Wavelet-based Image Coding: An Overview

Geoffrey M. Davis
Aria Nosratinia

ABSTRACT This paper presents an overview of wavelet-based image coding. We develop the basics of image coding with a discussion of vector quantization. We motivate the use of transform coding in practical settings, and describe the properties of various decorrelating transforms. We motivate the use of the wavelet transform in coding using rate-distortion considerations as well as approximation-theoretic considerations. Finally, we give an overview of current coders in the literature.

1 Introduction

Digital imaging has had an enormous impact on industrial applications and scientific projects. It is no surprise that image coding has been a subject of great commercial interest. The JPEG image coding standard has enjoyed widespread acceptance, and the industry continues to explore its various implementation issues. Efforts are underway to incorporate recent research findings in image coding into a number of new standards, including those for image coding (JPEG 2000), video coding (MPEG-4, MPEG-7), and video teleconferencing (H.263+).

In addition to being a topic of practical importance, the problems studied in image coding are also of considerable theoretical interest. The problems draw upon and have inspired work in information theory, applied harmonic analysis, and signal processing. This paper presents an overview of multiresolution image coding, arguably the most fruitful and successful direction in image coding, in the light of the fundamental principles in probability and approximation theory.

1.1 Image Compression

An image is a positive function on a plane. The value of this function at each point specifies the luminance or brightness of the picture at that point.[1] Digital images are sampled versions of such functions, where the value of the function is specified only at discrete locations on the image plane, known as *pixels*. The value of the luminance at each pixel is represented to a pre-defined precision M. Eight bits of precision for luminance is common in imaging applications. The eight-bit precision is motivated by both the existing computer memory structures (1 byte = 8 bits) as well as the dynamic range of the human eye.

The prevalent custom is that the samples (pixels) reside on a rectangular lattice which we will assume for convenience to be $N \times N$. The brightness value at each pixel is a number between 0 and $2^M - 1$. The simplest binary representation of such an image is a list of the brightness values at each pixel, a list containing N^2M bits. Our standard image example in this paper is a square image with 512 pixels on a side. Each pixel value ranges from 0 to 255, so this canonical representation requires $512^2 \times 8 = 2,097,152$ bits.

Image coding consists of mapping images to strings of binary digits. A good image coder is one that produces binary strings whose lengths are on average much smaller than the original canonical representation of the image. In many imaging applications, exact reproduction of the image bits is not necessary. In this case, one can perturb the image slightly to obtain a shorter representation. If this perturbation is much smaller than the blurring and noise introduced in the formation of the image in the first place, there is no point in using the more accurate representation. Such a coding procedure, where perturbations reduce storage requirements, is known as *lossy coding*. The goal of lossy coding is to reproduce a given image with minimum distortion, given some constraint on the total number of bits in the coded representation.

But why can images be compressed on average? Suppose for example that we seek to efficiently store photographs of all natural scenes. In principle, we can enumerate all such pictures and represent each image by its associated index. Assume we position hypothetical cameras at the vantage point of every atom in the universe (there are roughly 10^{80} of them), and with each of them take pictures in one trillion directions, with one trillion magnifications, exposure settings, and depths of field, and repeat this process one trillion times during each year in the past 10,000 years (once every 0.003 seconds). This will result in a total of 10^{144} images. But $10^{144} \approx 2^{479}$, which means that any image in this enormous ensemble can be represented with only 479 bits, or less than 60 bytes!

This collection includes any image that a modern human eye has ever seen, including artwork, medical images, and so on, because we include pictures of everything in the universe from essentially every vantage point.

[1] Color images are a generalization of this concept, and are represented by a three-dimensional vector function on a plane. In this paper, we do not explicitly treat color images, but most of the results can be directly extended to color images.

And yet the collection can be conceptually represented with a small number of bits. The remaining vast majority of the $2^{512\times512\times8} \approx 10^{600,000}$ possible images in the canonical representation are not of general interest, because they contain little or no structure, and are noise-like.

While the above conceptual exercise is intriguing, it is also entirely impractical. Indexing and retrieval from a set of size 10^{144} is completely out of the question. However, the example illustrates the two main properties that image coders exploit. First, only a small fraction of the possible images in the canonical representation are likely to be of interest. *Entropy coding* can yield a much shorter image representation on average by using short code words for likely images and longer code words for less likely images.[2] Second, in our initial image gathering procedure we sample a continuum of possible images to form a discrete set. The reason we can do so is that most of the images that are left out are visually indistinguishable from images in our set. We can gain additional reductions in stored image size by discretizing our database of images more coarsely, a process called *quantization*. By mapping visually indistinguishable images to the same code, we reduce the number of code words needed to encode images, at the price of a small amount of distortion.

It is possible to quantize each pixel separately, a process known as *scalar quantization*. Quantizing a group of pixels together is known as *vector quantization*, or VQ. Vector quantization can, in principle, capture the maximum compression that is theoretically possible. In Section 2.1 we review the basics of vector quantization, its optimality conditions, and underlying reasons for its powers of compression.

Although VQ is a very powerful theoretical paradigm, it can achieve optimality only asymptotically as its dimensions increase. But the computational cost and delay also grow exponentially with dimensionality, limiting the practicality of VQ. Due to these and other difficulties, most practical coding algorithms have turned to *transform coding* instead of high-dimensional VQ. Transform coding consists of scalar quantization in conjunction with a linear transform. This method captures much of the VQ gain, with only a fraction of the effort. In Section 3, we present the fundamentals of transform coding. We use a second-order model to motivate the use of transform coding, and derive the optimal transform.

The success of transform coding depends on how well the basis functions of the transform represent the features of the signal. At present, One of the most successful representations is the *wavelet transform*, which we present in Section 4.2. One can interpret the wavelet transform as a special case of a subband transform. This view is used to describe the mechanics of a basic wavelet coder in Section 5.

[2] For example, mapping the ubiquitous test image of Lena Sjööblom (see Figure 17) to a one-bit codeword would greatly compress the image coding literature.

There is more to the wavelet transform than this subband transform view, however. The theory underlying wavelets brings to bear a fundamentally different perspective than the frequency-based subband framework. The temporal properties of the wavelet transform have proved particularly useful in motivating some of the most recent coders, which we describe in sections 8 to 9. Finally, in Section 10 we discuss directions of current and future research.

This paper strives to make its subject accessible to a wide audience, while at the same time also portraying the latest developments in multi-resolution image coding. To achieve that end, a fair amount of introductory material is present, which the more advanced reader is encouraged to quickly navigate.

2 Quantization

At the heart of image compression is the idea of quantization and approximation. While the images of interest for compression are almost always in a digital format, it is instructive and more mathematically elegant to treat the pixel luminances as being continuously valued. This assumption is not far from the truth if the original pixel values are represented with a large number of levels.

The role of quantization is to represent this continuum of values with a finite — preferably small — amount of information. Obviously this is not possible without some loss. The quantizer is a function whose set of output values are discrete and usually finite (see Figure 1). Good quantizers are those that represent the signal with a minimum distortion.

Figure 1 also indicates a useful view of quantizers as concatenation of two mappings. The first map, the *encoder*, takes partitions of the x-axis to the set of integers $\{-2, -1, 0, 1, 2\}$. The second, the *decoder*, takes integers to a set of output values $\{\hat{x}_k\}$. We need to define a measure of distortion in order to characterize "good" quantizers. We need to be able to approximate

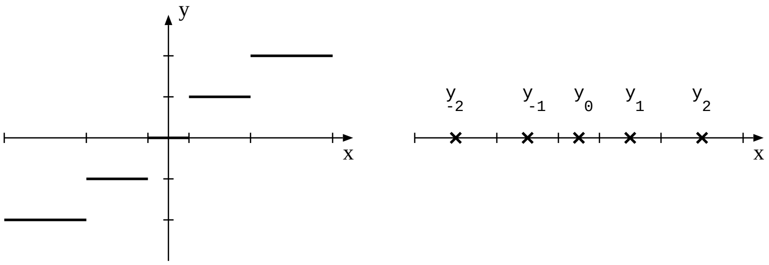


FIGURE 1. *(Left) Quantizer as a function whose output values are discrete. (Right) because the output values are discrete, a quantizer can be more simply represented only on one axis.*

any possible value of x with an output value $\hat{x}_k$. Our goal is to minimize the distortion on average, over all values of x. For this, we need a probabilistic model for the signal values. The strategy is to have few or no reproduction points in locations at which the probability of the signal is negligible, whereas at highly probable signal values, more reproduction points need to be specified. While improbable values of x can still happen — and will be costly — this strategy pays off *on average*. This is the underlying principle behind all signal compression, and will be used over and over again in different guises.

A quantizer is specified by its input partitions and its output reproduction points. It can be shown without much difficulty [1] that an optimal quantizer satisfies the following conditions:

- Given the encoder (partitions), the best decoder is one that puts the reproduction points $\{\hat{x}_i\}$ on the centers of mass of the partitions. This is known as the *centroid condition*
- Given the decoder (reproduction points), the best encoder is one that puts the partition boundaries exactly in the middle of the reproduction points. In other words, each x is grouped with its nearest reproduction point. This is known as the *nearest neighbor condition*.

