LZ77 method [ZL77] which are all asymptotically optimal for a wide range of sources.

Given a dictionary construction scheme, there is more than one way to *parse* the input, i.e., choose which substrings in the input text will be replaced by respective *codewords*. Almost all dynamic dictionary based algorithms in the literature use

*Part of this work was presented in Workshop on Algorithmic Engineering, Saarbrucken, Germany, 1998.

†Computer Science Dept., Tel-Aviv University; matias@math.tau.ac.il. Research supported in part by an Alon Fellowship, by the Israel Science Foundation founded by The Academy of Sciences and Humanities, and by the Israeli Ministry of Science.

‡Mathematics Dept., Yale University; Computer Science Dept., Warwick University; rajpoot@noodle.med.yale.edu.

§Computer Science Dept., Warwick University; Center for BioInformatics, Univ. of Pennsylvania; cenk@dcs.warwick.ac.uk. Research supported by NATO research grant CRG-972175 and ESPRIT LTR Project no. 20244 – ALCOM IT.

greedy parsing which is fast and can be applied on-line. However, it usually results in far from optimal parsing/compression: for the LZW dictionary method, there are strings T which can be (optimally) parsed to some m phrases, for which the greedy parsing obtains $\Omega(m^{3/2})$ phrases ([MS99]; a similar result for static dictionaries is in [GSS85]).

In [Hor95] it was demonstrated that the compression achieved by LZW algorithm on some standard benchmark files can be improved by looking ahead a few steps. However it was noted that: "An optimal parsing scheme would also have to consider the possibility of matching a short first string and then a short second string in order to match a very long third string. We will however reject such possibilities as being too expensive to implement (for minimal expected gain in compression). Our non-greedy version of LZW will only look ahead by one parsed string when choosing its course of action." In fact the algorithm proposed in [Hor95] not only changes the parsing scheme but also constructs an entirely different dictionary from that of LZW on a given string – hence compression improvement over LZW is not guaranteed, and the notion of optimality is not clear. The worst case running time of this algorithm is $O(|T^{3/2}|)$, where $|T|$ is the input size.

An interesting fact for static dictionaries is that greedy parsing is optimal for those dictionaries with the suffix property [CK96]. It follows that if all the input is available off-line, it can be parsed optimally via a rigt-to-left greedy parsing provided that the dictionary is static and has the prefix property.

Recently [MS99] demonstrated that *for all dictionary construction schemes with the prefix property, greedy parsing with a single step lookahead is optimal on all input strings* – this scheme is called *flexible parsing* or $\mathcal{FP}$. A new data structure which implements the algorithm that uses LZW dictionary construction with $\mathcal{FP}$ in $O(|T|)$ time and space proportional to the number of phrases in the dictionary is introduced as well. The space and time complexity of this data structure is comparable to that of the original LZW implementation, hence optimal compression can be achieved without any overhead in complexity. Note that suffix trees can also be used for this application [RPE81]. However the $O(|T|))$ space complexity of the suffix tree is expected to be much larger than the space complexity of the new data structure which is proportional to the number of output phrases.

In this study, we report an experimental evaluation of $\mathcal{FP}$ in the context of LZW dictionary construction scheme:

(1) We demonstrate that optimality of $\mathcal{FP}$ in the context of the LZW dictionary construction, denoted as LZW-$\mathcal{FP}$, translates into considerable improvement over greedy parsing in practice. We also consider the algorithm of [Hor95], which uses flexible dictionary construction, denoted here as FPA. The LZW-$\mathcal{FP}$ and FPA al-

gorithms are compared with UNIX `compress` (LZW) and `gzip` (LZ77). On the tested data files, both LZW-$\mathcal{FP}$ and FPA perform better, up to 20% improved compression, than UNIX `compress`. They are both inferior to `gzip` on small to moderate text files, such as in the Calgary corpus, but are typically superior to gzip for files larger than 1MB, and for non-textual data files of all sizes. For pseudo-random strings and DNA sequences, the improvement is up to 35%.
(2) We introduce a new data structure based on Karp-Rabin fingerprints [KR87] to efficiently implement $\mathcal{FP}$. Currently our algorithms run about $3-5$ times slower than `compress` which is the fastest among all algorithms, both during compression and decompression. We are in the process of improving our implementations and hence leave reporting on explicit timing results to the full paper.
(3) We investigate whether better asymptotic properties of LZ78 based algorithms in comparison to LZ77 translate into improved compression. We demonstrate that on pseudorandom bit streams (with various distributions) the redundancy in the output of each of the four programs approach to the expected asymptotic behavior very fast – requiring less than 1KB for each of the different distributions; better asymptotic properties of LZW in comparison to LZ77 is very visible.[1] For files of size $1MB$, `compress` can improve over `gzip` up to 20% in compression achieved [2].

2 The Compression Algorithms

In this section we describe how each of the algorithms we consider work. We give the descriptions of the standard algorithms as well as new ones for the sake of completeness. Each of the algorithms fit in a general framework that we describe below.

Model. We denote a compression algorithm by $\mathcal{C}$, and its corresponding decompression algorithm by $\mathcal{C}^{\leftarrow}$. The input to $\mathcal{C}$ is a string T, of n characters, chosen from a constant size alphabet Σ; in our experiments Σ is either ascii or is $\{0,1\}$. We denote by $T[i]$, the i^{th} character of T ($1 \leq i \leq n$), and by $T[i:j]$ the substring which begins at $T[i]$ and ends at $T[j]$; notice that $T = T[1:n]$.

The compression algorithm $\mathcal{C}$ compresses the input by reading the input charac-

[1] The average number of bits output by LZ78 or LZW, for the first n characters of an input string created by an i.i.d. source is only $O(1/\log n)$ more than its entropy [JS95, LS95]. A similar result for more general, unifilar, sources has been obtained by Savari [Sav97]. For the LZ77 algorithm, this redundancy is as much as $O(\log\log n/\log n)$ [Wyn95].

[2] All the software, documentation, and detailed experimental results reported in this paper are available on the WWW [Sou].

ters from left to right (i.e. from $T[1]$ to $T[n]$) and by partitioning it into substrings which are called blocks. Each block is replaced by a corresponding label that we call a codeword. We denote the j^{th} block by $T[b_j : b_{j+1} - 1]$, or shortly T_j, where $b_1 = 1$. The output of $\mathcal{C}$, hence, consists of codewords $C[1], C[2], \ldots, C[k]$ for some k, which are the codewords of blocks $T_1, T_2, \ldots, T_k$ respectively.

The algorithm $\mathcal{C}$ maintains a dynamic set of substrings called the dictionary, $\mathcal{D}$. Initially, $\mathcal{D}$ consists of all one-character substrings possible. The codewords of such substrings are their characters themselves. As the input T is read, $\mathcal{C}$ adds some of its substrings to $\mathcal{D}$ and assigns them unique codewords. We call such substrings of T phrases of $\mathcal{D}$. Each block T_j is identical to a phrase in $\mathcal{D}$: hence $\mathcal{C}$ achieves compression by replacing substrings of T with pointers to their earlier occurrences in T.

The decompression algorithm $\mathcal{C}^{\leftarrow}$ that corresponds to $\mathcal{C}$, takes $C[1 : k]$ as input and computes $T[1 : n]$ by replacing each $C[j]$ by its corresponding block T_j. Because the codeword $C[j]$ is a function of $T[1 : b_j - 1]$ only, the decompression can be correctly performed in an inductive fashion.

Below, we provide detailed descriptions of the compression algorithms considered, both the new and the old for the sake of completeness .

