5.4

Wavelet Image Compression

Zixiang Xiong
Texas A&M University

Kannan Ramchandran
University of California, Berkeley

1 What Are Wavelets: Why Are They Good for Image Coding?

During the past decade, wavelets have made quite a splash in the field of image compression. In fact, the FBI has already adopted a wavelet-based standard for fingerprint image compression. The new JPEG2000 image compression standard [1], which has dislodged the old JPEG standard (see Chapter 5.5), is also based on wavelets. Given these exciting developments, it is natural to ask why wavelets have made such an impact in image compression. This chapter will answer this question; providing both high-level intuition as well as illustrative details based on state-of-the-art wavelet-based coding algorithms. Visually appealing time-frequency based analysis tools are sprinkled generously to aid in our task.

Wavelets are tools for decomposing signals, such as images, into a hierarchy of increasing resolutions: as we consider more and more resolution layers, we get a more and more detailed look at the image. Figure 1 shows a three-level hierarchy wavelet decomposition of the popular test image *Lena* from coarse to fine resolutions (for a detailed treatment on wavelets and multiresolution decompositions, also see Chapter 4.2). Wavelets can be regarded as "mathematic microscopes" that permit one to "zoom in" and "zoom out" of images at multiple resolutions. The remarkable thing about the wavelet decomposition is that it enables this zooming feature at absolutely no cost in terms of excess redundancy: for an $M \times N$ image, there are exactly MN wavelet coefficients—exactly the same as the number of original image pixels (see Fig. 2).

As a basic tool for decomposing signals, wavelets can be considered as duals to the more traditional Fourier-based analysis methods that we encounter in traditional undergraduate engineering curricula. Fourier analysis associates the very intuitive engineering concept of "spectrum" or "frequency content" of a signal. Wavelet analysis, in contrast, associates the equally intuitive concept of "resolution" or "scale" of the signal. At a functional level, Fourier analysis is to wavelet analysis as spectrum analyzers are to microscopes.

As wavelets and multiresolution decompositions have been described in greater depth in Chapter 4.2, our focus here will be more on the image compression application. Our goal is to provide a self-contained treatment of wavelets within the scope of their role in image compression. More importantly, our goal is to provide a high-level explanation for why they have made such an impact in image compression. Indeed, wavelets have dislodged the more traditional Fourier-based method in the form of the discrete cosine transform (DCT) that is deployed in the old JPEG image compression standard (see Chapter 5.5). We will also cover powerful generalizations of wavelets, known as wavelet packets, that

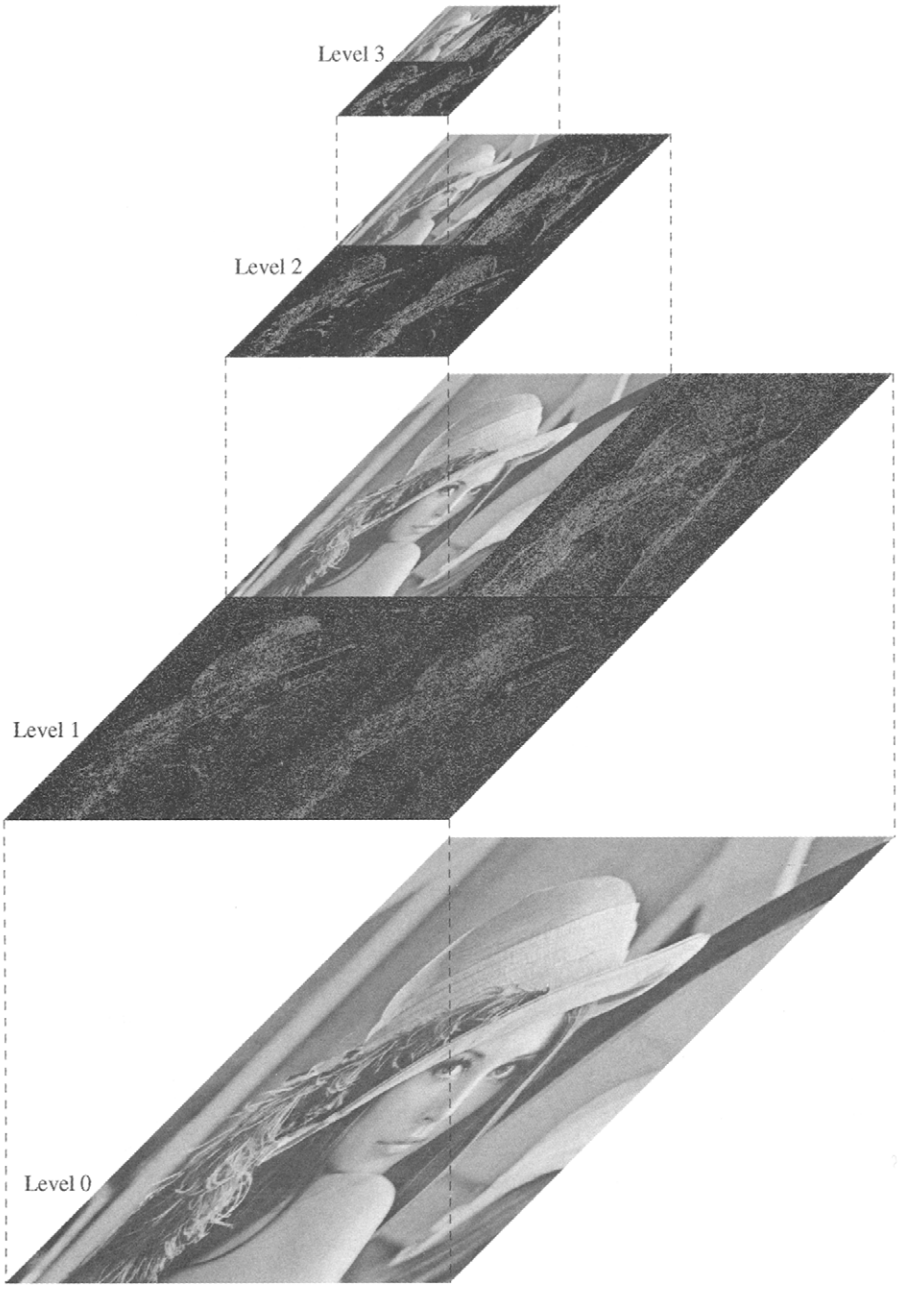

Figure 1 A three-level hierarchy wavelet decomposition of the *512* × *512* color *Lena* image. Level 1 (*512* × *512*) is the one-level wavelet representation of the original *Lena* at Level 0; Level 2 (*256* × *256*) shows the one-level wavelet representation of the lowpass image at Level 1; and Level 3 (*128* × *128*) gives the one-level wavelet representation of the lowpass image at Level 2. (See color insert.)

have already made an impact in the standardization world: the FBI fingerprint compression standard is based on wavelet packets.

Although this chapter is about image coding,[1] which involves two-dimensional (2D) signals or images, it is much easier to understand the role of wavelets in image coding using a one-dimensional (1D) framework, as the conceptual extension to 2D is straightforward. In the interests of clarity, we will therefore consider a 1D treatment here. The story begins with what is known as the time-frequency analysis of the 1D signal. As mentioned, wavelets are a tool for changing the coordinate system in which we represent the signal: we transform the signal into another domain that is much better suited for processing, e.g., compression. What makes for a good transform or analysis tool? At the basic level, the

[1]We use terms *image compression* and *image coding* interchangeably in this chapter

Figure 2 A three-level wavelet representation of the *Lena* image generated from the top view of the three-level hierarchy wavelet decomposition in Figure 1. It has exactly the same number of samples as in the image domain (see color insert).

goal is to be able to represent all the useful signal features and important phenomena in as compact a manner as possible. It is important to be able to compact the bulk of the signal energy into the fewest number of transform coefficients: this way, we can discard the bulk of the transform domain data without losing too much information. For example, if the signal is a time impulse, then the best thing is to do no transforms at all! Keep the signal information in its original time-domain version, as that will maximize the temporal energy concentration or time resolution. However, what if the signal has a critical frequency component (e.g., a low frequency background sinusoid) that lasts for a long time duration? In this case, the energy is spread out in the time domain, but it would be succinctly captured in a single frequency coefficient if one did a Fourier analysis of the signal. If we know that the signals of interest are pure sinusoids, then Fourier analysis is the way to go. But, what if we want to capture both the time impulse and the frequency impulse with good resolution? Can we get arbitrarily fine resolution in both time and frequency?

The answer is no. There exists an uncertainty theorem (much like what we learn in quantum physics), which disallows the existence of arbitrary resolution in time and frequency [2]. A good way of conceptualizing these ideas and the role of wavelet basis functions is through what is known as time-frequency "tiling" plots, as shown in Fig. 3, which shows where the basis functions live on the time-frequency plane: i.e., where is the bulk of the energy of the elementary basis elements localized? Consider the Fourier case first. As impulses in time are completely spread out in the frequency domain, all localization is lost with Fourier analysis. To alleviate this problem, one typically decomposes the signal into finite-length chunks using windows or so-called short-time-Fourier-transform (STFT). Then, the time-frequency tradeoffs will be determined by the window size. An STFT expansion consists of basis functions that are shifted versions

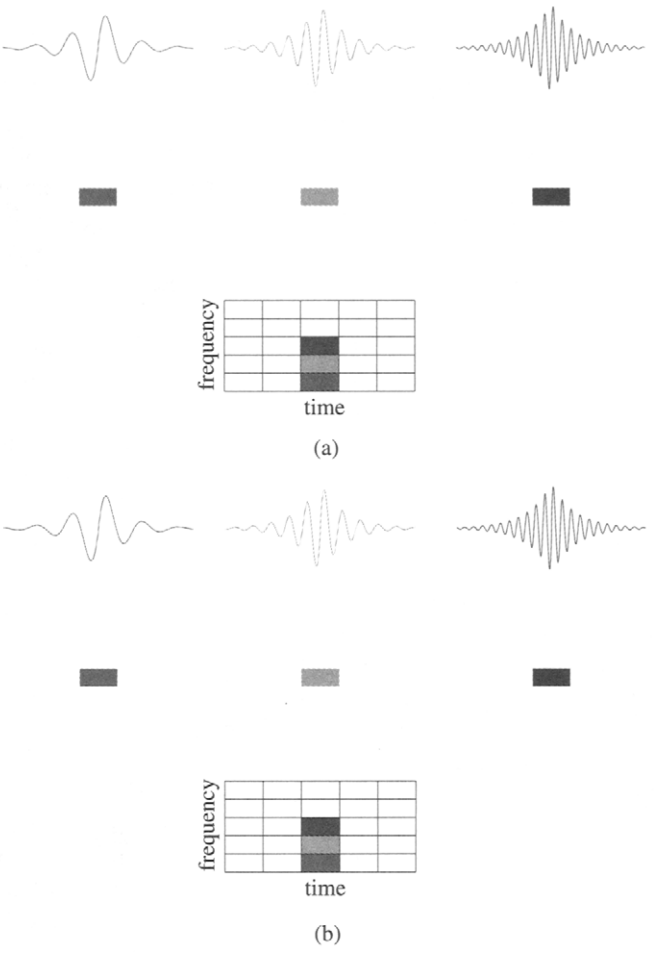

Figure 3 Tiling diagrams associated STFT bases and wavelet bases. (a) STFT bases and the tiling diagram associated with a STFT expansion. STFT bases of different frequencies have the same resolution (or length) in time. (b) Wavelet bases and tiling diagram associated with a wavelet expansion. The time resolution is inversely proportional to frequency for wavelet basis (see color insert).

of one another in both time and frequency: some elements capture low frequency events localized in time, and others capture high frequency events localized in time, but the resolution or window size is constant in both time and frequency (see Fig. 3(a)). Note that the uncertainty theorem says that the area of these tiles has to be nonzero.