These concepts extend directly to the case of vector quantization. We will therefore postpone the formal and detailed discussion of quantizer optimality until Section 2.2, where it will be explored in the full generality.

2.1 Vector Quantization

Shannon's source coding theorem [2] imposes theoretical limits on the performance of compression systems. According to this result, under a distortion constraint, the output of a given source cannot be compressed beyond a certain point. The set of optimal rate-distortion pairs form a convex function whose shape is a characteristic of the individual source. Although Shannon's results are not constructive, they do indicate that optimality cannot be achieved unless input data samples are encoded in blocks of increasing length, in other words, as vectors.

Vector quantization (VQ) is the generalization of scalar quantization to the case of a vector. The basic structure of a VQ is essentially the same as scalar quantization, and consists of an encoder and a decoder. The encoder determines a partitioning of the input vector space and to each partition assigns an index, known as a *codeword*. The set of all codewords is known as a *codebook*. The decoder maps the each index to a reproduction vector. Combined, the encoder and decoder map partitions of the space to a discrete set of vectors.

Although vector quantization is an extremely powerful tool, the computational and storage requirements become prohibitive as the dimensionality of the vectors increase. Memory and computational requirements have motivated a wide variety of constrained VQ methods. Among the most prominent are tree structured VQ, shape-gain VQ, classified VQ, multistage VQ, lattice VQ, and hierarchical VQ [1].

There is another important consideration that limits the practical use of VQ in its most general form: The design of the optimal quantizer requires knowledge of the underlying probability density function for the space of images. While we may claim empirical knowledge of lower order joint probability distributions, the same is not true of higher orders. A training set is drawn from the distribution we are trying to quantize, and is used to drive the algorithm that generates the quantizer. As the dimensionality of the model is increased, the amount of data available to estimate the density in each bin of the model decreases, and so does the reliability of the p.d.f. estimate.[3] The issue is commonly known as "the curse of dimensionality".

Instead of accommodating the complexity of VQ, many compression systems opt to move away from it and employ techniques that allow them to use sample-wise or scalar quantization more effectively. In the remainder of this section we discuss properties of optimal vector quantizers and their advantages over scalar quantizers. The balance of the paper will examine ways of obtaining some of the benefits of vector quantizers while maintaining the low complexity of scalar quantizers.

2.2 *Optimal Vector Quantizers*

Optimal vector quantizers are not known in closed form except in a few trivial cases. However, two necessary conditions for optimality provide insights into the structure of these optimal quantizers. These conditions also form the basis of an iterative algorithm for designing quantizers.

Let $p_{\mathbf{X}}(\mathbf{x})$ be the probability density function for the random variable $\mathbf{X}$ we wish to quantize. Let $D(\mathbf{x}, \mathbf{y})$ be an appropriate distortion measure. Like scalar quantizers, vector quantizers are characterized by two operations, an encoder and a decoder. The encoder is defined by a partition of the range of $\mathbf{X}$ into sets $\mathcal{P}_k$. All realizations of $\mathbf{X}$ that lie in $\mathcal{P}_k$ will be encoded to k and decoded to $\hat{\mathbf{x}}_k$. The decoder is defined by specifying the reproduction value $\hat{\mathbf{x}}_k$ for each partition $\mathcal{P}_k$.

A quantizer that minimizes the average distortion D must satisfy the following conditions:

1. *Nearest neighbor condition:* Given a set of reconstruction values $\{\hat{\mathbf{x}}_k\}$, the partition of the values of $\mathbf{X}$ into sets $\mathcal{P}_k$ is the one for which

[3] Most existing techniques do not estimate the p.d.f. to use it for quantization, but rather use the data directly to generate the quantizer. However, the reliability problem is best pictured by the p.d.f. estimation exercise. The effect remains the same with the so-called direct or data-driven methods.

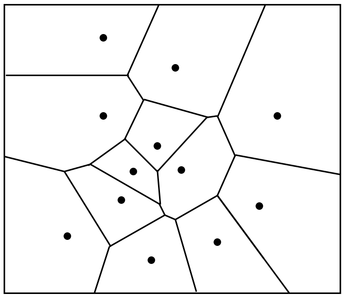


FIGURE 2. *A Voronoi Diagram*

each value $\mathbf{x}$ is mapped by the encoding and decoding process to the nearest reconstruction value. The optimal partitions $\mathcal{P}_k$ given the reconstruction values $\{\hat{\mathbf{x}}_k\}$ are given by

$$\mathcal{P}_k = \{\mathbf{x} : D(\mathbf{x}, \hat{\mathbf{x}}_k) \leq D(\mathbf{x}, \hat{\mathbf{x}}_j) \text{ for } j \neq k\}. \tag{1.1}$$

2. *Centroid condition:* Given a partition of the range of $\mathbf{X}$ into sets $\mathcal{P}_k$, the optimal reconstruction values values $\hat{\mathbf{X}}_k$ are the generalized centroids of the sets $\mathcal{P}_k$. They satisfy

$$\hat{\mathbf{x}}_k = \arg\min \int_{\mathcal{P}_k} p_{\mathbf{X}}(\mathbf{z}) D(\mathbf{z}, \hat{\mathbf{x}}_k) d\mathbf{z}. \tag{1.2}$$

With the squared error distortion, the generalized centroid corresponds to the $p_{\mathbf{X}}(\mathbf{x})$-weighted centroid.

The nearest neighbor condition places constraints on the structure of the partitions. We assume an distance function of the form $D(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|^r$ where the norm is the Euclidean distance. Suppose we have two N-dimensional reconstruction points $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$. Partition $\mathcal{P}_1$ will consist of the points closer to $\hat{\mathbf{x}}_1$ than to $\hat{\mathbf{x}}_2$, and partition $\mathcal{P}_2$ will consist of the points closer to $\hat{\mathbf{x}}_2$ than to $\hat{\mathbf{x}}_1$. These two sets partition $\mathbb{R}^N$ by a hyperplane. Additional reconstruction points result in further partitions of space by hyperplanes. The result is a partition into convex polytopes called *Voronoi cells.* A sample partition of the plane into Voronoi cells is shown in Figure 2.

Vector quantizers can be optimized using an iterative procedure called the Generalized Lloyd algorithm (GLA). This algorithm starts with n initial set of reconstruction values $\{\hat{\mathbf{x}}_k\}_{k=1}^n$. The algorithm proceeds as follows:

1. *Optimize the encoder given the current decoder.* Using the current set of reconstruction values $\{\hat{\mathbf{x}}_k\}$, divide a training set into partitions $\mathcal{P}_k$ according to the nearest neighbor condition. This gives an optimal partitioning of the training data given the reconstruction values.

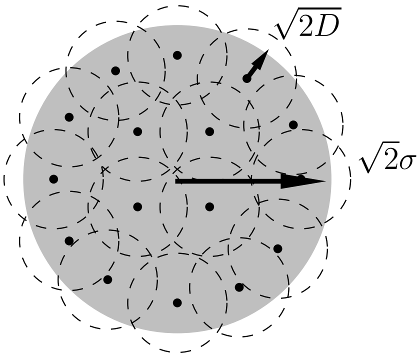


FIGURE 3. *Quantization as a sphere covering problem.*

2. *Optimize the decoder given the current encoder.* Set the reconstruction values $\hat{\mathbf{x}}_k$ to the generalized centroids of the sets $\mathcal{P}_k$. We now have optimal reconstruction values for the sets $\mathcal{P}_k$.

3. If the values $\hat{\mathbf{x}}_k$ have not converged, go to step 1.

The Generalized Lloyd algorithm (GLA) is a descent. Each step either reduces the average distortion or leaves it unchanged. For a finite training set, the distortion can be shown to converge to a fixed value in a finite number of iterations. The GLA does not guarantee a globally optimal quantizer, as there may be other solutions of the necessary conditions that yield smaller distortion. Nonetheless, under mild conditions the algorithm does yield a locally optimal quantizer, meaning that small perturbations in the sets and in the reconstruction values increase the average distortion [3]. The GLA together with stochastic relaxation techniques can be used to obtain globally optimal solutions [4].

2.3 Sphere Covering and Density Shaping

The problem of finding optimal quantizers is closely related to the problem of sphere-covering. An example in 2-D is illustrative. Suppose we want to use R bits per symbol to quantize a vector $\mathbf{X} = (X_1, X_2)$ of independent, identically distributed (i.i.d.) Gaussian random variables with mean zero and variance σ^2. The realizations of $\mathbf{X}$ will have an average length of $\sqrt{2}\sigma$, and most of the realizations of $\mathbf{X}$ will lie inside a circle of radius $\sqrt{2}\sigma$. Our $2R$ bits are sufficient to specify that $\mathbf{X}$ lies in one of 2^{2R} quantizer cells. The goal, then, is to cover a circle of radius $\sqrt{2}\sigma$ with 2^{2R} quantizer cells that have the minimum average distortion.

For the squared error distortion metric, the distortion of the in each partition $\mathcal{P}_k$ is approximately proportional to the second moment of the partition, the integral $\int_{\mathcal{P}_k} (\mathbf{x} - \hat{\mathbf{x}}_k)^2 d\mathbf{x}$. The lowest errors for a given rate are obtained when the ratios of the second moments of the partitions to their volumes is small. Because of the nearest neighbor condition, our partitions will be convex polytopes. We can get a lower bound on the distortion by considering the case of spherical (circular in our 2-D example) partitions, since every convex polytope has a second moment greater than that of a sphere of the same volume.