LZ-77 Algorithm. The LZ-77 algorithm reads the input characters from left to right while inserting all its substrings in $\mathcal{D}$. In other words, at the instance it reads $T[i]$, all possible substrings of the form $T[j : \ell]$, $j \leq \ell < i$ are in $\mathcal{D}$, together with all substrings of size one. The codeword of the substring $T[j : \ell]$, is the 2-tuple, $(i - j, \ell - j + 1)$, where the first entry denotes the relative location of $T[j : \ell]$, and the second entry denotes its size. LZ77 uses greedy parsing: the m^{th} block $T_m = T[b_m : b_{m+1} - 1]$ is recursively defined as the longest substring which is in $\mathcal{D}$ just before $\mathcal{C}$ reads $T[b_{m+1} - 1]$.

LZW Algorithm. The LZW algorithm reads the input characters from left to right while inserting in $\mathcal{D}$ all substrings of the form $T[b_m : b_{m+1}]$. Hence the phrases of LZW are the substrings obtained by concatenating the blocks of T with the next character following them, together with all possible substrings of size one. The codeword of the phrase $T[b_m : b_{m+1}]$ is the integer $|\Sigma| + m$, where $|\Sigma|$ is the size of the alphabet Σ. Thus, the codewords of substrings do not change in LZW algorithm. LZW uses greedy parsing as well: the m^{th} block T_m is recursively defined as the longest substring which is in $\mathcal{D}$ just before $\mathcal{C}$ reads $T[b_{m+1} - 1]$. Hence, no two phrases can be identical in the LZW algorithm.

LZW-$\mathcal{FP}$ Algorithm. The LZW-$\mathcal{FP}$ algorithm reads the input characters from left to right while inserting in $\mathcal{D}$ all substrings of the form $T[b'_m : b'_{m+1}]$, where b'_m denotes the beginning location of block m if the compression algorithm used were

LZW. Hence for dictionary construction purposes LZW-$\mathcal{FP}$ emulates LZW: for any input string LZW and LZW-$\mathcal{FP}$ build identical dictionaries. The output generated by these two algorithms however are quite different. The codeword of the phrase $T[b'_m : b'_{m+1}]$ is the integer $|\Sigma| + m$, where $|\Sigma|$ is the size of the alphabet Σ. LZW-$\mathcal{FP}$ uses flexible parsing: intuitively, the m^{th} block T_m is recursively defined as the substring which results in the longest advancement in the next iteration. More precisely, let the function f be defined on the characters of T such that $f(i) = \ell$ where $T[i : \ell]$ is the longest substring starting at $T[i]$, which is in $\mathcal{D}$ just before C reads $T[\ell]$. Then, given b_m, the integer b_{m+1} is recursively defined as the integer α for which $f(\alpha)$ is the maximum among all α such that $T[b_m : \alpha - 1]$ is in $\mathcal{D}$ just before C reads $T[\alpha - 1]$.

FPA Algorithm. The FPA algorithm reads the input characters from left to right while inserting in $\mathcal{D}$ all substrings of the form $T[b_m : f(b_m)]$, where the function f is as described in LZW-$\mathcal{FP}$ algorithm. Hence for almost all input strings, FPA constructs an entirely different dictionary with that of LZW-$\mathcal{FP}$. The codeword of the phrase $T[b_m : f(b_m)]$ is the integer $|\Sigma| + m$, where $|\Sigma|$ is the size of the alphabet Σ. FPA again uses flexible parsing: given b_m, the integer b_{m+1} is recursively defined as the integer α for which $f(\alpha)$ is the maximum among all α such that $T[b_m : \alpha - 1]$ is in $\mathcal{D}$.

3 Data Structures and Implementations

In this section we describe both the trie-reverse-trie data structure, and the new fingerprints based data structure for efficient on-line implementations of the LZW-$\mathcal{FP}$, and FPA methods. The trie-reverse-trie pair is a deterministic data structure, and hence guarantees a worst case linear running time for both algorithms as described in [MS99]). The new data structure based on *fingerprints* [KR87], is randomized, and guarantees an expected linear running time for the algorithm.

The two main operations to be supported by these data structures are (1) insert a phrase to $\mathcal{D}$ (2) search for a phrase, i.e., given a substring S, check whether it is in $\mathcal{D}$. The standard data structure used in many compression algorithms including LZW, the compressed trie $\mathcal{T}$ supports both operations in time proportional to $|S|$. A compressed trie is a rooted tree with the following properties: (1) each node with the exception of the root represents a dictionary phrase; (2) each edge is labeled with a substring of characters; (3) the first characters of two sibling edges can not be identical; (4) the concatenation of the substrings of the edges from the root to a given node is the dictionary phrase represented by that node; (5) each node is

labeled by the codeword corresponding to its phrase. Dictionaries with prefix properties, such as the ones used in LZW and LZ78 algorithms, build a regular trie rather than a compressed one. The only difference is that in a regular trie the substrings of all edges are one character long.

In our data structures, inserting a phrase S to $\mathcal{D}$ takes $O(|S|)$ time as in the case of a trie. Similarly, searching S takes $O(|S|)$ time if no information about substring S is provided. However, once it is known that S is in $\mathcal{D}$, searching strings obtained by concatenating or deleting characters to/from both ends of S takes only $O(1)$ time. More precisely, our data structures support two operations *extend* and *contract* in $O(1)$ time. Given a phrase S in $\mathcal{D}$, the operation extend(S, a) for a given character a, finds out whether the concatenation of S and a is a phrase in $\mathcal{D}$. Similarly, the operation contract(S), finds out whether the suffix $S[2 : |S|]$ is in $\mathcal{D}$. Notice that such operations can be performed in a suffix tree, if the phrases in $\mathcal{D}$ are all the suffixes of a given string as in the case of the LZ77 algorithm [RPE81]. For arbitrary dictionaries (such as the ones built by LZW) our data structures are unique in supporting contract and extend operations in $O(1)$ time, and insertion operation in time linear with the size of the phrase, while using $O(|\mathcal{D}|)$ space, where $|\mathcal{D}|$ is the number of phrases in $\mathcal{D}$.

Trie-reverse-trie-pair data structure. Our first data structure builds the trie, $\mathcal{T}$, of phrases as described above. In addition to $\mathcal{T}$, it also constructs $\mathcal{T}^r$, the compressed trie of the *reverses* of all phrases inserted in the $\mathcal{T}$. Given a string $S = s_1, s_2, \ldots, s_n$, its reverse S^r is the string $s_n, s_{n-1}, \ldots, s_2, s_1$. Therefore for each node v in $\mathcal{T}$, there is a corresponding node v^r in $\mathcal{T}^r$ which represents the reverse of the phrase represented by v. As in the case of the $\mathcal{T}$ alone, the insertion of a phrase S to this data structure takes $O(|S|)$ time. Given a dictionary phrase S, and the node n which represents S in $\mathcal{T}$, one can find out whether the substring obtained by concatenating S with any character a in is $\mathcal{D}$, by checking out if there is an edge from n with corresponding character a; hence extend operation takes $O(1)$ time. Similarly the contract operation takes $O(1)$ time by going from n to n', the node representing reverse of S in $\mathcal{T}^r$, and checking if the parent of n' represents $S[2 : |S|]^r$.

Fingerprints based data structure. Our second data structure is based on building a hash table H of size p, a suitably large prime number. Given a phrase $S = S[1 : |S|]$, its location in H is computed by the function h, where $h(S) = (s[1]|\Sigma|^{|S|} + s[2]|\Sigma|^{|S|-1} + \ldots + s[|S|])\ mod\ p$, where $s[i]$ denotes the lexicographic order of $S[i]$ in Σ [KR87]. Clearly, once the values of $|\Sigma|^k\ mod\ p$ are calculated for all k up to the maximum phrase size, computation of $h(S)$, takes $O(|S|)$ time. By taking p sufficiently large, one can decrease the probability of a collision on a hash value to some arbitrarily small $1/\epsilon$ value; thus the average running time of an insertion would be

$O(|S|)$ as well. Given the hash value $h(S)$ of a string, the hash value of its extension by any character a can be calculated by $h(Sa) = (h(S)|\Sigma| + lex(a)) \bmod p$, where $lex(a)$ is the lexicographic order of a in Σ. Similarly, the hash value of its suffix $S[2 : |S|]$ can be calculated by $h(S[2 : |S|]) = (h(S) - s[1]|\Sigma|^{|S|}) \bmod p$. Both operations take $O(1)$ time.