Shown in Fig. 3(b) is the corresponding tiling diagram associated with the wavelet expansion. The key difference between this and the Fourier case, which is the critical point, is that the tiles are not all of the same size in time (or frequency). Some basis elements have short time windows; others have short frequency windows. Of course, the uncertainty theorem ensures that the area of each tile is constant and nonzero. It can be shown that the basis functions are related to one another by shifts and scales as is the key to wavelet analysis.

Why are wavelets well-suited for image compression? The answer lies in the time-frequency (or more correctly, space-frequency) characteristics of typical natural images, which

turn out to be well captured by the wavelet basis functions shown in Fig. 3(b). Note that the STFT tiling diagram of Fig. 3(a) is conceptually similar to what commercial DCT-based image transform coding methods like JPEG use. Why are wavelets inherently a better choice? Looking at Fig. 3(b); one can note that the wavelet basis offers elements having good frequency resolution at lower frequency (the short and fat basis elements) while simultaneously offering elements that have good time resolution at higher frequencies (the tall and skinny basis elements).

This tradeoff works well for natural images and scenes that are typically composed of a mixture of important long-term low frequency trends that have larger spatial duration (such as slowly varying backgrounds e.g., the blue sky, the surface of lakes, etc.) as well as important transient short duration high frequency phenomena such as sharp edges. The wavelet representation turns out to be particularly well-suited to capturing both the transient high frequency phenomena such as image edges (using the tall and skinny tiles) as well as long-spatial-duration low frequency phenomena such as image backgrounds (the short and fat tiles). As natural images are dominated by a mixture of these kinds of events,[2] wavelets promise to be very efficient in capturing the bulk of the image energy in a small fraction of the coefficients.

To summarize, the task of separating transient behavior from long-term trends is a very difficult task in image analysis and compression. In the case of images, the difficulty stems from the fact that statistical analysis methods often require the introduction of at least some local stationarity assumption, i.e., that the image statistics do not change abruptly over time. In practice, this assumption usually translates into *ad hoc* methods to block data samples for analysis, methods that can potentially obscure important signal features: e.g., if a block is chosen too big, a transient component might be totally neglected when computing averages. The blocking artifact in JPEG decoded images at low rates is a result of the block-based DCT approach. A fundamental contribution of wavelet theory [3] is that it provides a unified framework in which transients and trends can be simultaneously analyzed without the need to resort to blocking methods.

As a way of highlighting the benefits of having a sparse representation, such as that provided by the wavelet decomposition, consider the lowest frequency band in the top level (level 3) of the three-level wavelet hierarchy of *Lena* in Fig. 1. This band is just a downsampled (by a factor of $8^2 = 64$) and smoothed version of the original image. A very simple way of achieving compression is to simply retain this lowpass version and throw away the rest of the wavelet data, instantly achieving a compression ratio of 64:1. Note that if we want a full-size approximation to the original, we would have to interpolate the lowpass band by a factor of 64—this can be done efficiently by using a three-stage synthesis filter bank (see Chapter 4.2). We may also desire better image fidelity, as we may be compromising high frequency image detail, especially perceptually important high frequency edge information. This is where wavelets are particularly attractive, as they are capable of capturing most image information in the highly subsampled low frequency band, and additional localized edge information in spatial clusters of coefficients in the high frequency bands (see Fig. 1). The bulk of the wavelet data is insignificant and can be discarded or quantized very coarsely.

Another attractive aspect of the coarse-to-fine nature of the wavelet representation naturally facilitates a transmission scheme that progressively refines the received image quality. That is, it would be highly beneficial to have an encoded bitstream that can be chopped off at any desired point to provide a commensurate reconstruction image quality. This is known as a progressive transmission feature or as an embedded bitstream (see Fig. 4). Many modern wavelet image coders have this feature, as will be covered in more detail in Section 5. This is ideally suited, for example, to Internet image applications. As is well known, the Internet is a heterogeneous mess in terms of the number of users and their computational capabilities and effective bandwidths. Wavelets provide a natural way to satisfy users having disparate bandwidth and computational capabilities: the low-end users can be provided a coarse quality approximation, whereas higher-end users can use their increased bandwidth to get better fidelity. This is also very useful for Web browsing applications, where having a coarse quality image with a short waiting time may be preferable to having a detailed quality with an unacceptable delay.

These are some of the high-level reasons why wavelets represent a superior alternative to traditional Fourier-based methods for compressing natural images: this is why the JPEG2000 standard [1] uses wavelets instead of the Fourier-based DCTs.

In the sequel, we will review the salient aspects of the general compression problem and the transform coding paradigm in particular, and highlight the key differences between the class of early subband coders and the recent more advanced class of modern-day wavelet image coders. We pick the celebrated embedded zerotree wavelet (EZW) coder as a representative of this latter class, and we describe its operation by using a simple illustrative example. We conclude with more powerful generalizations of the basic wavelet image coding framework to wavelet packets, which are particularly well-suited to handle special classes of images such as fingerprints.

2 The Compression Problem

Image compression falls under the general umbrella of data compression, which has been studied theoretically in the field

[2]Typical images also contain textures; however, conceptually, textures can be assumed to be a dense concentration of edges, and so it is fairly accurate to model typical images as smooth regions delimited by edges.

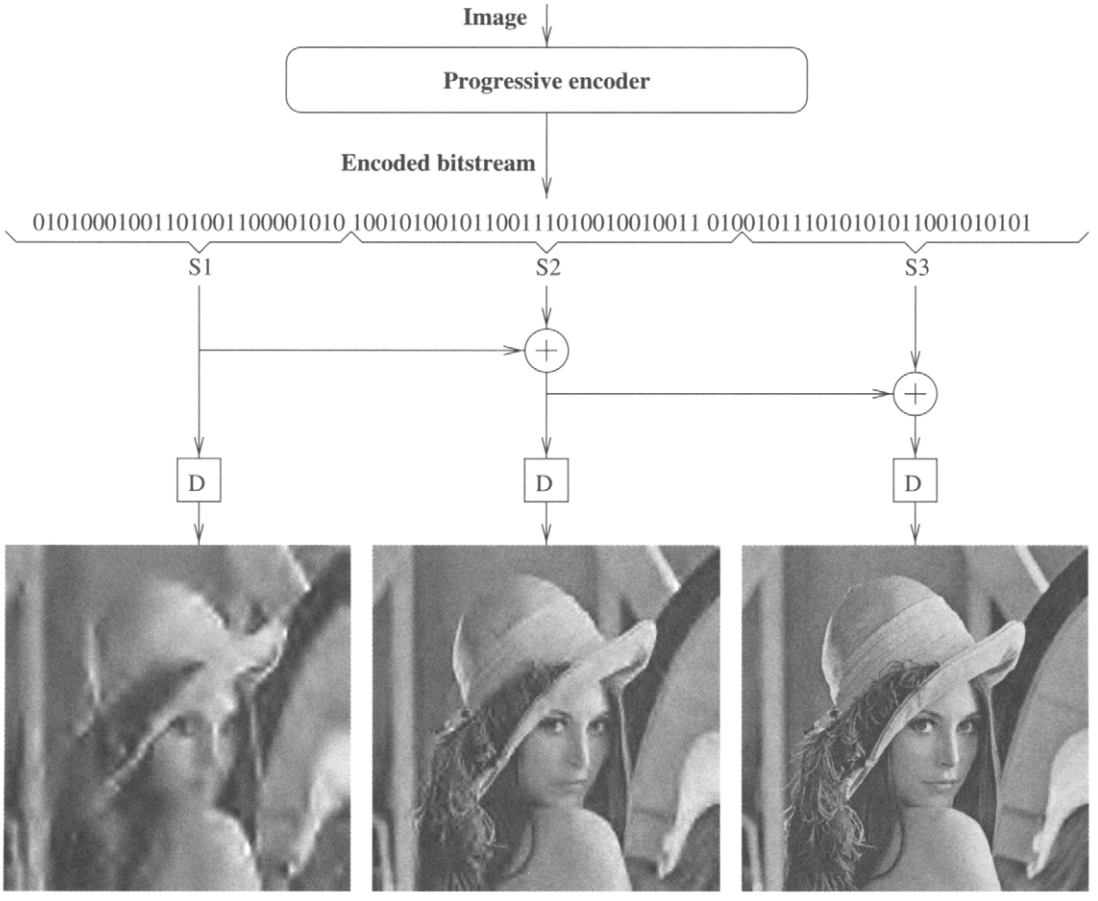

Figure 4 Multiresolution wavelet image representation naturally facilitates progressive transmission—a desirable feature for the transmission of compressed images over heterogeneous packet networks and wireless channels.

of information theory [4], pioneered by Claude Shannon [5] in 1948. Information theory sets the fundamental bounds on compression performance theoretically attainable for certain classes of sources. This is very useful because it provides a theoretical benchmark against which one can compare the performance of more practical but suboptimal coding algorithms.

Historically, the lossless compression problem came first. Here the goal is to compress the source with no loss of information. Shannon showed that given any discrete source with a well-defined statistical characterization (i.e., a probability mass function), there is a fundamental theoretical limit to how well you can compress the source before you start to lose information. This limit is called the *entropy* of the source. In lay terms, entropy refers to the uncertainty of the source. For example, a source that takes on any of N discrete values $a_1, a_2, \ldots, a_N$ with equal probability has an entropy given by $\log_2 N$ bits per source symbol. If the symbols are not equally likely, however, then one can do better because more predictable symbols should be assigned fewer bits. The fundamental limit is the Shannon entropy of the source.

Lossless compression of images has been covered in Chapter 5.1 and Chapter 5.6. For image coding, typical lossless compression ratios are of the order of 2:1 or at most 3:1. For a 512×512 8-bit grayscale image, the uncompressed representation is 256 Kbytes. Lossless compression would reduce this to at best $\sim$ 80 Kbytes, which may still be excessive for many practical low bandwidth transmission applications. Furthermore, lossless image compression is for the most part overkill, as our human visual system is highly tolerant to losses in visual information. For compression ratios in the range of 10:1 to 40:1 or more, lossless compression cannot do the job, and one needs to resort to lossy compression methods.