Figure 3 illustrates this covering problem for $R = 2$. Most realizations of $\mathbf{X}$ lie in the gray circle of radius $\sqrt{2}\sigma$. We want to distribute $2^2R = 16$ reconstruction values so that almost all values of $\mathbf{X}$ are within a distance of $\sqrt{2D}$ of a reconstruction value. The average squared error for $\mathbf{X}$ will be roughly $2D$. Since each $\mathbf{X}$ corresponds to two symbols, the average per-symbol distortion will be roughly D. Because polytopal partitions cover the circle less efficiently than do the circles, this distortion per symbol of D provides a lower limit on our ability to quantize $\mathbf{X}$.[4]

In n dimensions, covering a sphere of radius $\sqrt{n}\sigma$ with 2^{nR} smaller spheres requires that the smaller spheres have a radius of at least $\sqrt{n}\sigma 2^{-R}$. Hence our sphere covering argument suggests that for i.i.d. Gaussian random variables, the minimum squared error possible using R bits per symbol is $D(R) = \sigma^2 2^{-2R}$. A more rigorous argument shows that this is in fact the case [5].

The performance of vector quantizers in n dimensions is determined in part by how closely we can approximate spheres with n-dimensional convex polytopes [6]. When we quantize vector components separately using scalar quantizers, the resulting Voronoi cells are all rectangular prisms, which only poorly approximate spheres. Consider the case of coding 2 uniformly distributed random variables. If we scalar quantize both variables, we subdivide space into squares. A hexagonal partition more effectively approximates a partition of the plane into 2-spheres, and accordingly (ignoring boundary effects), the squared error from the hexagonal partition is 0.962 times that of the square partition for squares and hexagons of equal areas. The benefits of improved spherical approximations increase in higher dimensions. In 100 dimensions, the optimal vector quantizer for uniform densities has an error of roughly 0.69 times that of the optimal scalar quantizer for uniform densities, corresponding to a PSNR gain of 1.6 dB [6].

Fejes Toth [7] has shown that the optimal vector quantizer for a uniform density in 2 dimensions is given by a hexagonal lattice. The problem is unsolved in higher dimensions, but asymptotic results exist. Zador [8] has shown that for the case of asymptotically high quantizer cell densities, the optimal cell density for a random vector $\mathbf{X}$ with density function $p_{\mathbf{X}}(\mathbf{x})$ is

[4] Our estimates of distortion are a bit sloppy in low dimensions, and the per-symbol distortion produced by our circle-covering procedure will be somewhat less than D. In higher dimensions, however, most of a sphere's mass is concentrated in a thin rind just below the surface so we can ignore the interiors of spheres.

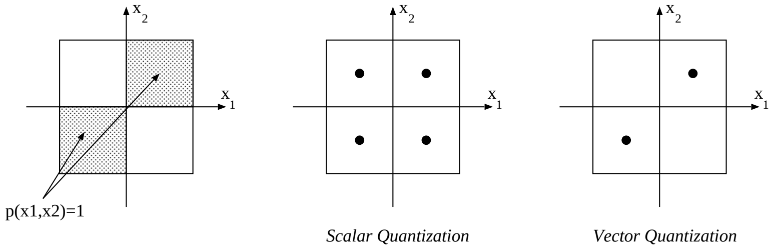


FIGURE 4. *The leftmost figure shows a probability density for a two-dimensional vector* $\mathbf{X}$*. The realizations of* $\mathbf{X}$ *are uniformly distributed in the shaded areas. The center figure shows the four reconstruction values for an optimal scalar quantizer for* $\mathbf{X}$ *with expected squared error* $\frac{1}{12}$*. The figure on the right shows the two reconstruction values for an optimal vector quantizer for* $\mathbf{X}$ *with the same expected error. The vector quantizer requires 0.5 bits per sample, while the scalar quantizer requires 1 bit per sample.*

given by

$$\frac{p_{\mathbf{X}}(\mathbf{x})^{\frac{n}{n+2}}}{\int p_{\mathbf{X}}(\mathbf{y})^{\frac{n}{n+2}} d\mathbf{y}}. \tag{1.3}$$

In contrast, the density obtained for optimal scalar quantization of the marginals of $\mathbf{X}$ is

$$\frac{\prod_k p_{X_k}(x)^{\frac{1}{3}}}{\prod_k \int p_{X_k}(y)^{\frac{1}{3}} dy}, \tag{1.4}$$

where $p_{X_k}(x)$'s are marginal densities for the components of $\mathbf{X}$. Even if the components X_k are independent, the resulting bin density from optimal scalar will still be suboptimal for the vector $\mathbf{X}$. The increased flexibility of vector quantization allows improved quantizer bin density shaping.

2.4 Cross Dependencies

The greatest benefit of jointly quantizing random variables is that we can exploit the dependencies between them. Figure 4 shows a two-dimensional vector $\mathbf{X} = (X_1, X_2)$ that is distributed uniformly over the squares $[0, 1] \times [0, 1]$ and $[-1, 0] \times [-1, 0]$. The marginal densities for X_1 and X_2 are both uniform on $[-1, 1]$. We now hold the expected distortion fixed and compare the cost of encoding X_1 and X_2 as a vector, to the cost of encoding these variables separately. For an expected squared error of $\frac{1}{12}$, the optimal scalar quantizer for both X_1 and X_2 is the one that partitions the interval $[-1, 1]$ into the subintervals $[-1, 0)$ and $[0, 1]$. The cost per symbol is 1 bit, for a total of 2 bits for $\mathbf{X}$. The optimal vector quantizer with the same average distortion has cells that divides the square $[-1, 1] \times [-1, 1]$ in half along the line $y = -x$. The reconstruction values for these two cells are $\hat{\mathbf{x}}_a =$

$(-\frac{1}{2}, -\frac{1}{2})$ and $\hat{\mathbf{x}}_b = (\frac{1}{2}, \frac{1}{2})$. The total cost per vector $\mathbf{X}$ is just 1 bit, only half that of the scalar case.

Because scalar quantizers are limited to using separable partitions, they cannot take advantage of dependencies between random variables. This is a serious limitation, but we can overcome it in part through a preprocessing step consisting of a linear transform. We discuss transform coders in detail in the next section.

2.5 Fractional Bitrates

In scalar quantization, each input sample is represented by a separate codeword. Therefore, the minimum bitrate achievable is one bit per sample, because our symbols cannot be any shorter than one bit. Since each symbol can only have an integer number of bits, the only way to generate fractional bitrates per sample is to code multiple samples at once, as is done in vector quantization. A vector quantizer coding N-dimensional vectors using a K-member codebook can achieve a rate of $(\log_2 K)/N$ bits per sample.

The only way of obtaining the benefit of fractional bitrates with scalar quantization is to process the codewords jointly after quantization. Useful techniques to perform this task include arithmetic coding, run-length coding, and zerotree coding. All these methods find ways to assign symbols to groups of samples, and are instrumental in the effectiveness of image coding. We will discuss these techniques in the upcoming sections.

3 Transform Coding

A great part of the difference in the performance of scalar and vector quantizers is due to VQ's ability to exploit dependencies between samples. Direct scalar quantization of the samples does not capture this redundancy, and therefore suffers. Transform coding allows scalar quantizers to make use of a substantial fraction of inter-pixel dependencies. Transform coders performing a linear pre-processing step that eliminates cross-correlation between samples. Transform coding enables us to obtain some of the benefits of vector quantization with much lower complexity.

To illustrate the usefulness of linear pre-processing, we consider a toy image model. Images in our model consist of two pixels, one on the left and one on the right. We assume that these images are realizations of a two-dimensional random vector $\mathbf{X} = (X_1, X_2)$ for which X_1 and X_2 are identically distributed and jointly Gaussian. The identically distributed assumption is a reasonable one, since there is no *a priori* reason that pixels on the left and on the right should be any different. We know empirically that adjacent image pixels are highly correlated, so let us assume that the

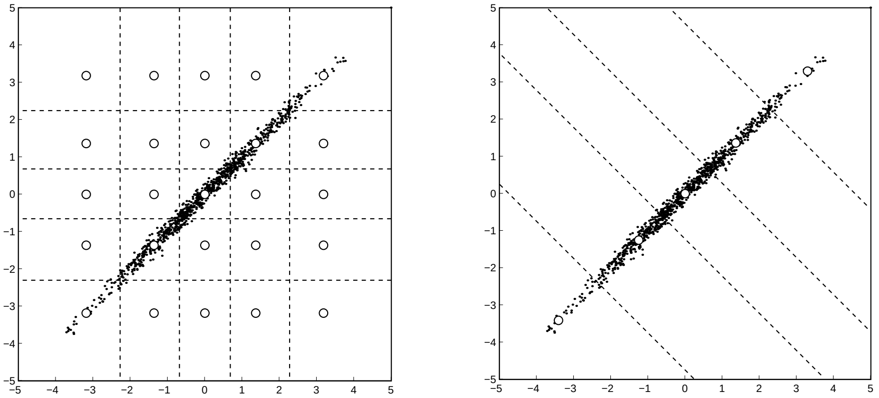


FIGURE 5. *Left: Correlated Gaussians of our image model quantized with optimal scalar quantization. Many reproduction values (shown as white dots) are wasted. Right: Decorrelation by rotating the coordinate axes. Scalar quantization is now much more efficient.*

autocorrelation matrix for these pixels is

$$E[\mathbf{X}\mathbf{X}^T] = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix} \tag{1.5}$$

By symmetry, X_1 and X_2 will have identical quantizers. The Voronoi cells for this scalar quantization are shown on the left in Figure 5. The figure clearly shows the inefficiency of scalar quantization: most of the probability mass is concentrated in just five cells. Thus a significant fraction of the bits used to code the bins are spent distinguishing between cells of very low probability. This scalar quantization scheme does not take advantage of the coupling between X_1 and X_2.