In order to verify if the hash table entry $h(S)$ includes S in $O(1)$ time we (1) give unique labels to each of the phrases in $\mathcal{D}$, and (2) in each phrase S in H, store the label of the suffix $S[2 : |S|]$ and the label of the prefix $S[1 : |S| - 1]$. The label of newly inserted phrase can be $|\mathcal{D}|$, the size of the dictionary. This enables both extend and contract operations to be performed in $O(1)$ time on the average: suppose the hash value of a given string S is h, and the label of S is ℓ. To extend S with character a, we first compute the hash value h' of the string Sa. Among the phrases whose hash value is h', the one whose prefix label matches the label of S gives the result of the extend operation. To contract S, we first compute the hash value h'' of the string $S[2 : |S|]$. Among the phrases whose hash value is h'', the one whose label matches the suffix label of S gives the result of the contract operation. Therefore, both extend and contract operations take expected $O(1)$ time.

Inserting a phrase in this data structure can be performed as follows. An insert operation is done only after an extend operation on some phrase S (which is in $\mathcal{D}$) with some character a. Hence, when inserting the phrase Sa in $\mathcal{D}$ its prefix label is already known: the label of S. Once it is decided that Sa is going to be inserted, we can spend $O(|S| + 1)$ time to compute the suffix label of Sa. In case the suffix $S[2 : |S|]a$ is not a phrase in $\mathcal{D}$, we temporarily insert an entry for $S[2 : |S|]a$ in the hash table. This entry is then filled up when $S[2 : |S|]$ is actually inserted in $\mathcal{D}$. Clearly, the insertion operation for a phrase R takes expected $O(|R|)$ time.

A linear time implementation of LZW-$\mathcal{FP}$. For any input T LZW-$\mathcal{FP}$ inserts to $\mathcal{D}$ the same phrases with LZW. The running time for insertion in both LZW and LZW-$\mathcal{FP}$ (via the data structures described above) are the same; hence the total time needed to insert all phrases in LZW-$\mathcal{FP}$ should be identical to that of LZW, which is linear with the input size. Parsing with $\mathcal{FP}$ consists of a series of extend and contract operations. We remind that: (1) the function f on characters of T is described as $f(i) = \ell$ where $T[i : \ell]$ is the longest substring starting at $T[i]$, which is in $\mathcal{D}$. (2) given b_m, the integer b_{m+1} is recursively defined as the integer α for which $f(\alpha)$ is the maximum among all α such that $T[b_m : \alpha - 1]$ is in $\mathcal{D}$. In order to compute b_{m+1}, we inductively assume that $f(b_m)$ is already computed. Clearly $S = T[b_m : f(b_m)]$ is in $\mathcal{D}$ and $S' = T[b_m : f(b_m)]$ is not in $\mathcal{D}$. We then contract S by i characters, until $S' = T[b_m + i : f(b_m) + 1]$ is in $\mathcal{D}$. Then we proceed with extensions to compute $f(b_m + i)$. After subsequent contract and extends we stop

once $i > f(b_m)$. The last value of i at which we started our final round of contracts is the value b_{m+1}. Notice that each character in T participates to exactly one extend and one contract operation, each of which takes $O(1)$ time via the data structures described above. Hence the total running time for the algorithm is $O(n)$.

4 Experiments

In this section we describe in detail the data sets we used, and discuss our test results verifying how well our theoretical expectations were supported.

The test programs. We used `gzip`, `compress`, LZW-$\mathcal{FP}$ and FPA programs for our experiments. In our LZW-$\mathcal{FP}$ implementation we limited the dictionary size to 2^{16} phrases, and reset it when it was full as in the case of `compress`; we also experimented with the extended version of LZW-$\mathcal{FP}$ which allows 2^{24} phrases. Similarly we experimented with two versions of FPA: one with 2^{16} and the other with 2^{24} phrases maximum.

The data sets. Our data sets come from three sources: (1) Data obtained via UNIX `drand48()` pseudorandom number generator - designed to measure the asymptotic redundancy in algorithms. (2) DNA and protein sequences provided by Center for BioInformatics, University of Pennsylvania and CT and MR scans provided by the St. Thomas Hospital, UK [Sou]. (3) Text files from two data compression benchmark suites: the new Canterbury corpus and the commonly used Calgary corpus [Sou].

Specifically, the first data set includes three binary files generated by the UNIX `drand48()` function. The data distribution is i.i.d. with bit probabilities (1) 0.7 – 0.3, (2) 0.9 – 0.1, and (3) 0.97 – 0.03. The second data set includes two sets of human DNA sequences from chromosome 23 (*dna1, dna2*), one MR (magnetic resonance) image of human (female) breast (*mr.pgm*), and one CT (computerized tomography) scan of a fractured human hip *ct.pgm* in uncompressed `pgm` format in ASCII [Sou]. The third set includes the complete Calgary corpus; the corresponding table is ommited here for lack of space, and can be found at [Sou]. It also includes all files of size $> 1MB$ from the new Canterbury corpus: a DNA sequence from E-coli bacteria, *E.coli*, the complete bible *bib.txt*, and *world192.txt*.

Test results. In summary, we observed that LZW-$\mathcal{FP}$ and FPA implementations with maximum dictionary size 2^{24} performs the best on all types of files with size $> 1MB$ and shorter files with non-textual content. For shorter files consisting text, `gzip` performs the best as expected.

Our tests on the human DNA sequences with LZW-$\mathcal{FP}$ and FPA show similar

improvements over *compress* and *gzip* - with a dictionary of maximum size 2^{16}, the improvement is about 1.5% and 5.7% respectively. Some more impressive results were obtained by increasing the dictionary size to 2^{24}, which further improved the compression ratio to 9%. The performance of LZW-$\mathcal{FP}$ and FPA on *mr* and *ct* scans differ quite a bit: LZW-$\mathcal{FP}$ was about $4\% - 6\%$ better than `compress` and was comparable to `gzip`; FPA's improvement was about 15% and 7% respectively. As the image files were rather short, we didn't observe any improvement by using a larger dictionary. One interesting observation is that the percentage improvement achieved by both FPA and LZW-$\mathcal{FP}$ increased consistently with increasing data size. This suggests that we can expect them to perform better in compressing massive archives as needed in many biomedical applications such as the human genome project.

Our results on text strings varied depending on the type and size of the file compressed. For short files with long repetitions, `gzip` is still the champion. However, for all text files of size $> 1MB$, the large dictionary implementation of FPA scheme outperforms `gzip` by $4.7\% - 8.5\%$, similar to the tests for DNA sequences. The following tables demonstrate the relative performances of the test programs on the data sets: Column 1 shows original file size (with some prefixes), column 2 and column 3 show the compressed file size by *gzip* and *compress* respectively, and the remaining columns show the improvement (%) made by *LZW-FP*, *FPA*, *FP-24*, and *FPA-24* over *gzip* and *compress*.

File	*Size*	*gzip*	*comprs*	*LZW-FP*		*FPA*		*FP-24*		*FPA-24*	
	(KB)	(KB)	(KB)	$\uparrow_g$ (%)	$\uparrow_c$ (%)	$\uparrow_g$ (%)	$\uparrow_c$ (%)	$\uparrow_g$ (%)	$\uparrow_c$ (%)	$\uparrow_g$ (%)	$\uparrow_c$ (%)
E.coli	4530	1341	1255	6.91	0.56	6.43	0.05	8.84	2.63	8.48	2.24
bible.txt	3953	1191	1401	-12.87	4.11	-7.79	8.42	0.13	15.15	4.68	19.01
world192.txt	2415	724	987	-31.70	3.32	-20.36	11.64	-2.38	24.84	6.54	31.39

Table 1: Compression evaluation using files in the Canterbury corpus (Large Set)

References

[CK96] M. Cohn and R. Khazan. Parsing with suffix and prefix dictionaries. In *IEEE Data Compression Conference*, 1996.