The formulation of the lossy data compression framework was also pioneered by Shannon in his work on rate-distortion (R-D) theory [6], in which he formalized the theory of compressing certain limited classes of sources having well-defined statistical properties, e.g., independent identically distributed (i.i.d.) sources having a Gaussian distribution subject to a fidelity criterion, i.e., subject to a tolerance on the maximum allowable loss or distortion that can be endured. Typical distortion measures used are mean square error (MSE) or peak signal-to-noise ratio (PSNR)[3] between the original and compressed versions. These fundamental

[3]The PSNR is defined as $10\log \frac{255^2}{\text{MSE}}$ and measured in decibels (dB).

compression performance bounds are called the theoretical R-D bounds for the source: they dictate the minimum rate R needed to compress the source if the tolerable distortion level is D (or alternatively, what is the minimum distortion D subject to a bit rate of R). These bounds are unfortunately not constructive; i.e., Shannon did not give an actual algorithm for attaining these bounds, and furthermore they are based on arguments that assume infinite complexity and delay, obviously impractical in real life. However, these bounds are useful in as much as they provide valuable benchmarks for assessing the performance of more practical coding algorithms. The major obstacle of course, as in the lossless case, is that these theoretical bounds are available only for a narrow class of sources, and it is difficult to make the connection to real world image sources which are difficult to model accurately with simplistic statistical models.

Shannon's theoretical R-D framework has inspired the design of more practical *operational* R-D frameworks, in which the goal is similar but the framework is constrained to be more practical. Within the operational constraints of the chosen coding framework, the goal of operational R-D theory is to minimize the rate R subject to a distortion constraint D, or vice versa. The message of Shannon's R-D theory is that one can come close to the theoretical compression limit of the source, if one considers *vectors* of source symbols that get infinitely large in dimension in the limit; i.e., it is a good idea not to code the source symbols one at a time, but to consider chunks of them at a time, and the bigger the chunks the better. This thinking has spawned an important field known as *vector quantization*, or VQ [7], which, as the name indicates, is concerned with the theory and practice of quantizing sources using high dimensional vector quantization (image coding using VQ is covered in Chapter 5.3). There are practical difficulties arising from making these vectors too high-dimensional because of complexity constraints, so practical frameworks involve relatively small dimensional vectors that are therefore further from the theoretical bound.

Because of this reason, there has been a much more popular image compression framework that has taken off in practice: this is the transform coding framework [8] that forms the basis of current commercial image and video compression standards like JPEG and MPEG (see Chapters 6.3 and 6.4). The transform coding paradigm can be construed as a practical special case of VQ that can attain the promised gains of processing source symbols in vectors through the use of efficiently implemented high dimensional source transforms.

3 The Transform Coding Paradigm

In a typical transform image coding system, the encoder consists of a linear transform operation, followed by quantization of transform coefficients, and lossless compression of the quantized coefficients using an entropy coder. After the encoded bitstream of an input image is transmitted over the channel (assumed to be perfect), the decoder undoes all the functionalities applied in the encoder and tries to reconstruct a decoded image that looks as close as possible to the original input image, based on the transmitted information. A block diagram of this transform image paradigm is shown in Fig. 5.

For the sake of simplicity, let us look at a 1D example of how transform coding is done (for 2D images, we treat the rows and columns separately as 1D signals). Suppose we have a two-point signal: $x_0 = 216$, $x_1 = 217$. It takes 16 bits (8 bits for each sample) to store this signal in a computer. In transform coding, we first put x_0 and x_1 in a column vector $X = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}$ and apply an orthogonal transformation T to X to get

$$Y = \begin{bmatrix} y_0 \\ y_1 \end{bmatrix} = TX = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}$$

$$= \begin{bmatrix} (x_0 + x_1)/\sqrt{2} \\ (x_0 - x_1)/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 306.177 \\ -.707 \end{bmatrix}.$$

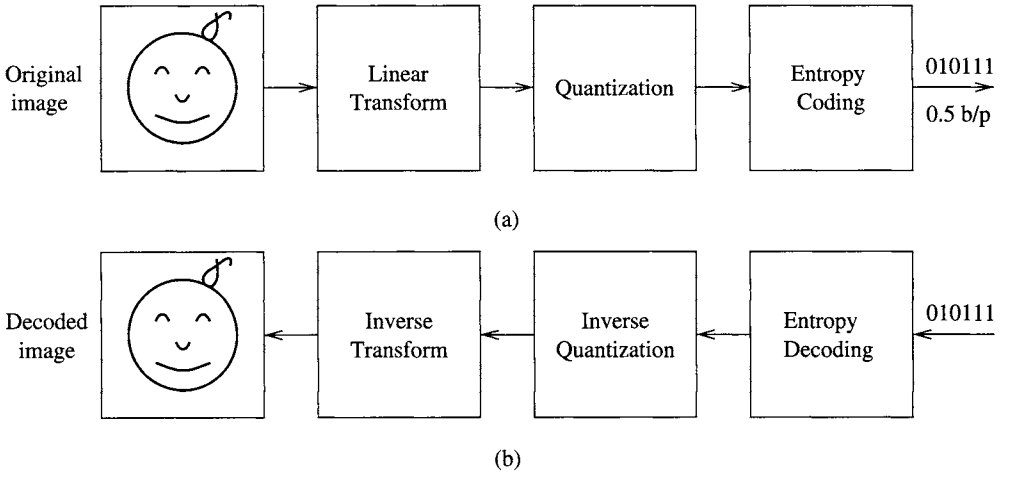

Figure 5 Block diagrams of a typical transform image coding system: (a) encoder and (b) decoder diagrams.

The transform T can be conceptualized as a counter-clockwise rotation of the signal vector X by 45° with respect to the original (x_0, x_1) coordinate system. Alternatively and more conveniently, one can think of the signal vector as being fixed and instead rotate the (x_0, x_1) coordinate system by 45° clockwise to the new (y_1, y_0) coordinate system (see Fig. 6). Note that the abscissa for the new coordinate system is now y_1.

Orthogonality of the transform simply means that the length of Y is the same as the length of X (which is even more obvious when one freezes the signal vector and rotates the coordinate system as discussed above). This concept still carries over to the case of high dimensional transforms. If we decide to use the simplest form of quantization known as uniform scalar quantization, where we round off a real number to the nearest integer multiple of a step size q (say $q=20$), then the quantizer index vector $\hat{I}$, which captures *what* integer multiples of q are nearest to the entries of Y, is given by

$$\hat{I} = \begin{bmatrix} \text{round}(y_0/q) \\ \text{round}(y_1/q) \end{bmatrix} = \begin{bmatrix} 15 \\ 0 \end{bmatrix}.$$

We store (or transmit) $\hat{I}$ as the compressed version of X using 4 bits, achieving a compression ratio of 4:1. To decode X from $\hat{I}$, we first multiply $\hat{I}$ by $q=20$ to dequantize, i.e., to form the quantized approximation $\hat{Y}$ of Y with $\hat{Y} = q \cdot \hat{I} = \begin{bmatrix} 300 \\ 0 \end{bmatrix}$, and then apply the inverse transform T^{-1} to $\hat{Y}$ (which corresponds in our example to a counter-clockwise rotation of the (y_1, y_0) coordinate system by 45°, just the reverse operation of the T operation on the original (x_0, x_1) coordinate system—see Fig. 6) to get

$$\hat{X} = T^{-1}\begin{bmatrix} qy_0 \\ qy_1 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}\begin{bmatrix} 300 \\ 0 \end{bmatrix} = \begin{bmatrix} 212.132 \\ 212.132 \end{bmatrix}.$$

We see from the above example that, although we "zero out" or throw away the transform coefficient y_1 in quantization, the decoded version $\hat{X}$ is still very close to X. This is because the transform effectively compacts most of the energy in X into the first coefficient y_0, and renders the second coefficient y_1 considerably insignificant to keep. The transform T in our example actually computes a weighted sum and difference of the two samples x_0 and x_1 in a manner that preserves the original energy. It is in fact the *simplest* wavelet transform!

The energy compaction aspect of wavelet transforms was highlighted in Section 1. Another goal of linear transformation is decorrelation. This can be seen from the fact that, although the values of x_0 and x_1 are very close (highly correlated) before the transform, y_0 (sum) and y_1 (difference) are very different (less correlated) after the transform. Decorrelation has a nice geometric interpretation. A cloud of input

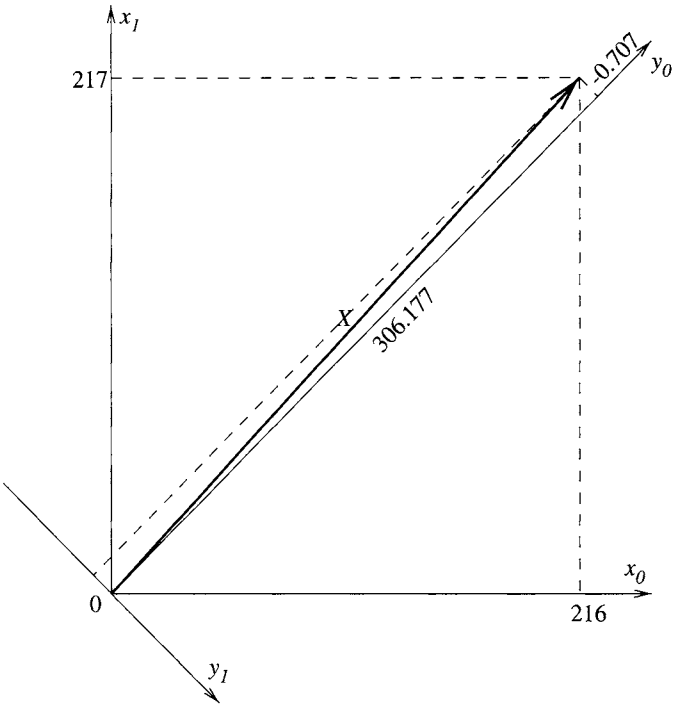

Figure 6 The transform T can be conceptualized as a counter-clockwise rotation of the signal vector X by 45° with respect to the original (x_0, x_1) coordinate system.

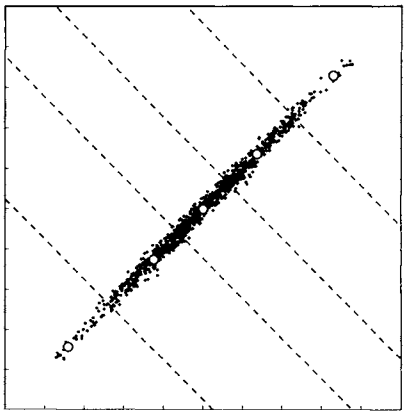

Figure 7 Linear transformation amounts to a rotation of the coordinate system, making correlated samples in the time domain less correlated in the transform domain.

samples of length-2 is shown along the 45° line in Fig. 7. The coordinates (x_0, x_1) at each point of the cloud are nearly the same, reflecting the high degree of correlation among neighboring image pixels. The linear transformation T essentially amounts to a rotation of the coordinate system. The axes of the new coordinate system are parallel and perpendicular to the orientation of the cloud. The coordinates (y_0, y_1) are less correlated, as their magnitudes can be quite different and the sign of y_1 is random. If we assume x_0 and x_1 are samples of a stationary random sequence $X(n)$, then the

correlation between y_0 and y_1 is $E\{y_0 y_1\} = E\{(x_0^2 - x_1^2)/2\} = 0$. This decorrelation property has significance in terms of how much gain one can get from transform coding than from doing signal processing (quantization and coding) directly in the original signal domain, called pulse code modulation (PCM) coding.