We can remove the correlation between X_1 and X_2 by applying a rotation matrix. The result is a transformed vector $\mathbf{Y}$ given by

$$\mathbf{Y} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \tag{1.6}$$

This rotation does not remove any of the variability in the data. What it does is to pack that variability into the variable Y_1. The new variables Y_1 and Y_2 are independent, zero-mean Gaussian random variables with variances 1.9 and 0.1, respectively. By quantizing Y_1 finely and Y_2 coarsely we obtain a lower average error than by quantizing X_1 and X_2 equally. In the remainder of this section we will describe general procedures for finding appropriate redundancy-removing transforms, and for optimizing related quantization schemes.

3.1 The Karhunen-Loève transform

We can remove correlations between pixels using an orthogonal linear transform called the Karhunen-Loève transform, also known as the Hotelling transform. Let $\mathbf{X}$ be a random vector that we assume has zero-mean and autocorrelation matrix $\mathbf{R}_X$. Our goal is to find a matrix $\mathbf{A}$ such that the components of $\mathbf{Y} = \mathbf{AX}$ will be uncorrelated. The autocorrelation matrix for $\mathbf{Y}$ will be diagonal and is given by $\mathbf{R}_\mathbf{Y} = E(\mathbf{AX})(\mathbf{AX})^T = \mathbf{AR}_\mathbf{X}\mathbf{A}^T$. The matrix $\mathbf{R}_X$ is symmetric and positive semidefinite, hence $\mathbf{A}$ is the matrix whose rows are the eigenvectors of $\mathbf{R}_\mathbf{X}$. We order the rows of $\mathbf{A}$ so that $\mathbf{R}_\mathbf{Y} = \mathrm{diag}(\lambda_\mathbf{0}, \lambda_\mathbf{1}, \ldots, \lambda_\mathbf{N-1})$ where $\lambda_0 \geq \lambda_1 \geq ... \geq \lambda_{N-1} \geq 0$.

The following is a commonly quoted result regarding the optimality of Karhunen-Loève transform.

Theorem 1 *Suppose that we truncate a transformed random vector* $\mathbf{AX}$*, keeping* m *out of the* N *coefficients and setting the rest to zero, Then among all linear transforms, the Karhunen-Loève transform provides the best approximation in the mean square sense to the original vector.*

Proof: We first express the process of forming a linear approximation to a random vector $\mathbf{X}$ from m transform coefficients as a set of matrix operations. We write the transformed version of $\mathbf{X}$ as

$$\mathbf{Y} = \mathbf{U}\,\mathbf{X}.$$

We multiply $\mathbf{Y}$ by a matrix $\mathbf{I}_m$ that retains the first m components of $\mathbf{Y}$ and sets to zero the last $N - m$ components.

$$\hat{\mathbf{Y}} = \mathbf{I}_m\,\mathbf{Y}$$

Finally, we reconstruct an approximation to $\mathbf{X}$ from the truncated set of transform coefficients, obtaining

$$\hat{\mathbf{X}} = \mathbf{V}\,\hat{\mathbf{Y}}.$$

The goal is to show that the squared error $E\|\mathbf{X} - \hat{\mathbf{X}}\|^2$ is a minimum when the matrices $\mathbf{U}$ and $\mathbf{V}$ are the Karhunen-Loève transform and its inverse, respectively.

We can decompose any $\mathbf{X}$ into a component $\mathbf{X}_N$ in the null-space of the matrix $\mathbf{U}\ \mathbf{I}_m\ \mathbf{V}$ and a component $\mathbf{X}_R$ in the range of $\mathbf{U}\ \mathbf{I}_m\ \mathbf{V}$. These components are orthogonal, so we have

$$E\|\mathbf{X} - \hat{\mathbf{X}}\|^2 = E\|\mathbf{X}_R - \hat{\mathbf{X}}\|^2 + E\|\mathbf{X}_N\|^2. \tag{1.7}$$

We assume without loss of generality that the matrices $\mathbf{U}$ and $\mathbf{V}$ are full rank. The null-space of our approximation is completely determined by our choice of $\mathbf{U}$. Hence we are free to choose $\mathbf{V}$ to minimize $E\|\hat{\mathbf{X}} - \mathbf{X}_R\|^2$.

Setting $\mathbf{V} = \mathbf{U^{-1}}$ gives $\|\hat{\mathbf{X}} - \mathbf{X}_R\|^2 = 0$, thus providing the necessary minimization.

Now we need to find the $\mathbf{U}$ that minimizes $E\|\mathbf{X}_N\|^2$. We expand

$$E\|\mathbf{X} - \hat{\mathbf{X}}\|^2 = E[\mathbf{X}^T(\mathbf{I} - \mathbf{U}^{-1}\mathbf{I}_m\mathbf{U})^T(\mathbf{I} - \mathbf{U}^{-1}\mathbf{I}_m\mathbf{U})\mathbf{X}]. \tag{1.8}$$

We now show that the requisite matrix $\mathbf{U}$ is orthogonal. We first factor $\mathbf{U}^{-1}$ into the product of an orthogonal matrix Q and an upper triangular matrix $\mathbf{S}$. After some algebra, we find that

$$E\|\mathbf{X} - \hat{\mathbf{X}}\|^2 = E[\mathbf{X}^T\mathbf{Q}(\mathbf{I} - \mathbf{I}_m)\mathbf{Q}^T\mathbf{X}] + E[\mathbf{X}^T\mathbf{Q}\mathbf{B}^T\mathbf{B}\mathbf{Q}^T\mathbf{X}], \tag{1.9}$$

where $\mathbf{B} = 0$ if and only if $\mathbf{S}$ is diagonal. The second term in (1.9) is always positive, since $\mathbf{B}^T\mathbf{B}$ is positive semidefinite. Thus, an orthogonal $\mathbf{U}$ provides the best linear transform.

The matrix $(\mathbf{I} - \mathbf{U}^T\mathbf{I}_m\mathbf{U})$ performs an orthogonal projection of $\mathbf{X}$ onto a subspace of dimension $N - m$. We need to find $\mathbf{U}$ such that the variation of $\mathbf{X}$ is minimized in this subspace. The energy of the projection of $\mathbf{X}$ onto a k-dimensional subspace spanned by orthogonal vectors $\mathbf{q_1}, \ldots, \mathbf{q_k}$ is given by

$$E\|\sum_{j=1}^{k}\mathbf{q_j}^T\mathbf{X}\mathbf{q_j}\|^2 = \sum_{j=1}^{k}\mathbf{q_j}^T\mathbf{R_X}\mathbf{q_j} \tag{1.10}$$

The one-dimensional subspace in which the projection of $\mathbf{X}$ has the smallest expected energy is the subspace spanned by the vector that minimizes the quadratic form $\mathbf{q}^T\mathbf{R_X}\mathbf{q}$. This minimum is attained when $\mathbf{q}$ is the eigenvector of $\mathbf{R_X}$ with the smallest eigenvalue.

In general, the k-dimensional subspace $\mathcal{P}_k$ in which the expected energy of the projection of $\mathbf{X}$ is minimized is the space spanned by the eigenvectors $\mathbf{v}_{N-k+1}, \ldots, \mathbf{v}_N$ of $\mathbf{R_X}$ corresponding to the k smallest eigenvalues $\lambda_{N-k+1}, \ldots, \lambda_N$. The proof is by induction. For simplicity we assume that the eigenvalues λ_k are distinct.

We have shown above that $\mathcal{P}_1$ is in fact the space spanned by $\mathbf{v}_N$. Furthermore, this space is unique because we have assumed the eigenvalues are distinct. Suppose that the unique subspace $\mathcal{P}_k$ in which the expected the energy of $\mathbf{X}$ is minimized is the space spanned by $\mathbf{v}_{N-k+1}, \ldots, \mathbf{v}_N$. Now the subspace $\mathcal{P}_{k+1}$ must contain a vector $\mathbf{q}$ that is orthogonal to $\mathcal{P}_k$. The expected energy of the projection of $\mathbf{X}$ onto $\mathcal{P}_{k+1}$ is equal to the sum of the expected energies of the projection onto $\mathbf{q}$ and the projection onto the k-dimensional complement of $\mathbf{q}$ in $\mathcal{P}_{k+1}$. The complement of $\mathbf{q}$ in $\mathcal{P}_{k+1}$ must be $\mathcal{P}_k$, since any other subspace would result in a larger expected energy (note that this choice of subspace does not affect the choice of $\mathbf{q}$ since $\mathbf{q} \perp \mathcal{P}_k$). Now $\mathbf{q}$ minimizes $\mathbf{q}^T\mathbf{R_X}\mathbf{q}$ over the span of $\mathbf{v}_1, \ldots \mathbf{v}_{N-k}$. The requisite $\mathbf{q}$ is $\mathbf{v}_{N-k}$, which gives us the desired result.