[GSS85] M. E. Gonzales-Smith and J. A. Storer. Parallel algorithms for data compression. *Journal of the ACM*, 32(2):344–373, April 1985.

[Hor95] R. N. Horspool. The effect of non-greedy parsing in Ziv-Lempel compression methods. In *IEEE Data Compression Conference*, 1995.

File	Size (KB)	gzip (KB)	comprs (KB)	LZW-FP		FPA		FP-24		FPA-24	
				$\uparrow_g$ (%)	$\uparrow_c$ (%)	$\uparrow_g$ (%)	$\uparrow_c$ (%)	$\uparrow_g$ (%)	$\uparrow_c$ (%)	$\uparrow_g$ (%)	$\uparrow_c$ (%)
P(0)=0.7	1	.2	.2	4.88	6.25	4.88	6.25	4.88	6.25	4.88	6.25
	100	15.7	13.4	15.60	1.62	17.89	4.29	15.61	1.64	17.89	4.29
	2048	**320**	**263**	**18.53**	**1.07**	**20.46**	**3.41**	**19.44**	**2.17**	**21.20**	**4.31**
P(0)=0.9	1	.1	.1	3.10	6.72	9.30	12.69	3.10	6.72	9.30	12.69
	100	9.7	7.75	22.73	2.89	25.86	6.83	22.74	2.90	25.86	6.83
	2048	**200**	**147**	**28.10**	**2.07**	**30.95**	**5.95**	**28.43**	**2.50**	**31.20**	**6.28**
P(0)=0.97	1	.09	.1	6.45	12.12	6.45	12.12	6.45	12.12	6.45	12.12
	100	4.6	3.8	22.39	4.42	28.03	11.37	22.41	4.45	28.03	11.37
	2048	**91.4**	**64.8**	**31.30**	**3.10**	**35.79**	**9.44**	**31.30**	**3.11**	**35.79**	**9.44**

Table 2: Compression evaluation using independent identically distributed random files containing only zeros and ones with different probability distributions

File	Size (KB)	gzip (KB)	comprs (KB)	LZW-FP		FPA		FP-24		FPA-24	
				$\uparrow_g$ (%)	$\uparrow_c$ (%)	$\uparrow_g$ (%)	$\uparrow_c$ (%)	$\uparrow_g$ (%)	$\uparrow_c$ (%)	$\uparrow_g$ (%)	$\uparrow_c$ (%)
dna1	3096	977	938	5.59	1.54	5.75	1.70	8.73	4.82	8.91	5.00
dna2	2877	846	813	4.64	0.75	4.33	0.43	6.09	2.26	5.89	2.05
mr.pgm	260	26	29	-7.23	3.60	6.38	15.84	-7.22	3.61	6.38	15.84
ct.pgm	1039	110	110	4.10	3.61	14.56	14.12	4.10	3.61	14.56	14.12

Table 3: Compression evaluation using experimental biological and medical data

[JS95] P. Jacquet and W. Szpankowski. Asymptotic behavior of the Lempel-Ziv parsing scheme and digital search trees. *Theoretical Computer Science*, (144):161–197, 1995.

[KR87] R. Karp and M. O. Rabin. Efficient randomized pattern matching algorithms. *IBM Journal of Research and Development*, 31(2):249–260, 1987.

[LS95] G. Louchard and W. Szpankowski. Average profile and limiting distribution for a phrase size in the Lempel-Ziv parsing algorithm. *IEEE Transactions on Information Theory*, 41(2):478–488, March 1995.

[MS99] Y. Matias and S. C. Sahinalp. On optimality of parsing in dynamic dictionary based data compression. ACM-SIAM Symposium on Discrete Algorithms, 1999.

[RPE81] M. Rodeh, V. Pratt, and S. Even. Linear algorithm for data compression via string matching. *Journal of the ACM*, 28(1):16–24, January 1981.

[Sav97] S. Savari. Redundancy of the Lempel-Ziv incremental parsing rule. In *IEEE Data Compression Conference*, 1997.

[Sou] http://www.dcs.warwick.ac.uk/people/research/Nasir.Rajpoot/work/fp/index.html.

[Wel84] T.A. Welch. A technique for high-performance data compression. *IEEE Computer*, pages 8–19, January 1984.

[Wyn95] A. J. Wyner. *String Matching Theorems and Applications to Data Compression and Statistics*. Ph.D. dissertation, Stanford University, Stanford, CA, 1995.

[ZL77] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, IT-23(3):337–343, May 1977.

[ZL78] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, IT-24(5):530–536, September 1978.

Appendix D

Paper Presented at IEEE ICIP'99

On Zerotree Quantization for Embedded Wavelet Packet Image Coding

N. M. Rajpoot [1,2], F. G. Meyer [3], R. G. Wilson [1], and R. R. Coifman [2]
[1]Department of Computer Science, University of Warwick, Coventry, United Kingdom
Department of [2]Mathematics and [3]Computer Science, Yale University, New Haven, CT
e-mail: rajpoot-nasir@cs.yale.edu

Abstract

Wavelet packets are an effective representation tool for adaptive waveform analysis of a given signal. We first combine the wavelet packet representation with zerotree quantization for image coding. A general zerotree structure is defined which can adapt itself to any arbitrary wavelet packet basis. We then describe an efficient coding algorithm based on this structure. Finally, the hypothesis for prediction of coefficients from coarser scale to finer scale is tested and its effectiveness is compared with that of zerotree hypothesis for wavelet coefficients.

1 Introduction

There has been a surge of interest in wavelet transforms for image and video coding applications, in recent years. This is mainly due to the nice localization properties of wavelets in both time (or space) and frequency. The zerotree quantization, proposed by Shapiro [8], is an effective way of exploiting the self-similarities among high-frequency subbands at various resolutions. The main thrust of this quantization strategy is in the prediction of corresponding wavelet coefficients in higher frequency subbands at the finer scales, by exploiting the parent-offspring dependencies. This prediction works well, in terms of efficiently coding the wavelet coefficients, due to the statistical characteristics of subbands at various resolutions, and due to the scale-invariance of edges in high frequency subbands of similar orientation. Various extensions of zerotree quantization, such as [3, 9], have been proposed ever since its introduction.

Wavelet packets [1] were developed in order to adapt the underlying wavelet bases to the contents of a signal. The basic idea is to allow non-octave subband decomposition to adaptively select the *best basis* for a particular signal. Results from various image coding schemes based on wavelet packets show that they are particularly good in coding images with oscillatory patterns, a special form of *texture*. While wavelet packet bases are well adapted to the corresponding signal (image), one loses the parent-offspring dependencies, or the *spatial orientation trees*, as defined in [8, 7].

In this work, we address the following question: can the zerotree quantization be applied to wavelet packet transformed images? The question is valid since one does not generally observe as much self-similarities among the wavelet packet subbands as among the wavelet subbands. Moreover, even if there does exist self-similarity among wavelet packet subbands, the question arises: *how would the spatial orientation trees, or zerotrees, be defined in order to predict the insignificance of corresponding wavelet packets at a finer scale, given a wavelet packet at a coarser scale*? The issue was addressed partially by Xiong et al. in [10]. In their work, however, the tree decomposition was restricted not to have a basis causing the *parenting conflict*, a problem described in the next section. We present a solution to this problem and define a set of rules to construct the zerotree structure for a given wavelet packet geometry, thus offering a general structure for an arbitrary wavelet packet decomposition. This generalized zerotree structure is termed as the *compatible zerotree* for reasons mentioned in the next section. The new tree structure provides an efficient way of encoding the wavelet packet coefficients. We combine the idea of zerotree quantization with wavelet packet transforms for image coding purposes.