Transform coding has been extensively developed for coding of images and video, where the DCT is commonly used because of its computational simplicity and its good performance. But as shown in Section 1, the DCT is giving way to the wavelet transform because of the latter's superior energy compaction capability when applied to natural images. Before discussing state-of-the-art wavelet coders and their advanced features, we address the functional units that comprise a transform coding system, namely the transform, quantizer, and entropy coder (see Fig. 5).

3.1 Transform Structure

The basic idea behind using a linear transformation is to make the task of compressing an image in the transform domain after quantization easier than direct coding in the spatial domain. A good transform, as has been mentioned, should be able to *decorrelate* the image pixels, and provide good *energy compaction* in the transform domain so that very few quantized non-zero coefficients have to be encoded. It is also desirable for the transform to be orthogonal so that the energy is conserved from the spatial domain to the transform domain, and the distortion in the spatial domain introduced by quantization of transform coefficients can be directly examined in the transform domain. What makes the wavelet transform special in all possible choices is that it offers an efficient space-frequency characterization for a broad class of natural images, as shown in Section 1.

3.2 Quantization

As the only source of information loss occurs in the quantization unit, efficient quantizer design is a key component in wavelet image coding. Quantizers come in many different shapes and forms, from very simple uniform scalar quantizers, such as the one in the example earlier, to very complicated vector quantizers. Fixed length uniform scalar quantizers are the simplest kind of quantizers: these simply round off real numbers to the nearest integer multiples of a chosen step size. The quantizers are fixed length in the sense that all quantization levels are assigned the same number of bits (e.g., an eight-level quantizer would be assigned all binary three-tuples between 000 and 111). Fixed length *nonuniform* scalar quantizers, in which the quantizer step sizes are not all the same, are more powerful: one can optimize the design of these nonuniform step sizes to get what is known as Lloyd–Max quantizers [9].

It is more efficient to do a joint design of the quantizer and the entropy coding functional unit (this will be described in the next subsection) that follows the quantizer in a lossy compression system. This joint design results in a so-called entropy-constrained quantizer that is more efficient but more complex, and results in variable length quantizers, in which the different quantization choices are assigned variable codelengths. Variable length quantizers can come in either scalar; known as entropy-constrained scalar quantization, or ECSQ [10]; or vector varieties; known as entropy-constrained vector quantization, or ECVQ [7]. An efficient way of implementing vector quantizers is by the use of so-called trellis coded quantization, or TCQ [11]. The performance of the quantizer (in conjunction with the entropy coder) characterizes the operational R-D function of the source. The theoretical R-D function characterizes the fundamental lossy compression limit theoretically attainable [12], and it is rarely known in analytical form except for a few special cases, such as the i.i.d. Gaussian source [4]:

$$D(R) = \sigma^2 2^{-2R}, \tag{1}$$

where the Gaussian source is assumed to have zero mean and variance σ^2 and the rate R is measured in bits per sample. Note from the formula that every extra bit reduces the expected distortion by a factor of 4 (or increases the signal to noise ratio by 6 dB). This formula agrees with our intuition that the distortion should decrease exponentially as the rate increases. In fact, this is true when quantizing sources with other probability distributions as well under high resolution (or bit rate) conditions: the optimal R-D performance of encoding a zero mean stationary source with variance σ^2 takes the form of [7]

$$D(R) = h\sigma^2 2^{-2R}, \tag{2}$$

where the factor h depends on the probability distribution of the source. For a Gaussian source, $h = \sqrt{3}\pi/2$ with optimal scalar quantization. Under high resolution conditions, it can be shown that the optimal entropy-constrained scalar quantizer is a uniform one, whose average distortion is only approximately 1.53 dB worse than the theoretical bound attainable that is known as the Shannon bound [7, 10]. For low bit rate coding, most current subband coders employ a uniform quantizer with a "deadzone" in the central quantization bin. This simply means that the all-important central bin is wider than the other bins: this turns out to be more efficient than having all bins be of the same size. The performance of deadzone quantizers is nearly optimal for memoryless sources even at low rates [13]. An additional advantage of using deadzone quantization is that, when the deadzone is twice as much as the uniform step size, an embedded bitstream can be generated by successive quantization. We will elaborate more on embedded wavelet image coding in Section 5.

3.3 Entropy Coding

Once the quantization process is completed, the last encoding step is to use entropy coding to achieve the entropy rate of the quantizer. Entropy coding works like the Morse code in electric telegraph: more frequently occurring symbols are represented by short codewords, whereas symbols occurring less frequently are represented by longer codewords. On average, entropy coding does better than assigning the same codelength to all symbols. For example, a source that can take on any of the four symbols $\{A, B, C, D\}$ with equal likelihood has two bits of information or uncertainty, and its entropy is 2 bits per symbol (e.g., one can assign a binary code of 00 to A, 01 to B, 10 to C, and 11 to D). However, if the symbols are not equally likely, e.g., if the probabilities of A, B, C, D are $0.5, 0.25, 0.125, 0.125$, respectively, then one can do much better on average by not assigning the same number of bits to each symbol, but rather by assigning fewer bits to the more popular or predictable ones. This results in a variable length code. In fact, one can show that the optimal code would be one in which A gets 1 bit, B gets 2 bits, and C and D get 3 bits each (e.g., $A = 0, B = 10, C = 110, D = 111$). This is called an entropy code. With this code, one can compress the source with an average of only 1.75 bits per symbol, a 12.5% improvement in compression over the original 2 bits per symbol associated with having fixed length codes for the symbols. The two popular entropy coding methods are Huffman coding [14] and arithmetic coding [15]. A comprehensive coverage of entropy coding is given in Chapter 5.1. The Shannon entropy [4] provides a lower bound in terms of the amount of compression entropy coding can best achieve. The optimal entropy code constructed in the example actually achieves the theoretical Shannon entropy of the source.

4 Subband Coding: The Early Days

Subband coding normally uses bases of roughly equal bandwidth. Wavelet image coding can be viewed as a special case of subband coding with logarithmically varying bandwidth bases that satisfy certain properties.[4] Early work on wavelet image coding was thus hidden under the name of subband coding [8, 16], which builds upon the traditional transform coding paradigm of energy compaction and decorrelation. The main idea of subband coding is to treat different bands differently as each band can be modeled as a statistically distinct process in quantization and coding.

To illustrate the design philosophy of early subband coders, let us again assume for example that we are coding a vector source $\{x_0, x_1\}$, where both x_0 and x_1 are samples of a stationary random sequence $X(n)$ with zero mean and variance σ_x^2. If we code x_0 and x_1 directly by using PCM coding, from our earlier discussion on quantization, the R-D performance can be approximated as

$$D_{\mathrm{PCM}}(R) = h\sigma_x^2 2^{-2R}. \tag{3}$$

In subband coding, two quantizers are designed: one for each of the two transform coefficients y_0 and y_1. The goal is to choose rates R_0 and R_1 needed for coding y_0 and y_1 so that the average distortion

$$D_{\mathrm{SBC}}(R) = (D(R_0) + D(R_1))/2 \tag{4}$$

is minimized with the constraint on the average bit rate

$$(R_0 + R_1)/2 = R. \tag{5}$$

Using the high rate approximation, we write $D(R_0) = h\sigma_{y_0}^2 2^{-2R_0}$ and $D(R_1) = h\sigma_{y_1}^2 2^{-2R_1}$; then the solutions to this *bit allocation* problem are [8]

$$R_0 = R + \frac{1}{2}\log_2 \frac{\sigma_{y_0}}{\sigma_{y_1}}; \quad R_1 = R - \frac{1}{2}\log_2 \frac{\sigma_{y_0}}{\sigma_{y_1}}, \tag{6}$$

with the minimum average distortion being

$$D_{\mathrm{SBC}}(R) = h\sigma_{y_0}\sigma_{y_1} 2^{-2R}. \tag{7}$$

Note that, at the optimal point, $D(R_0) = D(R_1) = D_{\mathrm{SBC}}(R)$. That is; the quantizers for y_0 and y_1 give the same distortion with optimal bit allocation. Since the transform T is orthogonal, we have $\sigma_x^2 = (\sigma_{y_0}^2 + \sigma_{y_1}^2)/2$. The *coding gain* of using subband coding over PCM is

$$\frac{D_{\mathrm{PCM}}(R)}{D_{\mathrm{SBC}}(R)} = \frac{\sigma_x^2}{\sigma_{y_0}\sigma_{y_1}} = \frac{(\sigma_{y_0}^2 + \sigma_{y_1}^2)/2}{(\sigma_{y_0}^2\sigma_{y_1}^2)^{1/2}}, \tag{8}$$

the ratio of arithmetic mean to geometric mean of coefficient variances $\sigma_{y_0}^2$ and $\sigma_{y_1}^2$. What this important result states is that subband coding performs no worse than PCM coding, and that the larger the disparity between coefficient variances, the bigger the subband coding gain, because $(\sigma_{y_0}^2 + \sigma_{y_1}^2)/2 \geq (\sigma_{y_0}^2\sigma_{y_1}^2)^{1/2}$, with equality if $\sigma_{y_0}^2 = \sigma_{y_1}^2$. This result can be easily extended to the case when $M > 2$ uniform subbands (of equal size) are used instead. The coding gain in this general case is

$$\frac{D_{\mathrm{PCM}}(R)}{D_{\mathrm{SBC}}(R)} = \frac{\frac{1}{M}\sum_{k=0}^{M-1}\sigma_k^2}{\left(\prod_{k=0}^{M-1}\sigma_k^2\right)^{1/M}}, \tag{9}$$

where σ_k^2 is the sample variance of the kth band ($0 \leq k \leq M-1$). The above assumes that all M bands are of the same size. In the case of the subband or wavelet transform, the sizes of the subbands are not the same (see Fig. 8 below), but the above formula can be generalized pretty easily to account

[4]Both wavelet image coding and subband coding are special cases of transform coding.

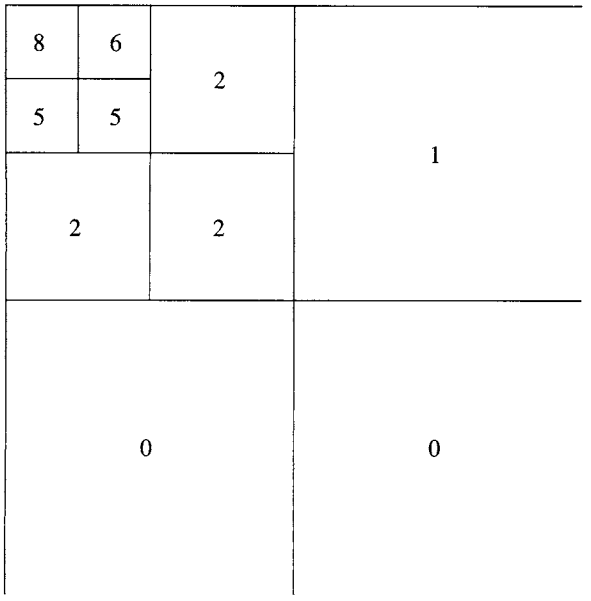

Figure 8 Typical bit allocation results for different subbands. The unit of the numbers is bits per pixel. These are designed to satisfy a total bit rate budget of one bit per pixel. That is: $\{[(8+6+5+5)/4+2+2+2]/4+1+0+0\}/4=1$.

for this. As another extension of the results given in the above example, it can be shown that the necessary condition for optimal bit allocation is that all subbands should incur the same distortion at optimality—else it is possible to steal some bits from the lower-distortion bands to the higher-distortion bands in a way that makes the overall performance better.