Retaining only the first m coordinates of the Karhunen-Loève transform of $\mathbf{X}$ is equivalent to discarding the projection of $\mathbf{X}$ on the span of the eigenvectors $\mathbf{v_{m+1}}, \ldots, \mathbf{v_N}$. The above derivation shows that this projection has the minimum expected energy of any $N - m$ dimensional projection, so the resulting expected error $E\|\mathbf{X} - \hat{\mathbf{X}}\|$ is minimized.

◇

While the former result is useful for developing intuition into signal approximations, it is not directly useful for compression purposes. In signal compression, we cannot afford to keep even one of the signal components exactly. Typically one or more components of the transformed signal are quantized and indices of the quantized coefficients are transmitted to the decoder. In this case it is desirable to construct transforms that, given a fixed bitrate, will impose the smallest distortion on the signal. In the next section we derive an optimal bit allocation strategy for transformed data.

3.2 Optimal bit allocation

Assuming the components of a random vector are to be quantized separately, it remains to be determined how many levels should each of the quantizers be given. Our goal is to get the most benefit out of a fixed bit budget. In other words, each bit of information should be spent on the quantizer that offers the biggest return in terms of reducing the overall distortion. In the following, we formalize this concept, and will eventually use it to formulate the optimal transform coder.

Suppose we have a set of k random variables, $X_1, ..., X_k$, all zero-mean, with variances $E[X_i] = \sigma_i^2$. Assuming that the p.d.f. of each of the random variables is known, we can design optimal quantizers for each variable for any given number of quantization levels. The log of the number of quantization levels represents the rate of the quantizer in bits.

We assume a high-resolution regime in which the distortion is much smaller than the input variance ($D_i \ll \sigma_i^2$). One can then show [1] that a quantizer with 2^{b_i} levels has distortion

$$D_i(b_i) \approx h_i \, \sigma_i^2 \, 2^{-2b_i}. \tag{1.11}$$

Here h_i is given by

$$h_i = \frac{1}{12} \left\{ \int_{-\infty}^{\infty} [p_{X_i}(x)]^{1/3} \, dx \right\}^3 \tag{1.12}$$

where $p_{X_i}(x)$ denotes the p.d.f. of the i-th random variable. The optimal bit allocation is therefore a problem of minimizing

$$\sum_{i=1}^{k} h_i \, \sigma_i^2 \, 2^{-2b_i}$$

subject to the constraint $\sum b_i = B$. The following result is due to Huang and Schultheiss [9], which we present without proof: *the optimal bit assignment is achieved by the following distribution of bits:*

$$b_i = \bar{b} + \frac{1}{2}\log_2 \frac{\sigma_i^2}{\rho^2} + \frac{1}{2}\log_2 \frac{h_i}{H} \tag{1.13}$$

where

$$\bar{b} = \frac{B}{k}$$

is the arithmetic mean of the bitrates,

$$\rho^2 = \left(\prod_{i=1}^{k} \sigma_i^2 \right)^{\frac{1}{k}}$$

is the geometric mean of the variances, and H is the geometric mean of the coefficients h_i. This distribution will result in the overall optimal distortion

$$D_{opt} = k\, H\, \rho^2\, 2^{-2\bar{b}} \tag{1.14}$$

Using this formula on the toy example at the beginning of this section, if the transform in Equation (1.6) is applied to the random process characterized by Equation (1.5), a gain of more than 7 dB in distortion will be achieved. The resulting quantizer is shown on the right hand side of Figure 4. We will next use Equation (1.14) to establish the optimality of the Karhunen-Loève transform for Gaussian processes.

3.3 Optimality of the Karhunen-Loève Transform

Once the signal is quantized, the Karhunen-Loève transform is no longer necessarily optimal. However, for the special case of a jointly Gaussian signal, the K-L transform retains its optimality even in the presence of quantization.

Theorem 2 *For a zero-mean, jointly Gaussian random vector, among all block transforms, the Karhunen-Loève transform minimizes the distortion at a given rate.*

Proof: Take an arbitrary orthogonal transformation on the Gaussian random vector $\mathbf{X}$, resulting in $\mathbf{Y}$. Let σ_i^2 be the variance of the i-th transform coefficient Y_i. Then, according to the Huang and Schultheiss result, the minimum distortion achievable for any transform is equal to

$$D_T = E\left[||\mathbf{Y} - \hat{\mathbf{Y}}||^2 \right] = N\, h_g\, 2^{-2\bar{b}} \left(\prod_{i=1}^{N} \sigma_i^2 \right)^{\frac{1}{N}} \tag{1.15}$$

where $h_g = \frac{\sqrt{3}\pi}{2}$ is the quantization coefficient for the Gaussian.

The key, then, is to find the transform that minimizes the product of the transformed coefficient variances, $\prod_{i=1}^{N} \sigma_i^2$. Hadamard's inequality [10] provides us with a way to find the minimum product. Hadamard's inequality states that for a positive semidefinite matrix, the product of the diagonal elements is greater than or equal to the determinant of the matrix. Equality is attained if and only if the matrix is diagonal. $\mathbf{R_Y}$ is positive semidefinite, so we have $\prod_{i=1}^{N} \sigma_i^2 \geq \det(\mathbf{R_Y})$. Hence

$$D_T \geq N\, h_g\, 2^{-2\bar{b}} (det\, \mathbf{R}_Y)^{\frac{1}{N}} = N\, h_g\, 2^{-2\bar{b}} (\det \mathbf{R}_X)^{\frac{1}{N}} \tag{1.16}$$

with equality achieved only when $\mathbf{R}_Y$ is diagonal. Because the Karhunen-Loève transform diagonalizes $\mathbf{R_Y}$, it provides the optimal decorrelating transform.

◇

3.4 The Discrete Cosine Transform

While the Karhunen-Loève transform (KLT) has nice theoretical properties, there are significant obstacles to its use in practice. The first problem is that we need to know the covariances for all possible pairs of pixels for images of interest. This requires estimating an extremely large number of parameters. If instead we make some stationarity assumptions and estimate correlations from the image we wish to code, the transform becomes image dependent. The amount of side information required to tell the decoder which transform to use is prohibitive. The second problem is that the KLT is slow, requiring $O(N^4)$ operations to apply to an $N \times N$ image. We need instead to find transforms that approximately duplicate the properties of the KLT over the large class of images of interest, and that can be implemented using fast algorithms.

The first step towards building an approximate K-L transform is noting the Toeplitz structure of autocorrelation matrices for stationary processes. Asymptotically, as the dimensions of a Toeplitz matrix increase, its eigenvectors converge to complex exponentials. In other words, regardless of the second order properties of the random process, the Fourier transform diagonalizes the autocorrelation function asymptotically. Therefore, in finite dimensions, the discrete Fourier transform (DFT) can serve as an approximation to the Karhunen-Loève transform.

In practice, a close relative of the DFT, namely the Discrete Cosine Transform (DCT) [11], is used to diagonalize $\mathbf{R}_X$. The DCT has the form[5]

$$c(k,n) = \begin{cases} \frac{1}{\sqrt{N}} & k = 0 \quad ,0 \leq n \leq N-1 \\ \sqrt{\frac{2}{N}} \cos \frac{\pi(2n+1)k}{2N} & 1 \leq k \leq N-1 \quad ,0 \leq n \leq N-1 \end{cases} \tag{1.17}$$

[5] Several slightly different forms of DCT exist. See [12] for more details.

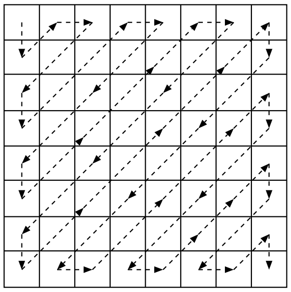


FIGURE 6. *Zig-zag scan of DCT coefficients*

The DCT has several advantages over the DFT. First, unlike the DFT, the DCT is a real-valued transform that generates real coefficients from real-valued data. Second, the ability of the DCT and the DFT to pack signal energy into a small number of coefficients is a function of the global smoothness of these signals. The DFT is equivalent to a discrete-time Fourier transform (DTFT) of a periodically extended version of the block of data under consideration. This block extension in general results in the creation of artificial discontinuities at block boundaries and reduces the DFT's effectiveness at packing image energy into the low frequencies. In contrast, the DCT is equivalent to the DTFT of the repetition of the *symmetric extension* of the data, which is by definition continuous. The lack of artificial discontinuities at the edges gives the DCT better energy compaction properties, and thus makes it a better approximation to the KLT for signals of interest.[6]

The DCT is the cornerstone of the JPEG image compression standard. In the baseline version of this standard, the image is divided into a number of 8×8 pixel blocks, and the block DCT is applied to each block. The matrix of DCT coefficients is then quantized by a bank of uniform scalar quantizers. While the standard allows direct specification of these quantizers by the encoder, it also provides a "default" quantizer bank, which is often used by most encoders. This default quantizer bank is carefully designed to approach optimal rate-distortion for a large class of visual signals. Such a quantization strategy is also compatible with the human visual system, because it quantizes high-frequency signals more coarsely, and the human eye is less sensitive to errors in the high frequencies.