2 Compatible Zerotree Quantization

The wavelet packet basis [1] is adaptively selected in order to tailor the representation to the contents of a signal (an image, in our case). The nodes in a full subband tree (or short-term Fourier transform, namely STFT, tree) are pruned following a series of split/merge decisions using certain criterion (see [2] for entropy-based best basis selection and [6] for optimizing the best basis from a rate-distortion viewpoint). Suppose the best basis has been selected us-

ing one of these methods. The issue is how to organize the spatial orientation trees so as to exploit the self-similarities, if any, among the subbands. The basis selected by any of the above methods does not, in general, yield the parent-offspring relationships like those in wavelet subbands. Moreover, there can be an instance in the wavelet packet tree where one or more of the child nodes are at a coarser scale than the parent node. This results in the association of each coefficient of such a child node to multiple parent coefficients, in the parent node, giving rise to a *parenting conflict*. To make the point clear, let us consider the segmentation shown in Figure 1, for a 3-level wavelet packet transform. The nodes C_1, C_2, C_3, and C_4 at a coarser scale are children of the node P at a finer scale. The problem with such a situation is that there are four candidates in P claiming the parenthood of the set of four coefficients belonging to the four children nodes. In [10], the best basis was selected in such a way that whenever a conflict like this arises, the four children are merged so as to resolve the conflict. This suboptimal approach constrains selection of the best basis at the loss of the freedom of adapting the wavelet packet basis to the contents of a particular signal.

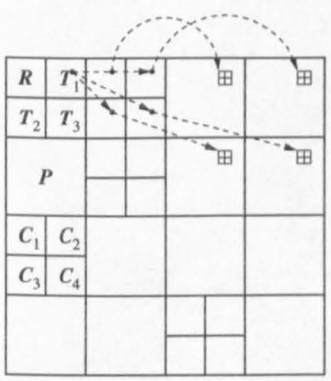


Figure 1: Sample segmentation in a 3-level wavelet packet decomposition and the parent-offspring dependencies for compatible zerotree originating from T_1.

We organize the zerotree structure, called *compatible zerotrees*, dynamically so that there is no restriction on selection of the wavelet packet basis (we call these the compatible zerotrees, since they are generated taking into account both the scale and orientation compatibility). It is to be noted that the compatible zerotrees differ from the spatial orientation trees of [7] in the sense that nodes of a compatible zerotree represent a full subband, rather than one or more coefficients. We utilize the compatible zerotrees to find the coefficients which are zerotree roots, for a given threshold. A set of rules is defined to construct the compatible zerotrees so that the overall zerotree structure can be constructed for any arbitrary wavelet packet basis. The only *assumption* made is that the basis has its lowest frequency bands at the coarsest scale. This is based upon the fact that a significant amount of the signal energy is concentrated in the lowest frequency subbands, which cannot be merged by any of the tree pruning algorithms.

Let R denotes the node representing the lowest frequency subband, situated in the top-left corner of a conventional subband decomposed image. Then R represents the root node of overall compatible zerotree. It has three children (T_1, T_2, and T_3 as shown in Figure 1) which represent the coarsest scale high-frequency subbands. These children are themselves root nodes of three compatible zerotrees with different global orientations; horizontal, vertical, and diagonal respectively. These zerotrees are generated in a recursive manner (only once for a given wavelet packet basis) using the following set of rules, each of them applied to nodes with similar global orientation.

a) If a node P is followed by four nodes C_1, C_2, C_3, and C_4 (at the same scale), then P is declared to be the parent of all these four nodes.

b) If four subbands P_1, P_2, P_3, and P_4 at a coarser scale are followed by four subbands C_1, C_2, C_3, and C_4 at immediately next finer scale, then node P_i is declared to be the parent of node C_i (for $i = 1, 2, 3, 4$).

c) If a node P at a coarser scale is followed only by a node C at the immediately next finer scale (like in a wavelet transform), the node P is declared as a parent of C.

d) If a node P is at a finer scale than four of its children, say C_1, C_2, C_3, and C_4, then P is disregarded as being the parent of all these nodes and all this bunch is moved up in the tree.

Let us consider the segmentation shown in Figure 1 to explain these rules. The construction of final compatible zerotrees proceeds in two steps. In the first step, the primary compatible zerotrees T_1, T_2, and T_3 are constructed using rules a, b, and c, as shown in Figure 2. In the second step, the overall tree is re-organized (as shown in Figure 3) using rule d, in order to resolve the parenting conflict under node P. This rule first identifies nodes with the parenting conflicts and when such nodes are found, the whole bunch (consisting of C_1, C_2, C_3, and C_4) is plucked from its current position in the tree. The algorithm then climbs up the

tree to look for an appropriate node, a compatible ancestor at the nearest scale and having the nearest orientation, and glues the bunch under this newly found compatible parent. The parent-offspring dependencies for the primary compatible zerotree T_1 are shown in Figure 1. For more details of the algorithm to construct the compatible zerotrees, please refer to [5].

Given an arbitrarily segmented wavelet packet basis, the compatible zerotrees are constructed using the set of rules described as above. In order to be able to make predictions for wavelet packet coefficients at finer scale subbands, the *compatible zerotree hypothesis* is defined as follows: If a wavelet packet coefficient of a node from the compatible zerotree is insignificant, it is more likely that the wavelet packet coefficients at similar spatial locations in all the descendent nodes of the same compatible zerotree will be insignificant as well.

We tested the success of compatible zerotree hypothesis, as defined above, for wavelet packets. The plots of wavelet packet coefficients' amplitude against the coefficient indices for the wavelet packet decomposition, using the entropy-based basis selection criterion as in [4], of a 512×512 Barbara image are shown in Figures 4 and 5. The amplitude axis in both plots was restricted to ± 1000 to facilitate the display of smaller coefficients. While the subbands were arranged in an increasing frequency order for the plot in Figure 4, the plot in Figure 5 refers to the subbands as organized in the compatible zerotrees. The three families of primary compatible zerotrees $T1$, $T2$, and $T3$ are clearly visible in the latter of these plots showing that the new arrangement was successful in isolating the coefficients of similar orientation in a hierarchy to be used for prediction in a top-down order.

3 The Coder Algorithm

Once the compatible zerotrees have been generated, based upon knowledge of the best basis, the encoding takes place by repeatedly running the *detection stage* and the *fine-tuning stage* until the bit budget is expired. Our algorithm requires only one list of detected coefficients (LDC) to be maintained, as opposed to two (three) lists kept by the EZW (SPIHT). The decoder first reads geometry of the best basis and generates the compatible zerotrees, and then proceeds by entropy decoding and interpreting the codewords (see [5] for more details of the algorithm).

4 Experiments and Conclusions

The idea of compatible zerotree quantization was coupled with the wavelet packet transform for progressive image coding and its performance was tested against the fast adaptive wavelet packet (FAWP) [4] image codec algorithm. A comparison of the peak signal to noise ratio (PSNR) achieved by CZQ and FAWP for three standard 512×512 images encoded at target bit rates of 0.25 and 0.10 bits per pixel (bpp) is presented in Table 1. While being relatively faster, by a factor ranging from two to four, than the bitplane coding of FAWP, the CZQ performs comparably well. We note that the effectiveness of zerotree quantization depends strongly upon the scale-invariance of edges among subbands of similar global orientation. The success of prediction of finer scale coefficients (set of all descendents $\mathcal{D}_i$ at similar spatial location) based upon a given coefficient i at a coarser scale is a function of the conditional probability of $\mathcal{D}_i$ being insignificant, given i is insignificant. We compared the number of zerotree root nodes detected, for various values of threshold, both for wavelet and wavelet packet bases. It was observed that the probability of detection of the zerotree root nodes in a wavelet decomposition is far greater than that in a wavelet packet decomposition.