Figure 8 shows typical bit allocation results for different subbands under a total bit rate budget of 1 bit per pixel for wavelet image coding. Since low frequency bands in the upper-left corner have far more energy than high frequency bands in the lower-right corner (see Fig. 1), more bits have to be allocated to lowpass bands than to highpass bands. The last two frequency bands in the bottom half are not coded (set to zero) because of limited bit rate. Since subband coding treats wavelet coefficients according to their frequency bands, it is effectively a *frequency* domain transform technique.

Initial wavelet-based coding algorithms, e.g., [17], followed exactly this subband coding methodology. These algorithms were designed to exploit the energy compaction properties of the wavelet transform only in the frequency domain by applying quantizers optimized for the statistics of each frequency band. Such algorithms have demonstrated small improvements in coding efficiency over standard transform-based algorithms.

5 New and More Efficient Class of Wavelet Coders

Because wavelet decompositions offer *space-frequency* representations of images, i.e., low frequency coefficients have large spatial support (good for representing large image background regions), whereas high frequency coefficients have small spatial support (good for representing spatially local phenomena such as edges), the wavelet representation calls for new quantization strategies that go beyond traditional subband coding techniques to exploit this underlying space-frequency image characterization.

Shapiro made a breakthrough in 1993 with his embedded zerotree wavelet (EZW) coding algorithm [18]. Since then a new class of algorithms have been developed that achieve significantly improved performance over the EZW coder. In particular, Said and Pearlman's work on set partitioning in hierarchic trees (SPIHT) [19], which improves the EZW coder, has established zerotree techniques as the current state-of-the-art of wavelet image coding since the SPIHT algorithm proves to be very successful for both lossy and lossless compression.

5.1 Zerotree-based Framework and EZW Coding

A wavelet image representation can be thought of as a tree-structured spatial set of coefficients. *A wavelet coefficient tree* is defined as the set of coefficients from different bands that represent the same spatial region in the image. Figure 9 shows a three-level wavelet decomposition of the *Lena* image, together with a wavelet coefficient tree structure representing the eye region of *Lena*. Arrows in Fig. 9(b) identify the parent-children dependencies in a tree. The lowest frequency band of the decomposition is represented by the root nodes (top) of the tree, the highest frequency bands by the leaf nodes (bottom) of the tree, and each parent node represents a lower frequency component than its children. Except for a root node, which has only three children nodes, each parent node has four children nodes, the 2×2 region of the same spatial location in the immediately higher frequency band.

Both the EZW and SPIHT algorithms [18, 19] are based on the idea of using multipass zerotree coding to transmit the largest wavelet coefficients (in magnitude) at first. We hereby use "zero coding" as a generic term for both schemes, but we focus on the popular SPIHT coder because of its superior performance. A set of tree coefficients is significant if the largest coefficient magnitude in the set is greater than or equal to a certain threshold (e.g., a power of 2); otherwise, it is insignificant. Similarly, a coefficient is significant if its magnitude is greater than or equal to the threshold; otherwise, it is insignificant. In each pass the significance of a larger set in the tree is tested at first: if the set is insignificant, a binary "zerotree" bit is used to set all coefficients in the set to zero; otherwise, the set is partitioned into subsets (or child sets) for further significance tests. After all coefficients are tested in one pass, the threshold is halved before the next pass.

The underlying assumption of the zerotree coding framework is that most images can be modeled as having decaying

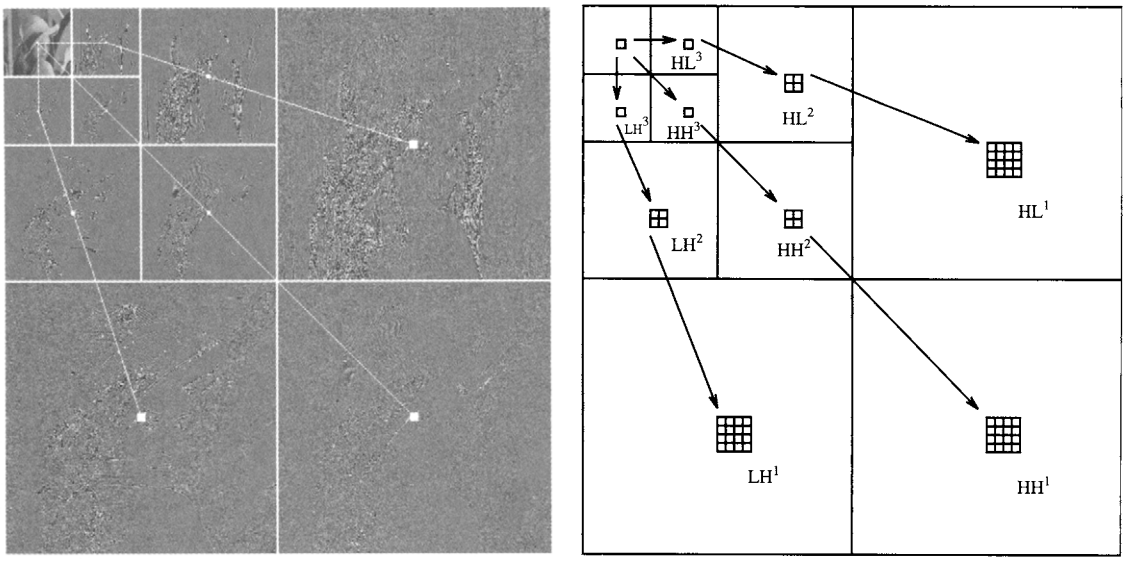

Figure 9 Wavelet decomposition offers a tree-structured image representation. (a) Three-level wavelet decomposition of the *Lena* image. (b) Spatial wavelet coefficient tree consisting of coefficients from different bands that correspond to the same spatial region of the original image (e.g., the eye of *Lena*). Arrows identify the parent-children dependencies.

power spectral densities. That is, if a parent node in the wavelet coefficient tree is insignificant, it is very likely that its descendents are also insignificant. The zerotree symbol is used very efficiently in this case to signify a spatial subtree of zeros.

We give a SPIHT coding example to highlight the order of operations in zerotree coding. Start with a simple three-level wavelet representation of an 8×8 image,[5] as shown in Fig. 10. The largest coefficient magnitude is 63. We can choose the threshold in the first pass between 31.5 and 63. Let $T_1 = 32$. Table 1 shows the first pass of the SPIHT coding process, with the following comments:

1. The coefficient value 63 is greater than the threshold 32 and positive, so a significance bit "1" is generated, followed by a positive sign bit "0". After decoding these symbols, the decoder knows the coefficient is between 32 and 64 and uses the midpoint 48 as an estimate.[6]
2. The descendant set of coefficient -34 is significant; a significance bit "1" is generated, followed by significance test of each of its four children $\{49, 10, 14, -13\}$.
3. The descendant set of coefficient -31 is significant; a significance bit "1" is generated, followed by significance test of each of its four children $\{15, 14, -9, -7\}$.
4. The descendant set of coefficient 23 is insignificant; an insignificance bit "0" is generated. This zerotree bit is the only symbol generated in the current pass for the whole descendant set of coefficient 23.
5. The grandchild set of coefficient -34 is insignificant; a binary bit "0" is generated.[7]
6. The grandchild set of coefficient -31 is significant; a binary bit "1" is generated.
7. The descendant set of coefficient 15 is insignificant; an insignificance bit "0" is generated. This zerotree bit is the only symbol generated in the current pass for the whole descendant set of coefficient 15.
8. The descendant set of coefficient 14 is significant; a significance bit "1" is generated, followed by significance test of each of its four children $\{-1, 47, -3, 2\}$.
9. Coefficient -31 has four children $\{15, 14, -9, -7\}$. Descendant sets of child 15 and child 14 were tested for significance before. Now descendant sets of the remaining two children -9 and -7 are tested.

In this example, the encoder generates 29 bits in the first pass. Along the process, it identifies four significant coefficients $\{63, -34, 49, 47\}$. The decoder reconstructs each coefficient based on these bits. When a set is insignificant, the decoder knows each coefficient in the set is between -32 and 32 and uses the midpoint 0 as an estimate. The reconstruction result at the end of the first pass is shown in Fig. 11(a).

[5]This set of wavelet coefficients is the same as the one used by Shapiro in an example to showcase EZW coding [18]. Curious readers can compare these two examples to see the difference between EZW and SPIHT coding.

[6]The reconstruction value can be anywhere in the uncertainty interval [32,64]. Choosing the midpoint is the result of a simple form of minimax estimation.

[7]In this example, we use the following convention: when a coefficient or set is significant, a binary bit "1" is generated; otherwise, a binary bit "0" is generated. In the actual SPIHT implementation [19], this convention was not always followed—when a grandchild set is significant, a binary bit "0" is generated, otherwise, a binary bit "1" is generated.

The threshold is halved ($T_2 = T_1/2 = 16$) before the second pass, where insignificant coefficients and sets in the first pass are tested for significance again against T_2, and significant coefficients found in the first pass are refined. The second pass thus consists of the following:

1. Significance tests of the 12 insignificant coefficients found in the first pass—those having reconstruction value 0 in Table 1. Coefficients −31 at (0, 1) and 23 at (1, 1) are found to be significant in this pass; a sign bit is generated for each. The decoder knows the coefficient

63	-34	49	10	7	13	-12	7
-31	23	14	-13	3	4	6	-1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4-	2	3	2
-5	9	-1	47	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

Figure 10 Example of a three-level wavelet representation of an 8 × 8 image.

TABLE 1 First pass of the SPIHT coding process at threshold $T_1 = 32$.

Coefficient Coordinates	Coefficient Value	Binary Symbol	Reconstruction Value	Comments
(0,0)	63	1		(1)
		0	48	
(1,0)	−34	1		
		1	−48	
(0,1)	−31	0	0	
(1,1)	23	0	0	
(1,0)	−34	1		(2)
(2,0)	49	1		
		0	48	
(3,0)	10	0	0	
(2,1)	14	0	0	
(3,1)	−13	0	0	
(0,1)	−31	1		(3)
(0,2)	15	0	0	
(1,2)	14	0	0	
(0,3)	−9	0	0	
(1,3)	−7	0	0	
(1,1)	23	0		(4)
(1,0)	−34	0		(5)
(0,1)	−31	1		(6)
(0,2)	15	0		(7)
(1,2)	14	1		(8)
(2,4)	−1	0	0	
(3,4)	47	1		
		0	48	
(2,5)	−3	0	0	
(3,5)	2	0	0	
(0,3)	−9	0		(9)
(1,3)	−7	0		

48	-48	48	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	48	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(a)

56	-40	56	0	0	0	0	0
-24	24	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	40	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(b)

Figure 11 Reconstructions after the (a) first and (b) second passes in SPIHT coding.

magnitude is between 16 and 32 and decode them as −24 and 24.