The quantized coefficients are then zig-zag scanned as shown in Figure 6 and entropy-coded. The syntax of JPEG for transmitting entropy coded coefficients makes further use of our *a priori* knowledge of the likely values of these coefficients. Instead of coding and transmitting each coefficient

[6] This is true only because the signals of interest are generally lowpass. It is perfectly possible to generate signals for which the DFT performs better energy compaction than DCT. However, such signals are unlikely to appear in images and other visual contexts.

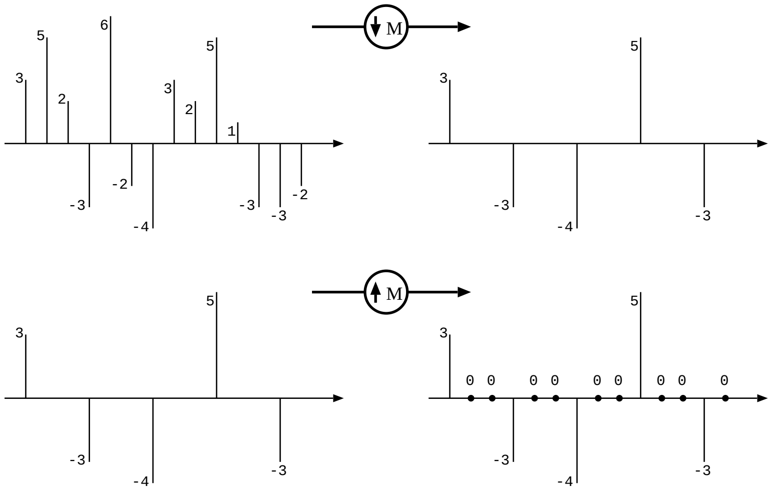


FIGURE 7. *Example of decimation and interpolation on a sampled signal, with* $M = 3$.

separately, the encoder transmits the runlength of zeros before the next nonzero coefficient, jointly with the value of the next non-zero coefficient. It also has a special "End of Block" symbol that indicates no more non-zero coefficients remain in the remainder of the zig-zag scan. Because large runs of zeros often exist in typical visual signals, usage of these symbols gives rise to considerable savings in bitrate.

Note that coding zeros jointly, while being simple and entirely practical, is outside the bounds of scalar quantization. In fact, run-length coding of zeros can be considered a special case of vector quantization. It captures redundancies beyond what is possible even with an optimal transform and optimal bitrate allocation. This theme of jointly coding zeros re-emerges later in the context of zerotree coding of wavelet coefficients, and is used to generate very powerful coding algorithms.

DCT coding with zig-zag scan and entropy coding is remarkably efficient. But the popularity of JPEG owes at least as much to the computational efficiency of the DCT as to its performance. The source of computational savings in fast DCT algorithms are folding of multiplies, as well as the redundancies inherent in a 2-D transform. We refer the reader to the extensive literature for further details [12, 13, 14, 15].

3.5 Subband transforms

The Fourier-based transforms, including the DCT, are a special case of subband transforms. A subband transformer is a multi-rate digital signal processing system. There are three elements to multi-rate systems: filters, interpolators, and decimators. Decimation and interpolation operations are

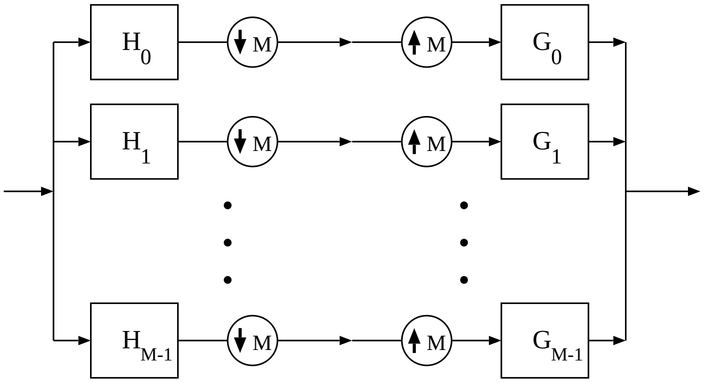


FIGURE 8. *Filter bank*

illustrated in Figure 7. A decimator is is an element that reduces the sampling rate of a signal by a factor M. For of every M samples, we retain one sample and discard the rest. An interpolator increases the sampling rate of a signal by a factor M by introducing $M-1$ zero samples in between each pair samples of the original signal. Note that "interpolator" is somewhat of a misnomer, since these "interpolators" only add zeros in between samples and make for very poor interpolation by themselves.

A subband system, as shown in Figure 8, consists of two sets of filter banks, along with decimators and interpolators. On the left side of the figure we have the forward stage of the subband transform. The signal is sent through the input of the first set of filters, known as the *analysis filter bank*. The output of these filters is passed through decimators, which retain only one out of every M samples. The right hand side of the figure is the inverse stage of the transform. The filtered and decimated signal is first passed through a set of *interpolators*. Next it is passed through the *synthesis filter bank*. Finally, the components are recombined.

The combination of decimation and interpolation has the effect of zeroing out all but one out of M samples of the filtered signal. Under certain conditions, the original signal can be reconstructed exactly from this decimated M-band representation. The ideas leading to the perfect reconstruction conditions were discovered in stages by a number of investigators, including Croisier et al. [16], Vaidyanathan [17], Smith and Barnwell [18, 19] and Vetterli [20, 21]. For a detailed presentation of these developments, we refer the reader to the comprehensive texts by Vaidyanathan [22] and Vetterli and Kovačević [23].

We have seen in our discussion of quantization strategies that we need to decorrelate pixels in order for scalar quantization to work efficiently. The Fourier transform diagonalizes Toeplitz matrices asymptotically as matrix dimensions increase. In other words, it decorrelates pixels as the length of our pixel vectors goes to infinity, and this has motivated the use of

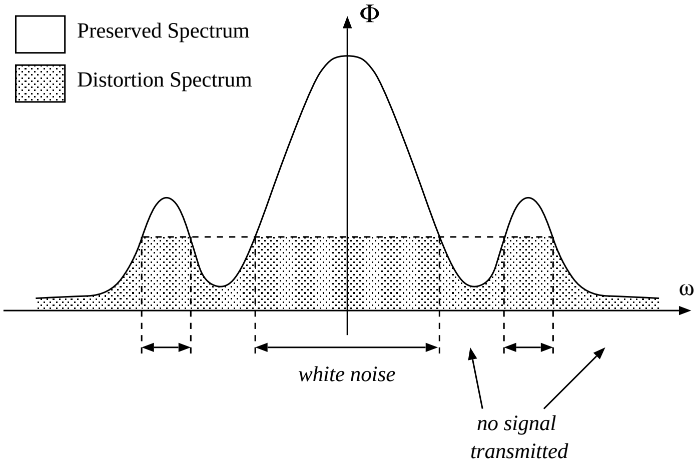


FIGURE 9. *Inverse water filling of the spectrum for the rate-distortion function of a Gaussian source with memory.*

Fourier-type transforms like DCT. Filter banks provide an alternative way to approximately decorrelate pixels, and in general have certain advantages over the Fourier transform.

To understand how filterbanks decorrelate signals, consider the following simplified analysis: Assume we have a Gaussian source with memory, i.e. correlated, with power spectral density (p.s.d.) $\Phi_X(\omega)$. The rate-distortion function for this source is [24]

$$D(\theta) = \frac{1}{2\pi^2} \int_\omega \min(\theta, \Phi_X(\omega)) d\omega \tag{1.18}$$

$$R(\theta) = \frac{1}{4\pi^2} \int_\omega \max\left(0, \log(\frac{\Phi_X(\omega)}{\theta})\right) d\omega \tag{1.19}$$

Each value of θ produces a point on the rate-distortion curve. The goal of any quantization scheme is to mimic the rate-distortion curve, which is optimal. Thus, a simple approach is suggested by the equations above: at frequencies where signal power is less than θ, it is not worthwhile to spend any bits, therefore all the signal is thrown away (signal power = noise power). At frequencies where signal power is greater than θ, enough bitrate is assigned so that the noise power is exactly θ, and signal power over and above θ is preserved. This procedure is known as *inverse water-filling*. The solution is illustrated in Figure 9.

Of course it is not practical to consider each individual frequency separately, since there are uncountably many of them. However, it can be shown that instead of considering individual frequencies, one can consider bands of frequencies together, as long as the power spectral density within each band is constant. This is where filter banks come into the picture.