It is important to note here that the wavelet packet basis was selected by simply using first-order entropy as a criterion for the split/merge decisions. The coefficients of basis were, therefore, imposed to be zerotree quantized. As opposed to wavelets, the wavelet packet basis might not necessarily produce coefficients that help towards success of the zerotree hypothesis (that is, produce more zerotrees). Figure 6 shows the geometry of wavelet packet basis selected for the 512×512 Barbara image. In fact, we found that using the entropy as a criterion for the best basis selection might result into many coarse scale high frequency subbands (whose coefficients represent oscillatory patterns in some parts of the image), as is clear from basis geometry in Figure 6. It is the encoding of coefficients in these types of subbands which takes considerable amount of the bit budget for the following reason. If a coefficient somewhere at the bottom of a compatible zerotree is found to be significant, the significance of all the corresponding coefficients in sibling and the parent nodes needs to be encoded as well, even if some or most of them are insignificant. Our future work aims to look at the methods of optimizing the selection of best basis using a criterion which results into optimal number of zerotrees and optimal distortion value, given the bit budget. Moreover, the ordering of wavelet packet coefficients with respect to increasing frequency (that is, decaying magnitude) within the compatible zerotrees will provide a gain in the coding performance, by increasing the efficiency of arithmetic coder to encode the zerotree codewords.

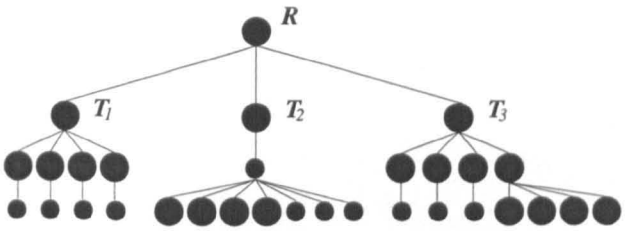


Figure 2: The overall compatible zerotree structure comprising of the three primary compatible zerotrees T_1, T_2, and T_3; each node represents a whole wavelet packet subband, and the node radius is directly proportional to the support of wavelet packets at that transform level.

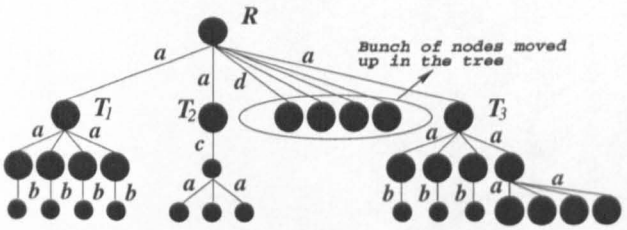


Figure 3: The overall compatible zerotree structure, after re-organization to resolve the parenting conflicts; the edge labels depict the rules used to generate the links between parent and the children nodes.

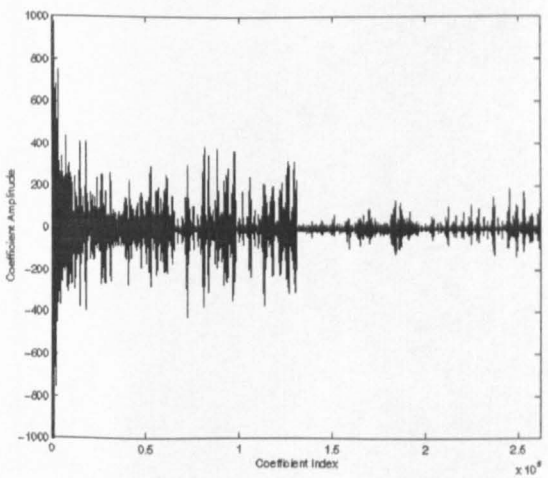


Figure 4: Plot of wavelet packet coefficients' amplitude versus coefficient indices for the wavelet packet decomposition of 512×512 Barbara image; the subbands were arranged in an increasing frequency order.

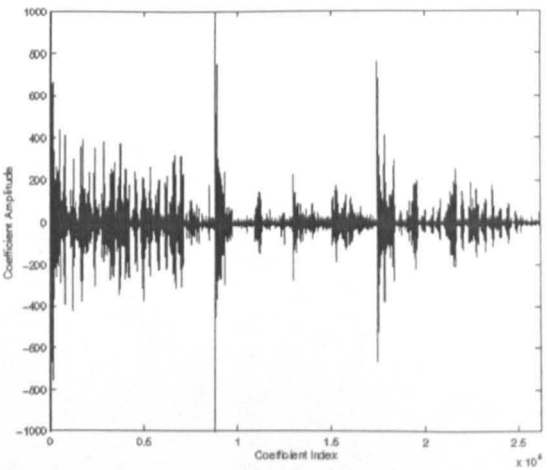


Figure 5: Plot of wavelet packet coefficients' amplitude versus coefficient indices for the wavelet packet decomposition of 512×512 Barbara image; the subbands were organized in compatible zerotrees with the help of rules mentioned in Section 2.

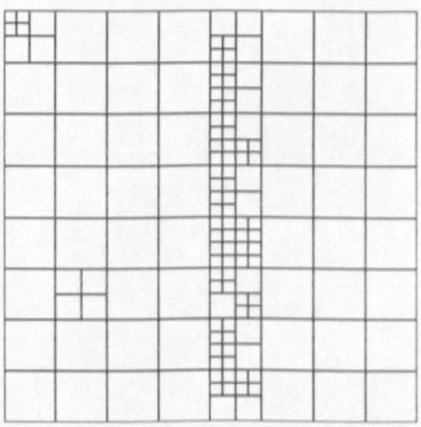


Figure 6: Wavelet packet basis selected for the 512×512 Barbara image using first-order entropy as selection criterion.

Image	Rate (bpp)	CZQ	FAWP
Fingerprint	0.25	26.68	27.21
	0.10	23.25	23.81
Goldhill	0.25	29.19	30.32
	0.10	26.66	27.54
Lena	0.25	31.88	33.26
	0.10	28.07	29.33

Table 1: Coding results for 512 × 512 standard images at 0.25 bpp and 0.10 bpp.

Acknowledgement

This work was supported in part by DARPA/AFOSR under Grant F49620-97-1-0011.

References

[1] R.R. Coifman and Y. Meyer, *Othornormal wave packet bases*, preprint, Dept. of Mathematics, Yale University, New Haven (1990).

[2] R.R. Coifman and M.V. Wickerhauser, *Entropy-based algorithms for best basis selection*, IEEE Trans. on Information Theory **38** (1992), no. 2, 713–718.

[3] M. Effros, *Zerotree design for image compression: Toward weighted universal zerotree coding*, IEEE Intl. Conf. on Image Processing, November 1997.

[4] F.G. Meyer, A.Z. Averbuch, J-O. Strömberg, and R.R. Coifman, *Fast wavelet packet image compression*, Data Compression Conference - DCC'98, 1998.

[5] N.M. Rajpoot, F.G. Meyer, and R.R. Coifman, *Wavelet packet image coding using compatible zerotree quantization*, Technical Report, Dept. of Computer Science, Yale University, New Haven (1999).

[6] K. Ramchandran and M. Vetterli, *Best wavelet packet bases in a rate-distortion sense*, IEEE Trans. on Image Processing (1993), 160–175.

[7] A. Said and W.A. Pearlman, *A new fast and efficient image codec based on set partioning in hierarchical trees*, IEEE Trans.on Circ.& Sys. for Video Tech. **6** (1996), 243–250.

[8] J.M. Shapiro, *Embedded image coding using zerotrees of wavelet coefficients*, IEEE Trans. on Signal Processing (1993), 3445–3462.

[9] Z. Xiong, O. Guleryuz, and M.T. Orchard, *Embedded image coding based on DCT*, VCIP'97, San Jose, CA, Feb. 1997.

[10] Z. Xiong, K. Ramchandran, and M.T. Orchard, *Wavelet packet image coding using space-frequency quantization*, IEEE Trans. on Image Processing **7, N. 6** (1998), 892–898.