2. The descendant set of coefficient 23 at (1, 1) is insignificant; so are the grandchild set of coefficient 49 at (2, 0) and descendant sets of coefficients 15 at (0, 2), −9 at (0, 3), and −7 at (1, 3). A zerotree bit is generated in the current pass for each insignificant descendant set.
3. Refinement of the four significant coefficients {63, −34, 49, 47} found in the first pass. The coefficient magnitudes are identified as being either between 32 and 48, which will be encoded with "0" and decoded as the midpoint 40, or between 48 and 64, which will be encoded with "1" and decoded as 56.

The encoder generates 23 bits (14 from step 1, five from step 2, and four from step 3) in the second pass. Along the process it identifies two more significant coefficients. Together with the four found in the first pass, the set of significant coefficients now becomes {63, −34, 49, 47, −31, 23}. The reconstruction result at the end of the second pass is shown in Fig. 11(b).

The above encoding process continues from one pass to another and can stop at any point. For better coding performance, arithmetic coding [15] can be used to further compress the binary bitstream out of the SPIHT encoder.

From this example, we note that when the thresholds are powers of 2, zerotree coding can be thought of as a bit-plane coding scheme. It encodes one bit-plane at a time, starting from the most significant bit. The effective quantizer in each pass is a deadzone quantizer with the deadzone being twice the uniform step size. With the sign bits and refinement bits (for coefficients that become significant in previous passes) being coded on the fly, zerotree coding generates an *embedded* bitstream, which is highly desirable for progressive transmission (see Fig. 4). A simple example of embedded representation is the approximation of an irrational number (say $\pi = 3.1415926535\cdots$) by a rational number. If we were only allowed two digits after the decimal point, then $\pi \approx 3.14$; if three digits after the decimal point were allowed, then $\pi \approx 3.141$; and so on. Each additional bit of the embedded bitstream is used to improve upon the previously decoded image for successive approximation, so rate control in zerotree coding is exact and no loss is incurred if decoding stops at any point of the bitstream. The remarkable thing about zerotree coding is that it outperforms almost all other schemes (such as JPEG coding) while being embedded. This good performance can be partially attributed to the fact that zerotree coding captures across-scale interdependencies of wavelet coefficients. The zerotree symbol effectively zeros out a set of coefficients in a subtree, achieving the coding gain of vector quantization [7] over scalar quantization.

Figure 12 shows the original *Lena* and *Barbara* images and their decoded versions at 0.25 bit per pixel (32:1 compression ratio) by baseline JPEG and SPIHT [19]. These images are coded at a relatively low bit rate to emphasize coding artifacts. The *Barbara* image is known to be hard to compress because of its insignificant high frequency content (see the periodic stripe texture on *Barbara*'s trousers and scarf, and the checkerboard texture pattern on the tablecloth). The subjective difference in reconstruction quality between the two decoded versions of the same image is quite perceptible on a high-resolution monitor. The JPEG decoded images show highly visible blocking artifacts while the wavelet-based SPIHT decoded images have much sharper edges and preserve most of the striped texture.

5.2 Advanced Wavelet Coders: High Level Characterization

We saw that the main difference between the early class of subband image coding algorithms and the zerotree-based compression framework is that the former exploits only the frequency characterization of the wavelet image representation, whereas the latter exploits both the *spatial* and *frequency* characterization. To be more precise, the early class of coders were adept at exploiting the wavelet transform's ability to concentrate the image energy disparately in the different frequency bands, with the lower frequency bands having a much higher energy density. What these coders fail to exploit was the very definite spatial characterization of the wavelet representation. In fact, this is even apparent to the naked eye if one views the wavelet decomposition of the *Lena* image in Fig. 1, where the spatial structure of the image is clearly exposed in the high frequency wavelet bands, e.g., the edge structure of the hat and face and the feather texture, etc. Failure to exploit this spatial structure limited the performance potential of the early subband coders.

In explicit terms, not only is it true that the energy density of the different wavelet subbands is highly disparate, resulting in gains by separating the data set into statistically dissimilar frequency groupings of data, but it is also true that the data in the high frequency subbands are highly spatially structured and clustered around the spatial edges of the original image. The early class of coders exploited the conventional coding gain associated with dissimilarity in the statistics of the frequency bands, but not the potential coding gain from separating individual frequency band energy into spatially localized clusters.

It is insightful to note that unlike the coding gain based on the frequency characterization, which is statistically predictable for typical images (the low frequency subbands have much higher energy-density than the high frequency ones), there is a difficulty in going after the coding gain associated with the spatial characterization that is not statistically predictable; after all, there is no reason to expect the upper left corner of the image to have more edges than the lower right. This calls for a drastically different way of exploiting

Figure 12 Coding of the *512 × 512 Lena* and *Barbara* images at 0.25 bit per pixel (compression ratio of 32:1). Top: the original *Lena* and *Barbara* images. Middle: baseline JPEG decoded images, PSNR=31.6 dB for *Lena*, and PSNR=25.2 dB for *Barbara*. Bottom: SPIHT decoded images, PSNR=34.1 dB for *Lena*, and PSNR = 27.6 dB for *Barbara*.

this structure—a way of pointing to the *spatial location* of significant edge regions within each subband. At a high level, a zerotree is no more than an efficient "pointing" data-structure that incorporates the spatial characterization of wavelet coefficients by identifying tree-structured collections of insignificant spatial subregions across hierarchic subbands.

Equipped with this high-level insight, it becomes clear that the zerotree approach is but only one way to skin the cat. Researchers in the wavelet image compression community have found other ways to exploit this phenomenon by using an array of creative ideas. The array of successful data-structures in the research literature include: (a) R-D optimized

zerotree-based structures, (b) morphology- or region-growing-based structures, (c) spatial context modeling—based structures, (d) statistical mixture modeling based structure, (e) classification-based structures, and so on. As the details of these advanced methods are beyond the intended scope of this chapter, we refer the reader to "Wavelet image coding: PSNR results" (www.icsl.ucla.edu/~ipl/psnr_results.html) on the World Wide Web for the latest results [20] on wavelet image coding.

6 Adaptive Wavelet Transforms: Wavelet Packets

In noting how transform coding has become the *de facto* standard for image and video compression, it is important to realize that the traditional approach of using a transform with fixed frequency resolution (be it the logarithmic wavelet transform or the DCT) is good only in an ensemble sense for a typical *statistical* class of images. This class is well-suited to the characteristics of the chosen fixed transform. This raises the natural question; is it possible to do better by being *adaptive* in the transformation so as to best match the features of the transform to the specific attributes of arbitrary individual images that may not belong to the typical ensemble?

To be specific, the wavelet transform is a good fit for typical natural images that have an exponentially decaying spectral density, with a mixture of strong stationary low frequency components (such as the image background) and perceptually important short-duration high frequency components (such as sharp image edges). The fit is good because of the wavelet transform's logarithmic decomposition structure, which results in its well-advertised attributes of good frequency resolution at low frequencies, and good time resolution at high frequencies (see Fig. 3(b)).

There are, however, important classes of images (or significant subimages) whose attributes go against those offered by the wavelet decomposition, e.g., images having strong highpass components. A good example is the periodic texture pattern in the *Barbara* image of Fig. 12—see the trousers and scarf textures, as well as the tablecloth texture. Another special class of images for which the wavelet is not a good idea is the class of fingerprint images (see Fig. 13 for a typical example) which has periodic high frequency ridge patterns. These images are better matched with decomposition elements that have good frequency localization at high frequencies (corresponding to the texture patterns), which the wavelet decomposition does not offer in its menu.

This motivates the search for alternative transform descriptions that are more adaptive in their representation, and that are more robust to a large class of images of unknown or mismatched space-frequency characteristics. Although the task of finding an optimal decomposition for every individual image in the world is an ill-posed problem, the situation gets more interesting if we consider a large but finite library of desirable transforms, and match the best transform in the library adaptively to the individual image. In order to make this feasible, there are two requirements. First, the library must contain a good representative set of entries (e.g., it would be good to include the conventional wavelet decomposition). Second, it is essential that there exists a fast way of searching through the library to find the best transform in an image-adaptive manner.

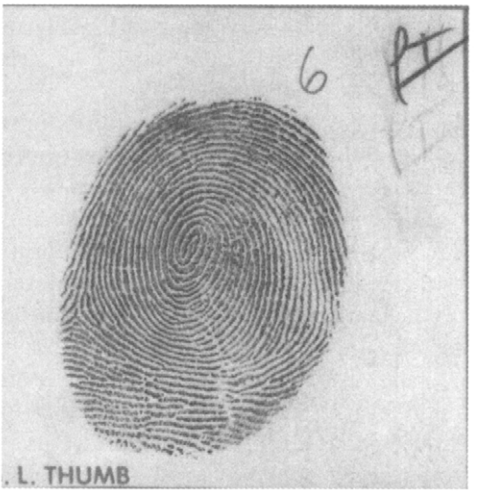

Figure 13 *Fingerprint* image: image coding using logarithmic wavelet transform does not perform well for fingerprint images such as this one with strong highpass ridge patterns.

Both these requirements are met with an elegant generalization of the wavelet transform, called the *wavelet packet* decomposition, also known sometimes as the *best basis* framework. Wavelet packets were introduced to the signal processing community by Coifman and Wickerhauser in [21]. They represent a huge library of orthogonal transforms having a rich time-frequency diversity that also come with an easy-to-search capability, thanks to the existence of fast algorithms that exploit the tree-structured nature of these basis expansions—the tree-structure comes from the cascading of multirate filter bank operations; see Chapter 4.2 and [3]. Wavelet packet bases essentially look like wavelet bases shown in Figure 3(b), but they have more oscillations.

The wavelet decomposition, which corresponds to a logarithmic tree structure, is the most famous member of the wavelet packet family. Whereas wavelets are best matched to signals having a decaying energy spectrum, wavelet packets can be matched to signals having almost arbitrary spectral profiles, such as signals having strong high-frequency or midfrequency stationary components, making them attractive for decomposing images having significant texture patterns, as discussed earlier. There are an astronomical number of basis choices available in the typical wavelet packet library: for example, it can be shown that the library has over 10^{78} transforms for typical five-level 2D wavelet packet image

decompositions. The library is thus well-equipped to deal efficiently with arbitrary classes of images requiring diverse spatial-frequency resolution tradeoffs.

Using the concept of time-frequency tilings introduced in Section 1, it is easy to see what wavelet packet tilings look like, and how they are a generalization of wavelets. We again start with 1D signals. Tiling representations of several expansions are plotted in Fig. 14. Figure 14(a) shows a uniform STFT-like expansion, where the tiles are all of the same shape and size; Fig. 14(b) is the familiar wavelet expansion or the logarithmic subband decomposition; Figure 14(c) shows a wavelet packet expansion where the bandwidths of the bases are neither uniformly nor logarithmically varying; and Fig. 14(d) highlights a wavelet packet expansion where the time-frequency attributes are exactly the reverse of the wavelet case: the expansion has good frequency resolution at higher frequencies, and good time localization at lower frequencies: we might call this the "anti-wavelet" packet. There are a plethora of other options for the time-frequency resolution tradeoff, and these all correspond to admissible wavelet packet choices.