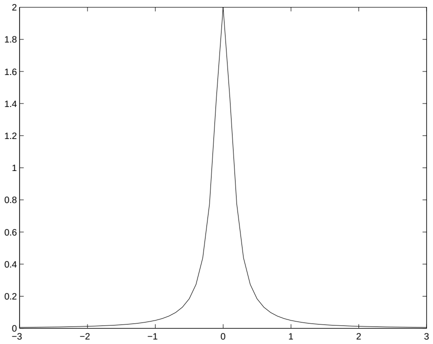


FIGURE 10. *Spectral power density corresponding to an exponential correlation profile.*

Filter banks are used to divide signals into frequency bands, or subbands. For example, a filter bank with two analysis filters can be used to divide a signal into highpass and lowpass components, each with half the bandwidth of the original signal. We can approximately decorrelate a Gaussian process by carving its power spectrum into flat segments, multiplying each segment by a suitable factor, and adding the bands together again to obtain an overall flat (white) p.s.d.

We see that both filter banks and Fourier transform are based on frequency domain arguments. So is one superior the other, and why? The answer lies in the space-frequency characteristics of the two methods. Fourier bases are very exact in frequency, but are spatially not precise. In other words, the energy of the Fourier basis elements is concentrated in one frequency, but spread over all space. This would not be a problem if image pixels were individually and jointly Gaussian, as assumed in our analysis. However, in reality, pixels in images of interest are generally not jointly Gaussian, especially not across image discontinuities (edges). In contrast to Fourier basis elements, subband bases not only have fairly good frequency concentration, but also are spatially compact. If image edges are not too closely packed, most of the subband basis elements will not intersect with them, thus performing a better decorrelation on average.

The next question is: how should one carve the frequency spectrum to maximize the benefits, given a fixed number of filter banks? A common model for the autocorrelation of images [25] is that pixel correlations fall off exponentially with distance. We have

$$R_X(\delta) := e^{-\omega_0|\delta|} \,, \tag{1.20}$$

where δ is the lag variable. The corresponding power spectral density is given by

$$\Phi_X(\omega) = \frac{2\omega_0}{\omega_0^2 + (2\pi\omega)^2} \tag{1.21}$$

This spectral density is shown in Figure 10. From the shape of this density we see that in order to obtain segments in which the spectrum is flat, we need to partition the spectrum finely at low frequencies, but only coarsely at high frequencies. The subbands we obtain by this procedure will be approximately vectors of white noise with variances proportional to the power spectrum over their frequency range. We can use an procedure similar to that described for the KLT for coding the output. As we will see below, this particular partition of the spectrum is closely related to the wavelet transform.

4 Wavelets: A Different Perspective

4.1 Multiresolution Analyses

The discussion so far has been motivated by probabilistic considerations. We have been assuming our images can be reasonably well-approximated by Gaussian random vectors with a particular covariance structure. The use of the wavelet transform in image coding is motivated by a rather different perspective, that of approximation theory. We assume that our images are locally smooth functions and can be well-modeled as piecewise polynomials. Wavelets provide an efficient means for approximating such functions with a small number of basis elements. This new perspective provides some valuable insights into the coding process and has motivated some significant advances.

We motivate the use of the wavelet transform in image coding using the notion of a multiresolution analysis. Suppose we want to approximate a continuous-valued square-integrable function $f(x)$ using a discrete set of values. For example, $f(x)$ might be the brightness of a one-dimensional image. A natural set of values to use to approximate $f(x)$ is a set of regularly-spaced, weighted local averages of $f(x)$ such as might be obtained from the sensors in a digital camera.

A simple approximation of $f(x)$ based on local averages is a step function approximation. Let $\phi(x)$ be the box function given by $\phi(x) = 1$ for $x \in [0, 1)$ and 0 elsewhere. A step function approximation to $f(x)$ has the form

$$Af(x) = \sum_n f_n \phi(x - n), \tag{1.22}$$

where f_n is the height of the step in $[n, n+1)$. A natural value for the heights f_n is simply the average value of $f(x)$ in the interval $[n, n+1)$. This gives $f_n = \int_n^{n+1} f(x) dx$.

We can generalize this approximation procedure to building blocks other than the box function. Our more generalized approximation will have the form

$$Af(x) = \sum_n \langle \tilde{\phi}(x-n), f(x) \rangle \phi(x-n). \tag{1.23}$$

Here $\tilde{\phi}(x)$ is a weight function and $\phi(x)$ is an interpolating function chosen so that $\langle \phi(x), \tilde{\phi}(x-n) \rangle = \delta[n]$. The restriction on $\phi(x)$ ensures that our approximation will be exact when $f(x)$ is a linear combination of the functions $\phi(x-n)$. The functions $\phi(x)$ and $\tilde{\phi}(x)$ are normalized so that $\int |\phi(x)|^2 dx = \int |\tilde{\phi}(x)|^2 dx = 1$. We will further assume that $f(x)$ is periodic with an integer period so that we only need a finite number of coefficients to specify the approximation $Af(x)$.

We can vary the resolution of our approximations by dilating and contracting the functions $\phi(x)$ and $\tilde{\phi}(x)$. Let $\phi^j(x) = 2^{\frac{j}{2}} \phi(2^j x)$ and $\tilde{\phi}^j(x) = 2^{\frac{j}{2}} \tilde{\phi}(2^j x)$. We form the approximation $A^j f(x)$ by projecting $f(x)$ onto the span of the functions $\{\phi^j(x - 2^{-j}k)\}_{k \in \mathbb{Z}}$, computing

$$A^j f(x) = \sum_k \langle f(x), \tilde{\phi}^j(x - 2^{-j}k) \rangle \phi^j(x - 2^{-j}k). \tag{1.24}$$

Let V_j be the space spanned by the functions $\{\phi^j(x - 2^{-j}k)\}$. Our resolution j approximation $A^j f$ is simply a projection (not necessarily an orthogonal one) of $f(x)$ onto the span of the functions $\phi^j(x - 2^{-j}k)$.

For our box function example, the approximation $A^j f(x)$ corresponds to an orthogonal projection of $f(x)$ onto the space of step functions with step width 2^{-j}. Figure 11 shows the difference between the coarse approximation $A^0 f(x)$ on the left and the higher resolution approximation $A^1 f(x)$ on the right. Dilating scaling functions give us a way to construct approximations to a given function at various resolutions. An important observation is that if a given function is sufficiently smooth, the differences between approximations at successive resolutions will be small.

Constructing our function $\phi(x)$ so that approximations at scale j are special cases of approximations at scale $j+1$ will make the analysis of differences of functions at successive resolutions much easier. The function $\phi(x)$ from our box function example has this property, since step functions with width 2^{-j} are special cases of step functions with step width 2^{-j-1}. For such $\phi(x)$'s the spaces of approximations at successive scales will be nested, i.e. we have $V_j \subset V_{j+1}$.

The observation that the differences $A^{j+1}f - A^j f$ will be small for smooth functions is the motivation for the Laplacian pyramid [26], a way of transforming an image into a set of small coefficients. The 1-D analog of the procedure is as follows: we start with an initial discrete representation of a function, the N coefficients of $A^j f$. We first split this function into the sum

$$A^j f(x) = A^{j-1} f(x) + [A^j f(x) - A^{j-1} f(x)]. \tag{1.25}$$

Because of the nesting property of the spaces V_j, the difference $A^j f(x) - A^{j-1} f(x)$ can be represented exactly as a sum of N translates of the func-

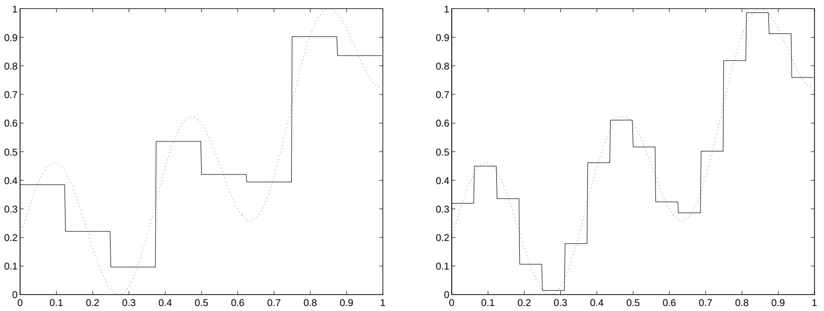


FIGURE 11. *A continuous function $f(x)$ (plotted as a dotted line) and box function approximations (solid lines) at two resolutions. On the left is the coarse approximation $A^0 f(x)$ and on the right is the higher resolution approximation $A^1 f(x)$.*

tion $\phi^j(x)$. The key point is that the coefficients of these ϕ^j translates will be small provided that $f(x)$ is sufficiently smooth, and hence easy to code. Moreover, the dimension of the space V_{j-1} is only half that of the space V_j, so we need only $\frac{N}{2}$ coefficients to represent $A^{j-1}f$. (In our box-function example, the function $A^{j-1}f$ is a step function with steps twice as wide as $A^j f$, so we need only half as many coefficients to specify $A^{j-1}f$.) We have partitioned $A^j f$ into N difference coefficients that are easy to code and $\frac{N}{2}$ coarse-scale coefficients. We can repeat this process on the coarse-scale coefficients, obtaining $\frac{N}{2}$ easy-to-code difference coefficients and $\frac{N}{4}$ coarser scale coefficients, and so on. The end result is $2N-1$ difference coefficients and a single coarse-scale coefficient.