Bibliography

[1] E.H. Adelson, E. Simoncelli, and R. Hingorani. Orthogonal pyramid transforms for image coding. In *Proceedings SPIE Visual Communications and Image Processing II*, volume 845, pages 50–58, 1987.

[2] N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, 23:88–93, January 1974.

[3] N. Ahmed and K.R. Rao. *Orthogonal Transforms for Digital Signal Processing*. Springer-Verlag, New York, 1975.

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

[5] C. S. Barreto and G. V. Mendonca. Enhanced zerotree wavelet transform image coding exploiting similarities inside subbands. In *Proceedings IEEE Conf. on Image Proc.*, volume II, pages 549–551. IEEE Press, 1996.

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

[7] K. A. Birney and T. R. Fischer. On the modeling of DCT and subband image data for compression. *IEEE Transactions on Image Processing*, 4(2):186–193, 1995.

[8] M. Boliek, C. Christopoulos, and E. Majani. FDIS JPEG 2000 - Part I. *ISO/IEC JTC1/SC29 WG1, N2072*, March 2001.

[9] R.N Bracewell. *The Fourier Transform and Its Applications, Second Edition.* McGraw-Hill, New York, 1986.

[10] C. Brislawn. Fingerprints go digital. *Notices of the AMS*, 42(11):1278–1283, November 1995. `http://www.c3.lanl.gov/~brislawn`.

[11] R.W. Buccigrossi and E.P. Simoncelli. Image compression via joint statistical characterization in the wavelet domain. *IEEE Transactions on Image Processing*, 1999. to appear.

[12] P.J. Burt and E.H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31:532–540, 1983.

[13] A. Calway. *The Multiresolution Fourier Transform: A General Purpose Tool for Image Analysis.* PhD thesis, University of Warwick, Coventry, UK, Department of Computer Science, September 1989.

[14] H.D. Cho and J.B. Ra. A rearrangement algorithm of wavelet packet coefficients for zerotree coding. In *IEEE International Conference on Image Processing*, volume 3, pages 556–559, October 1999.

[15] R.J. Clarke. *Transform Coding of Images.* Academic Press, 1985.

[16] R.J. Clarke. *Digital Compression of Still Images and Video.* Academic Press, 1995.

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

[18] R.R. Coifman and Y. Meyer. Orthornormal wave packet bases. *Technical Report, Department of Mathematics, Yale University, New Haven*, 1990.

[19] R.R. Coifman and Y. Meyer. Remarques sur l'analyse de Fourier à fenêtre. *C.R. Acad. Sci. Paris I*, pages 259–261, 1991.

[20] R.R. Coifman and Y. Meyer. Size properties of wavelet packets. In Ruskai et al, editor, *Wavelets and their Applications*, pages 125–150. Jones and Bartlett, 1992.

[21] R.R. Coifman, Y. Meyer, and V. Wickehauser. Wavelet analysis and signal processing. In Ruskai et al, editor, *Wavelets and their Applications*, pages 153–178. Jones and Bartlett, 1992.

[22] R.R. Coifman and M.V. Wickerhauser. Entropy-based algorithms for best basis selection. *IEEE Transactions on Information Theory*, 38(2):713–718, March 1992.

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

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

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

[26] D.L. Donoho and I.M. Johnstone. Ideal spatial adaptation via wavelet shrinkage. *Biometrika*, 31:425–455, 1994.

[27] D.L. Donoho, M. Vetterli, R.A. Devore, and I. Daubechies. Data compression and harmonic analysis. *IEEE Transactions on Information Theory*, 44:2435–2476, 1998.

[28] M. Effros. Zerotree design for image compression: Toward weighted universal zerotree coding. In *IEEE Intl. Conference on Image Processing*, November 1997.

[29] D. Gabor. Theory of communication. *Journal of IEE*, 93:429–457, 1946.

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

[31] A. Grossman and J. Morlet. Decomposition of Hardy function into square integrable wavelets of constant shape. *SIAM Journal of Applied Mathematics*, pages 723–773, June 1984.

[32] R.N. Haber and M. Hershenson. *The Psychology of Visual Perception.* Holt, Rinehart and Winston, 1980.

[33] C. Herley, J. Kovacevic, K. Ramchandran, and M. Vetterli. Tilings of the time-frequency plane: Construction of arbitrary orthogonal bases and fast tiling algorithms. *IEEE Transactions on Signal Processing*, 41(12):3341–3359, December 1993.

[34] C. Herley, Z. Xiong, K. Ramchandran, and M.T. Orchard. Joint space-frequency segmentation using balanced wavelet packet trees for least-cost image representation. *IEEE Transactions on Image Processing*, 6(9):1213–1230, 1997.

[35] B. B. Hubbard. *The World According to Wavelets: The Story of a Mathematical Technique in Making.* A K Peters Ltd, 1998.

[36] D. Huffman. A method for the construction of minimum redundancy codes. *Proceedings of IRE*, 40:1098–1101, 1952.

[37] N.J. Jayant, J. Johnstone, and B. Safranek. Signal compression based on models of human perception. *Proceedings IEEE*, 81(10):1385–1422, October 1993.

[38] J.J. Koenderink. Operational significance of receptive field assemblies. *Biological Cybernetics*, 58:163–171, 1988.

[39] E. Kreyszig. *Advanced Engineering Mathematics*. John Wiley & Sons Inc., 1999.

[40] I.K. Levy. *Self-Similarity and Wavelet Transforms for the Compression of Still Image and Video Data*. PhD thesis, University of Warwick, Coventry, UK, Department of Computer Science, February 1998.

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

[42] Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, COM-28:84–95, January 1980.

[43] S.P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, IT-28:127–135, March 1982.

[44] S.M. LoPresto, K. Ramchandran, and M.T. Orchard. Image coding based on mixture modeling of wavelet coefficients and a fast estimation-quantization framework. In *IEEE Data Compression Conference, Snowbird, Utah*, March 1997.

[45] S. Mallat. A compact multiresolution representation: The wavelet model. In *IEEE Computer Society Workshop on Computer Vision*, pages 2–7, 1987.

[46] S. Mallat. A theory for multiresolution signal decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:674–693, 1989.

[47] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 1998.

[48] S. Mallat and S. Zhong. Characterization of signals from multiscale edges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(7):710–732, July 1992.

[49] H. Malvar. The LOT: Transform coding without blocking effects. *Proceedings IEEE Intl. Symp. on Circuits and Systems*, pages 835–838, 1988.

[50] D. Marpe, H.L. Cycon, and W. Li. Complexity-constrained best-basis wavelet packet algorithm for image compression. *IEE Proceedings Visual Image Processing*, 145(6):391–398, December 1998.

[51] D. Marr. *Vision, A Computational Investigation into the Human Representation and Processing of Visual Information*. W.H. Freeman, 1982.

[52] Y. Matias, N.M. Rajpoot, and S.C. Sahinalp. Implementation and experimental evaluation of flexible parsing for dynamic dictionary based data compression. In *Proceedings Workshop on Experimental Algorithms (WAE'98)*, pages 49–51, August 1998.

[53] Y. Matias, N.M. Rajpoot, and S.C. Sahinalp. The effect of flexible parsing for dynamic dictionary based data compression. In *Proceedings IEEE Data Compression Conference (DCC'99)*, pages 238–246, March 1999.

[54] Y. Matias and S. C. Sahinalp. On optimality of parsing in dynamic dictionary based data compression. In *Proceedings ACM-SIAM Symposium on Discrete Algorithms (SODA'99)*, pages 943–944, 1999.

[55] F.G. Meyer, A.Z. Averbuch, and J-O. Strömberg. Fast adaptive wavelet packet image compression. *IEEE Transactions on Image Processing*, 9(5):792–800, May 2000.

[56] F.G. Meyer, A.Z. Averbuch, J-O. Strömberg, and R.R. Coifman. Multi-layered image representation: Application to image compression. In *International Conference on Image Processing, ICIP'98, Chicago, October 4-7, 1998*, 1998.