The extra adaptivity of the wavelet packet framework is obtained at the price of added computation in searching for the best wavelet packet basis, so an efficient fast search algorithm is the key in applications involving wavelet packets. The problem of searching for the best basis from the wavelet packet library for the compression problem using an R-D optimization framework and a fast tree-pruning algorithm was described in [22].

The 1D wavelet packet bases can be easily extended to 2D by writing a 2D basis function as the product of two 1D basis functions. In another words, we can treat the rows and columns of an image separately as 1D signals. The performance gains associated with wavelet packets are obviously image-dependent. For difficult images such as *Barbara* in Fig. 12, a wavelet packet decomposition shown in Fig. 15(a) gives much better coding performance than the wavelet decomposition. The wavelet packet decoded *Barbara* image at 0.1825 b/p is shown in Figure 15(b), whose visual quality (or PSNR) is the same as the wavelet SPIHT decoded *Barbara* image at 0.25 b/p in Fig. 12. The bit rate saving achieved by using a wavelet packet basis instead of the wavelet basis in this case is 27% at the same visual quality.

An important practical application of wavelet packet expansions is the FBI wavelet scalar quantization (WSQ) standard

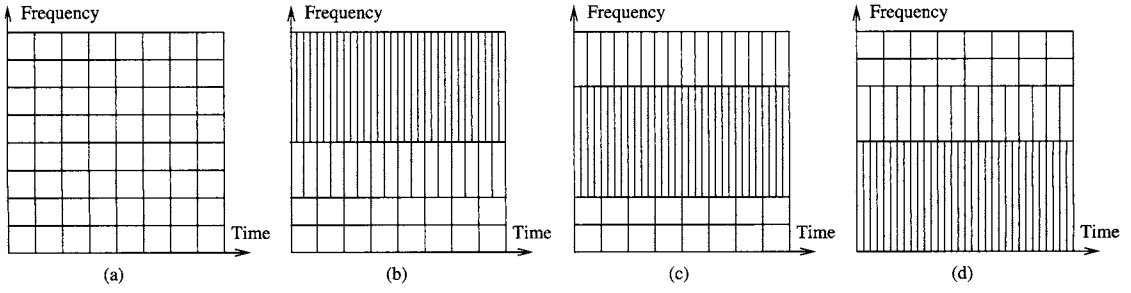

Figure 14 Tiling representations of several expansions for 1D signals. (a) STFT-like decomposition, (b) wavelet decomposition, (c) wavelet packet decomposition, and (d) "anti-wavelet" packet decomposition.

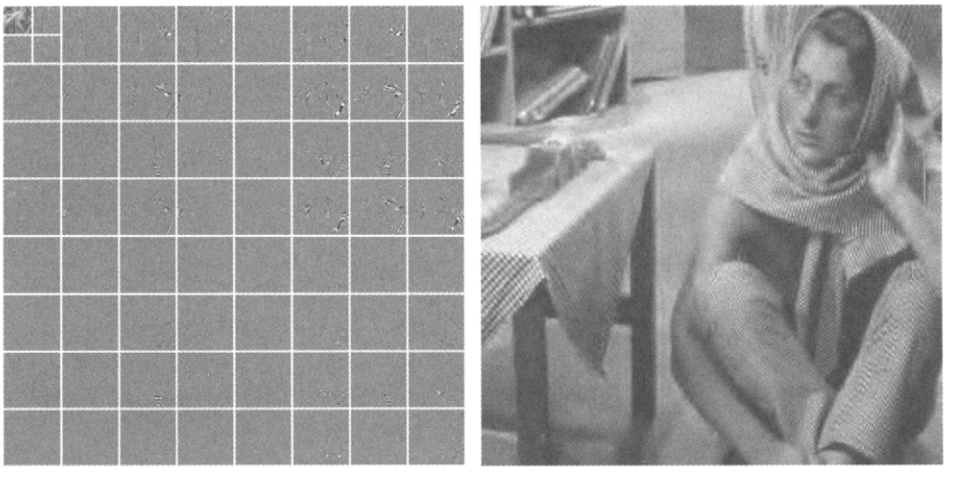

Figure 15 (a) A wavelet packet decomposition for the *Barbara* image. White lines represent frequency boundaries. High pass bands are processed for display. (b) Wavelet packet decoded *Barbara* at 0.1825 b/p. PSNR=27.6 dB.

for fingerprint image compression [23]. Because of the complexity associated with adaptive wavelet packet transforms, the FBI WSQ standard uses a fixed wavelet packet decomposition in the transform stage. The transform structure specified by the FBI WSQ standard is shown in Fig. 16. It was designed for 500 dots per inch fingerprint images by spectral analysis and trial-and-error. A total of 64 subbands are generated with a five-level wavelet packet decomposition. Trials by the FBI have shown that the WSQ standard benefited from having fine frequency partitions in the middle frequency region containing the fingerprint ridge patterns.

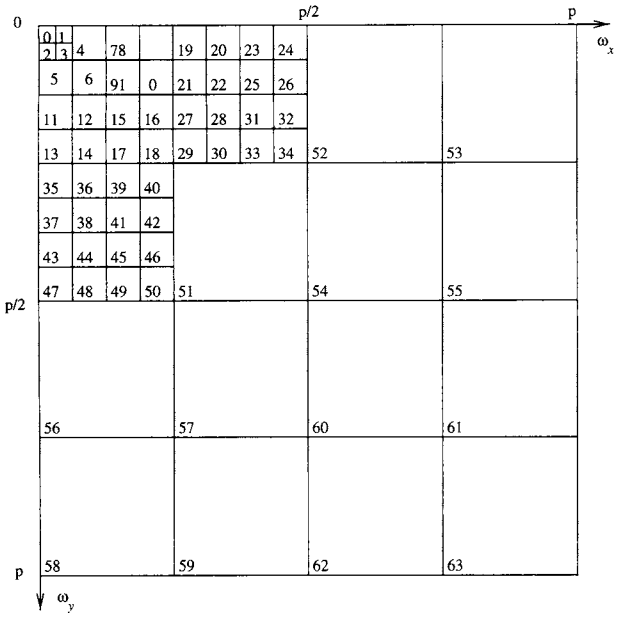

Figure 16 The wavelet packet transform structure given in the FBI WSQ specification. The number sequence shows the labeling of the different subbands.

As an extension of adaptive wavelet packet transforms, one can introduce time-variation by segmenting the signal in time and allowing the wavelet packet bases to evolve with the signal. The result is a time-varying transform coding scheme that can adapt to signal nonstationarities. Computationally fast algorithms are again very important for finding the optimal signal expansions in such a time-varying system. For 2D images, the simplest of these algorithms performs adaptive frequency segmentations over regions of the image selected through a quadtree decomposition. More complicated algorithms provide combinations of frequency decomposition and spatial segmentation. These jointly adaptive algorithms work particularly well for highly nonstationary images. Figure 17 shows the space-frequency tree segmentation and tiling for the *Building* image [24]. The image to the left shows the spatial segmentation result that separates the sky in the background from the building and the pond in the foreground. The image to the right gives the best wavelet packet decomposition for each spatial segment.

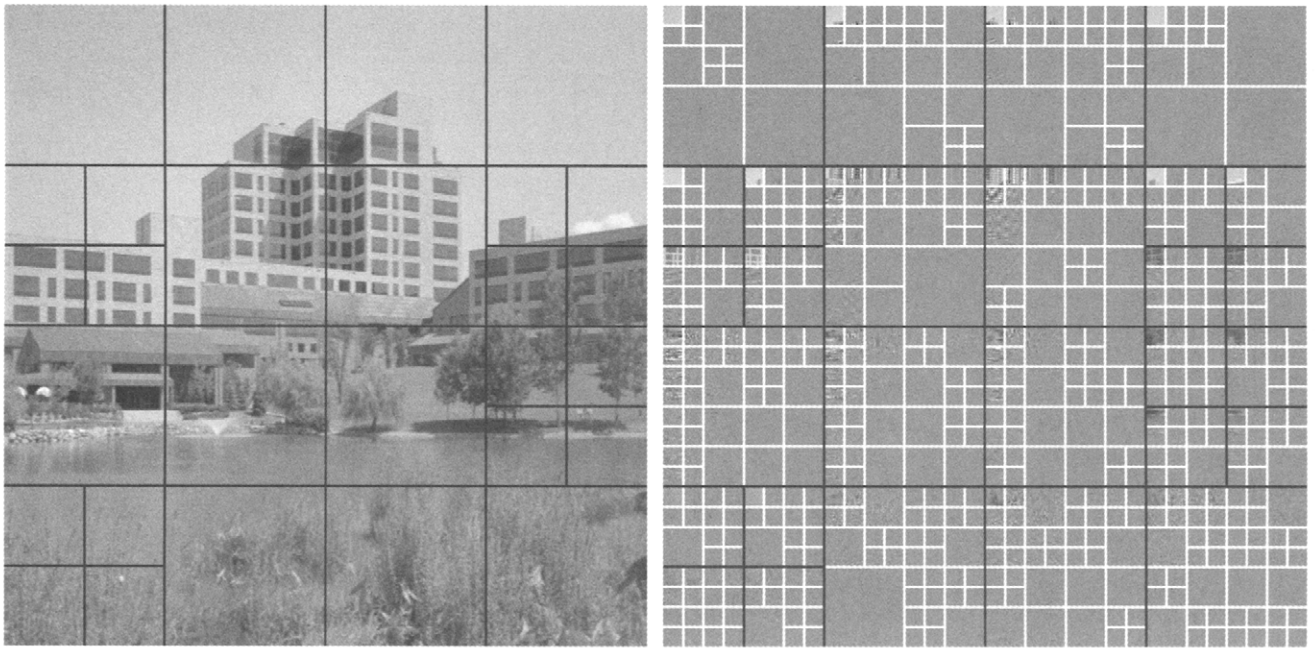

Figure 17 Space-frequency segmentation and tiling for the *Building* image. The image to the left shows that spatial segmentation separates the sky in background from the building and the pond in the foreground. The image to the right gives the best wavelet packet decomposition of each spatial segment. Dark lines represent spatial segments; white lines represent subband boundaries of wavelet packet decompositions. Note that the upper-left corners are the low pass bands of wavelet packet decompositions.

7 JPEG2000 and Recent Developments

JPEG2000 by default employs the dyadic wavelet transform for natural images in many standard applications. It also allows the choice of the more general wavelet packet transforms for certain types of imagery (e.g., fingerprints and radar images). Instead of using the zerotree-based SPIHT algorithm, JPEG2000 relies on embedded block coding with optimized truncation (EBCOT) [25] to provide a rich set of features such as quality scalability, resolution scalability, spatial random access and region-of-interest coding. Besides robustness to image type changes in terms of compression performance, the main advantage of the block-based EBCOT algorithm is that it provides easier random access to local image components. On the other hand, both encoding and decoding in SPIHT require non-local memory access to the whole tree of wavelet coefficients, causing reduction in throughput when coding large-size images. A thorough description of the JPEG2000 standard is in [1]. Other JPEG2000 related references are Chapter 5.5 and [26, 27].