[57] F.G. Meyer and R.R. Coifman. Brushlets: a tool for directional image analysis and image compression. *Applied and Computational Harmonic Analysis*, pages 147–187, 1997.

[58] Y. Meyer. Principe d'incertitude, bases Hilbertiennes et algebres d'operateurs. Seminaire Bourbaki, 662, 1985.

[59] H. Murakami, Y. Hatori, and H. Yamamoto. Comparison between DPCM and Hadamard transform coding in the composite coding of the NTSC color TV signal. *IEEE Transactions on Communications*, COM-30:469–479, March 1982.

[60] A. Papoulis. *The Fourier Integral and Its Applications, Second Edition.* McGraw-Hill, New York, 1987.

[61] E.R.S. Pearson. *The Multiresolution Fourier Transform and Its Application to the Analysis of Polyphonic Music.* PhD thesis, University of Warwick, Coventry, UK, Department of Computer Science, 1989.

[62] W. Pennebaker and J. Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1992.

[63] M. Rabbani and P.W. Jones. *Digital Image Compression Techniques.* PIE Optical Engineering Press, 1991.

[64] N.M. Rajpoot, F.G. Meyer, R.G. Wilson, and R.R. Coifman. On zerotree quantization for embedded wavelet packet image coding. In *Proceedings IEEE International Conference on Image Processing*, pages 283–287, October 1999.

[65] N.M. Rajpoot, F.G. Meyer, R.G. Wilson, and R.R. Coifman. Wavelet packet image coding using compatible zerotree quantization. Technical report, YALEU/DCS/RR-1187, Department of Computer Science, Yale University, September 1999.

[66] N.M. Rajpoot and R.G. Wilson. Progressive image coding using augmented zerotrees of wavelet coefficients. Technical report, Department of Computer Science, University of Warwick, Coventry, UK, September 1998. Technical Report CS-RR-350.

[67] K. Ramchandran and M. Vetterli. Best wavelet packet bases in a rate-distortion sense. *IEEE Transactions on Image Processing*, pages 160–175, April 1993.

[68] R. C. Reininger and J. D. Gibson. Distributions of the two-dimensional DCT coefficients for images. *IEEE Transactions on Communications*, COM-31(6):835–839, 1983.

[69] J. Rissanen. Generalized kraft inequality and arithmetic coding. *IBM Journal of Research Development*, 20:198–203, 1976.

[70] A. Rosenfeld and M. Thurston. Edge and curve detection for visual scene analysis. *IEEE Transactions on Computers*, 20:562–569, 1971.

[71] A. Said and W.A. Pearlman. A new fast and efficient image codec based on set partioning in hierarchical trees. *IEEE Transactions on Circuits & Systems for Video Technology*, 6:243–250, June 1996.

[72] C.E. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423, 623–656, 1948.

[73] J.M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, pages 3445–3462, Dec. 1993.

[74] J.R. Smith and S-F. Chang. Frequency and spatially adaptive wavelet packets. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 4, pages 2233–2236, May 1995.

[75] P. Sriram and M.W. Marcellin. Image coding using wavelet transforms and entropy-constrained trellis quantization. *IEEE Transactions on Image Processing*, 4:725–733, 1995.

[76] J.A. Storer. *Data Compression: Methods and Theory.* Computer Science Press, 1988.

[77] G. Strang and T. Nguyen. *Wavelets and Filter Banks.* Wellesley-Cambridge Press, revised edition edition, 1997.

[78] J.-O. Strömberg. A modified Franklin system and higher order spline systems on R^n as unconditional bases for Hardy spaces. In W. Beckner et al., editor, *Proceedings Conference in Honor of A. Zygmund*, pages 475–493. Wadsworth Mathematical Series, 1981.

[79] G.J. Sullivan. Efficient scalar quantization of exponential and Laplacian random variables. *IEEE Transactions on Information Theory*, 42(5):1365–1374, September 1996.

[80] N. Tanabe and N. Farvardin. Subband image coding using entropy-coded quantization over noisy channels. Technical report, *Computer Science Technical Report Series*, University of Maryland, College Park, MD, August 1995.

[81] S.L. Tanimoto and T. Pavlidis. A hierarchical data structure for picture processing. *Computer Graphics and Image Processing*, 4:104–119, 1974.

[82] C. Taswell. Near-best bases selection algorithms with non-additive information cost functions. In *Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, pages 13–16. IEEE Press, 1994.

[83] C. Taswell. Satisficing search algorithms for selecting near-best bases in adaptive tree-structured wavelet transforms. *IEEE Transactions on Signal Processing*, 44(10):2423–2438, October 1996.

[84] M.J. Tsai, J.D. Villasenor, and F. Chen. Stack-run image coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 6:519–521, October 1996.

[85] M. Vetterli and J. Kovacevic. *Wavelets and Subband Coding*. Prentice Hall, New Jersey, 1995.

[86] J. Ville. Theorie et applications de la notion de signal analytique. *Cables et Transm.*, 2A(1):61–74, 1948.

[87] Y. Viniotis. *Probability and Random Processes for Electrical Engineers*. McGraw-Hill, 1998.

[88] T.A. Welch. A technique for high-performance data compression. *IEEE Computer*, pages 8–19, Jan 1984.

[89] M.V. Wickerhauser. *Adapted Wavelet Analysis from Theory to Software*. A.K. Peters, 1995.

[90] E.P. Wigner. On the quantum correction for thermodynamic equilibrium. *Physics Review*, 40:749–759, 1932.

[91] K.G. Wilson. Generalized Wannier functions. Technical report, Cornell University, 1987.

[92] R. Wilson. BMVC93 tutorial notes on wavelet transforms. In *British Machine Vision Conference*, 1993.

[93] R. Wilson, I. Levy, and P. Meulemans. Symmetry and wavelet transforms for image data compression. In J.G. McWhirter and I.K. Proudler, editors, *Mathematics in Signal Processing*. Oxford University Press, 1998.

[94] R. Wilson and M. Spann. *Image Segmentation and Uncertainty*. Wiley, 1988.

[95] R.G. Wilson. Quad-tree predictive coding: A new class of image data compression algorithms. In *Proceedings IEEE Conference on Acoustics, Speech, and Signal Processing*, pages 29.3.1–29.3.4, San Diego, March 1984.

[96] R.G. Wilson, A.D. Calway, and E.R.S. Pearson. A generalized wavelet transform for Fourier analysis: The multiresolution Fourier transform and its applications to image and audio signal analysis. *IEEE Transancation on Information Theory*, 38(2):674–690, March 1992.

[97] R.G. Wilson and G.H. Granlund. The uncertainty principle in image processing. *IEEE Transancation on Pattern Analysis and Machine Intelligence*, PAMI-6(6):758–767, November 1984.

[98] A.P. Witkin. Scale space filtering. In *Proceedings IJCAI*. Karlsruhe, 1983.

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

[100] J.W. Woods. *Subband Image Coding*. Kluwer, Boston MA, USA, 1991.

[101] X. Wu. High-order context modeling and embedded conditional entropy coding of wavelet coefficients for image compression. submitted to IEEE Transactions on Image Processing, 1997.

[102] Z. Xiong, O. Guleryuz, and M.T. Orchard. A DCT-based embedded image coder. *IEEE Signal Processing Letters*, 3(11):289–290, 1996.

[103] Z. Xiong, K. Ramchandran, and M.T. Orchard. Wavelet packet image coding using space-frequency quantization. *IEEE Transactions on Image Processing*, 7(6):892–898, 1998.

[104] H. Yokoo. Improved variations relating the Ziv-Lempel and welch-type algorithms for sequential data compression. *IEEE Transactions on Information Theory*, 38(1):73–81, Jan 1992.

[105] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, IT-23(3):337–343, May 1977.

[106] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, IT-24(5):530–536, Sep 1978.