Although this chapter is about wavelet coding of 2D images, the wavelet coding framework and its extension to wavelet packets apply to 3D video as well. Recent research works (see [28] and references therein) on 3D scalable wavelet video coders based on the framework of motion-compensated temporal filtering (MCTF) [29] have shown competitive or better performance than the best MC-DCT based standard video coder (e.g., H.264/AVC [30]). They have stirred considerable excitement in the video coding community and stimulated research efforts towards subband/wavelet interframe video coding, especially in the area of scalable motion coding [31] within the context of MCTF. MCTF can be conceptually viewed as the extension of wavelet-based coding in JPEG2000 from 2D images to 3D video. It nicely combines scalability features of wavelet-based coding with motion compensation, which has been proven to be very efficient and necessary in MC-DCT based standard video coders. MPEG is currently exploring a scalable video coding standard based on MCTF. We refer the readers to a recent special issue [32] on the latest results and Chapter 6.2 for an exposition of 3D subband/wavelet video coding.

8 Conclusion

Since the introduction of wavelets as a signal processing tool in the late 1980s, a variety of wavelet-based coding algorithms have advanced the limits of compression performance well beyond that of the current commercial JPEG image coding standard. In this chapter, we have provided very simple high-level insights, based on the intuitive concept of time-frequency representations, into why wavelets are good for image coding. After introducing the salient aspects of the compression problem in general and the transform coding problem in particular, we have highlighted the key important differences between the early class of subband coders and the more advanced class of modern-day wavelet image coders. Selecting the embedded zerotree wavelet coding structure embodied in the celebrated SPIHT algorithm as a representative of this latter class, we have detailed its operation by using a simple illustrative example. We have also described the role of wavelet packets as a simple but powerful generalization of the wavelet decomposition, in order to offer a more robust and adaptive transform image coding framework.

JPEG2000 is the result of the rapid progress made in wavelet image coding research in the 1990s. The triumph of wavelet transform in the evolution of the JPEG2000 standard underlines the importance of the fundamental insights provided in this chapter into why wavelets are so attractive for image compression.

References

[1] D. Taubman and M. Marcellin, *JPEG2000: Image Compression Fundamentals, Standards, and Practice*, Kluwer, 2001.

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

[3] M. Vetterli and J. Kovačević, *Wavelets and Subband Coding*, Prentice-Hall, Englewood Cliffs, NJ, 1995.

[4] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley & Sons, Inc., New York, 1991.

[5] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, 27, 379–423, 623–656, 1948.

[6] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE Nat. Conv. Rec.*, 4, 142–163, March 1959.

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

[8] N. S. Jayant and P. Noll, *Digital Coding of waveforms*, Prentice-Hall, Englewood Clilffs, NJ, 1984.

[9] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. on Info. Theory*, IT-28, 127–135, March 1982.

[10] H. Gish and J. N. Pierce, "Asymptotically efficient quantizing," *IEEE Trans. on Info. Theory*, IT-14, 5, 676–683, September 1968.

[11] M. W. Marcellin and T. R. Fischer, "Trellis coded quantization of memoryless and Gauss-Markov sources," *IEEE Trans. on Comm.*, 38, 1, 82–93, January 1990.

[12] T. Berger, *Rate Distortion Theory*, Prentice-Hall, Englewood Cliffs, NJ, 1971.

[13] N. Farvardin and J. W. Modestino, "Optimum quantizer performance for a class of non-Gaussian memoryless sources," *IEEE Trans. on Info. Theory*, 30, 485–497, May 1984.

[14] D. A. Huffman, "A method for the construction of minimum redundancy codes," *Proc. IRE*, 40, 1098–1101, 1952.

[15] T. C. Bell, J. G. Cleary, and I. H. Witten, *Text Compression*, Prentice-Hall, Englewood Cliffs, New Jersey, 1990.

[16] J. W. Woods, *Subband Image coding*, Kluwer Academic, Boston, MA, 1991.

[17] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. on Image Process.*, 1, 2, 205–220, April 1992.

[18] J. Shapiro, "Embedded image coding using zero-trees of wavelet coefficients," *IEEE Trans. on Signal Process.*, 41, 12, 3445–3462, December 1993.

[19] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. on Circuits Syst. for Video Tech.*, 6, 3, 243–250, June 1996.

[20] University of California at Los Angeles (UCLA) Image Communications Laboratory, "Wavelet image coding: PSNR results," in *http://www.icsl.ucla.edu/~ipl/psnr_results.html.*

[21] R. R. Coifman and M. V. Wickerhauser, "Entropy based algorithms for best basis selection," *IEEE Trans. on Info. Theory*, 32, 712–718, March 1992.

[22] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Trans. on Image Process.*, 2, 2, 160–175, April 1992.

[23] Criminal Justice Information Services, *WSQ Gray-Scale Fingerprint Image Compression Specification (ver. 2.0)*, Federal Bureau of Investigation, February 1993.

[24] K. Ramchandran, Z. Xiong, K. Asai, and M. Vetterli, "Adaptive transforms for image coding using spatially-varying wavelet packets," *IEEE Trans. on Image Process.*, 5, 1197–1204, July 1996.

[25] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Transactions on Image Processing*, 9, 7, 1151–1170, July 2000.

[26] *Special Issue on JPEG2000, Signal Processing: Image Communication*, 17, 1, January 2002.

[27] D. Taubman and M. Marcellin, "JPEG2000: Standard for interactive imaging," *Proc. of the IEEE*, 90, 8, 1336–1357, August 2002.

[28] J. Ohm, M. van der Schaar, and J. Woods, "Interframe wavelet coding – motion picture representation for universal scalability," *Signal Processing: Image Communication*, 19, 9, 877–908, October 2004.

[29] S.-T. Hsiang and J. Woods, "Embedded video coding using invertible motion compensated 3D subband/wavelet filter bank," *Signal Processing: Image Communication*, 16, 8, 705–724, May 2001.

[30] T. Wiegand, G. Sullivan, G. Bjintegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits and Systems for Video Tech.*, 13, 560–576, July 2003.

[31] A. Secker and D. Taubman, "Highly scalable video compression with scalable motion coding," *IEEE Transactions on Image Processing*, 13, 8, 1029–1041, August 2004.

[32] *Special issue on subband/wavelet interframe video coding, Signal Processing: Image Communication*, 19, August 2004.

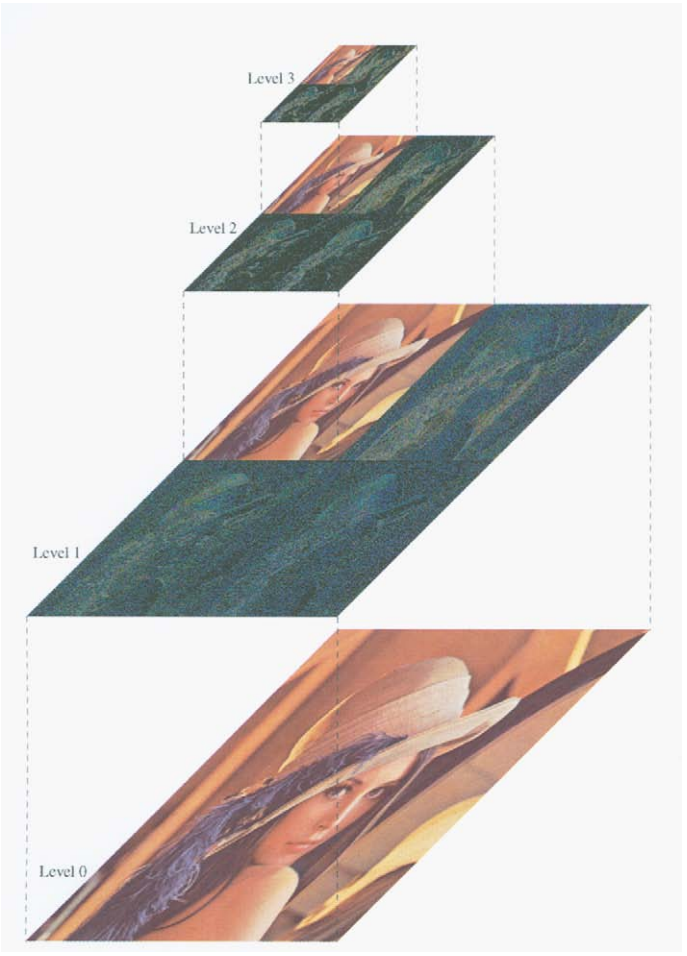

FIGURE 5.4.1 A three-level hierarchy wavelet decomposition of the 512 × 512 color *Lena* image. Level 1 (512 × 512) is the one-level wavelet representation of the original *Lena* at Level 0; Level 2 (256 × 256) shows the one-level wavelet representation of the lowpass image at Level 1; and Level 3 (128×128) gives the one-level wavelet representation of the lowpass image at Level 2.

FIGURE 5.4.2 A three-level wavelet representation of the *Lena* image generated from the top view of the three-level hierarchy wavelet decomposition in Figure 1. It has exactly the same number of samples as in the image domain.

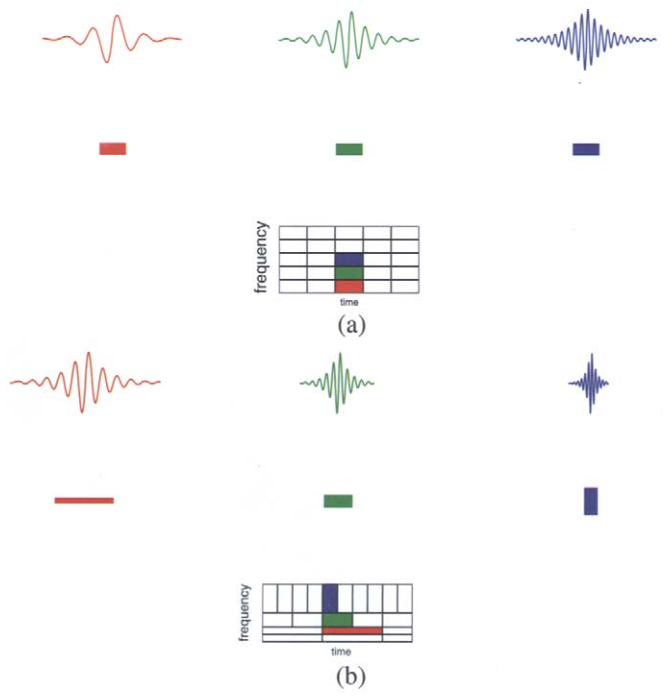

FIGURE 5.4.3 Tiling diagrams associated STFT bases and wavelet bases. (a) STFT bases and the tiling diagram associated with an STFT expansion. STFT bases of different frequencies have the same resolution (or length) in time. (b) Wavelet bases and tiling diagram associated with a wavelet expansion. The time resolution is inversely proportional to frequency for wavelet basis.