Burt and Adelson [26] have employed a two-dimensional version of the above procedure with some success for an image coding scheme. The main problem with this procedure is that the Laplacian pyramid representation has more coefficients to code than the original image. In 1-D we have twice as many coefficients to code, and in 2-D we have $\frac{4}{3}$ as many.

4.2 Wavelets

We can improve on the Laplacian pyramid idea by finding a more efficient representation of the difference $D^{j-1}f = A^j f - A^{j-1}f$. The idea is that to decompose a space of fine-scale approximations V_j into a direct sum of two subspaces, a space V_{j-1} of coarser-scale approximations and its complement, W_{j-1}. This space W_{j-1} is a space of differences between coarse and fine-scale approximations. In particular, $A^j f - A^{j-1}f \in W^{j-1}$ for any f. Elements of the space can be thought of as the additional details that must be supplied to generate a finer-scale approximation from a coarse one.

Consider our box-function example. If we limit our attention to functions on the unit interval, then the space V_j is a space of dimension 2^j. We can

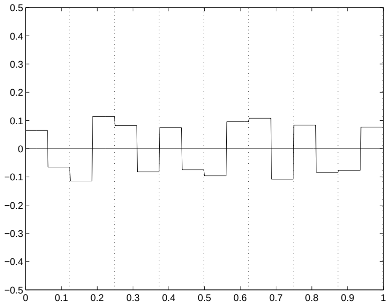


FIGURE 12. *$D^0 f(x)$, the difference between the coarse approximation $A^0 f(x)$ and the finer scale approximation $A^1 f(x)$ from figure 11.*

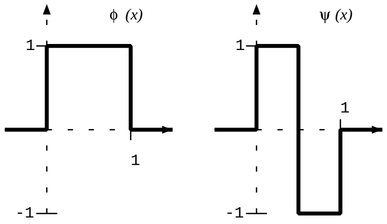


FIGURE 13. *The Haar scaling function and wavelet.*

decompose V_j into the space V_{j-1}, the space of resolution 2^{-j+1} approximations, and W_{j-1}, the space of details. Because V_{j-1} is of dimension 2^{j-1}, W_{j-1} must also have dimension 2^{j-1} for the combined space V_j to have dimension 2^j. This observation about the dimension of W_j provides us with a means to circumvent the Laplacian pyramid's problems with expansion.

Recall that in the Laplacian pyramid we represent the difference $D^{j-1}f$ as a sum of N fine-scale basis functions $\phi^j(x)$. This is more information than we need, however, because the space of functions $D^{j-1}f$ is spanned by just $\frac{N}{2}$ basis functions. Let c^j_k be the expansion coefficient $\langle \tilde{\phi}^j(x - 2^{-j}k), f(x)\rangle$ in the resolution j approximation to $f(x)$. For our step functions, each coefficient c^{j-1}_k is the average of the coefficients c^j_{2k} and c^j_{2k+1} from the resolution j approximation. In order to reconstruct $A^j f$ from $A^{j-1}f$, we only need the $\frac{N}{2}$ differences $c^j_{2k+1} - c^j_{2k}$. Unlike the Laplacian pyramid, there is no expansion in the number of coefficients needed if we store these differences together with the coefficients for $A^{j-1}f$.

The differences $c^j_{2k+1} - c^j_{2k}$ in our box function example correspond (up to a normalizing constant) to coefficients of a basis expansion of the space of details W_{j-1}. Mallat has shown that in general the basis for W_j consists of translates and dilates of a single prototype function $\psi(x)$, called a *wavelet* [27]. The basis for W_j is of the form $\psi^j(x - 2^{-j}k)$ where $\psi^j(x) = 2^{\frac{j}{2}}\psi(x)$.

Figure 13 shows the scaling function (a box function) for our box function example together with the corresponding wavelet, the Haar wavelet. Figure 12 shows the function $D^0 f(x)$, the difference between the approximations $A^1 f(x)$ and $A^1 f(x)$ from Figure 11. Note that each of the intervals separated by the dotted lines contains a translated multiple of $\psi(x)$.

The dynamic range of the differences $D^0 f(x)$ in Figure 12 is much smaller than that of $A^1 f(x)$. As a result, it is easier to code the expansion coefficients of $D^0 f(x)$ than to code those of the higher resolution approximation $A^1 f(x)$. The splitting $A^1 f(x)$ into the sum $A^0 f(x) + D^0 f(x)$ performs a packing much like that done by the Karhunen-Loève transform. For smooth functions $f(x)$ the result of the splitting of $A^1 f(x)$ into a sum of a coarser approximation and details is that most of the variation is contained in $A^0 f$, and $D^0 f$ is near zero. By repeating this splitting procedure, partitioning $A^0 f(x)$ into $A^{-1} f(x) + D^{-1} f(x)$, we obtain the wavelet transform. The result is that an initial function approximation $A^j f(x)$ is decomposed into the telescoping sum

$$A^j f(x) = D^{j-1} f(x) + D^{j-2} f(x) + \ldots + D^{j-n} f(x) + A^{j-n} f(x). \quad (1.26)$$

The coefficients of the differences $D^{j-k} f(x)$ are easier to code than the expansion coefficients of the original approximation $A^j f(x)$, and there is no expansion of coefficients as in the Laplacian pyramid.

4.3 Recurrence Relations

For the repeated splitting procedure above to be practical, we will need an efficient algorithm for obtaining the coefficients of the expansions $D^{j-k} f$ from the original expansion coefficients for $A^j f$. A key property of our scaling functions makes this possible.

One consequence of our partitioning of the space of resolution j approximations, V_j, into a space of resolution $j-1$ approximations V_{j-1} and resolution $j-1$ details W_{j-1} is that the scaling functions $\phi(x)$ possess self-similarity properties. Because $V_{j-1} \subset V_j$, we can express the function $\phi_{j-1}(x)$ as a linear combination of the functions $\phi_j(x-n)$. In particular we have

$$\phi(x) = \sum_k h_k \phi(2x - k). \quad (1.27)$$

Similarly, we have

$$\begin{aligned} \tilde{\phi}(x) &= \sum_k \tilde{h}_k \tilde{\phi}(2x-k) \\ \psi(x) &= \sum_k g_k \phi(2x-k) \\ \tilde{\psi}(x) &= \sum_k \tilde{g}_k \tilde{\phi}(2x-k). \end{aligned} \quad (1.28)$$

These recurrence relations provide the link between wavelet transforms and subband transforms. Combining (4.3) and (4.3) with (4.1), we obtain a simple means for splitting the N expansion coefficients for $A^j f$ into the $\frac{N}{2}$ expansion coefficients for the coarser-scale approximation $A^{j-1}f$ and the $\frac{N}{2}$ coefficients for the details $D^{j-1}f$. Both the coarser-scale approximation coefficients and the detail coefficients are obtained by convolving the coefficients of $A^j f$ with a filter and downsampling by a factor of 2. For the coarser-scale approximation, the filter is a low-pass filter with taps given by $\tilde{h}_{-k}$. For the details, the filter is a high-pass filter with taps $\tilde{g}_{-k}$. A related derivation shows that we can invert the split by upsampling the coarser-scale approximation coefficients and the detail coefficients by a factor of 2, convolving them with synthesis filters with taps h_k and g_k, respectively, and adding them together.

We begin the forward transform with a signal representation in which we have very fine temporal localization of information but no frequency localization of information. Our filtering procedure splits our signal into low-pass and high-pass components and downsamples each. We obtain twice the frequency resolution at the expense of half of our temporal resolution. On each successive step we split the lowest frequency signal component in to a low pass and high pass component, each time gaining better frequency resolution at the expense of temporal resolution. Figure 14 shows the partition of the time-frequency plane that results from this iterative splitting procedure. As we discussed in Section 3.5, such a decomposition, with its wide subbands in the high frequencies and narrow subbands at low frequencies leads to effective data compression for a common image model, a Gaussian random process with an exponentially decaying autocorrelation function.

The recurrence relations give rise to a fast algorithm for splitting a fine-scale function approximation into a coarser approximation and a detail function. If we start with an N coefficient expansion $A^j f$, the first split requires kN operations, where k depends on the lengths of the filters we use. The approximation A^{J-1} has $\frac{N}{2}$ coefficients, so the second split requires

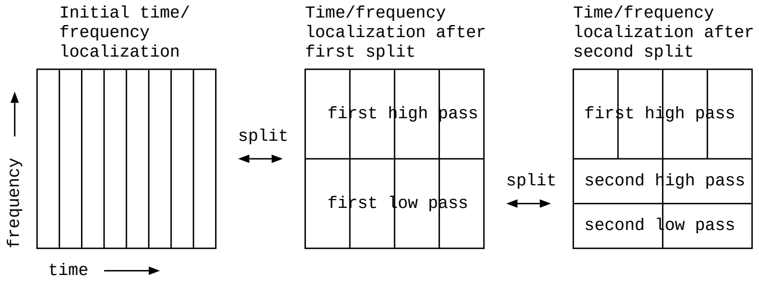


FIGURE 14. *Partition of the time-frequency plane created by the wavelet transform.*

$k\frac{N}{2}$ operations. Each successive split requires half as much work, so the overall transform requires $O(N)$ work.

4.4 Wavelet Transforms vs. Subband Decompositions

The wavelet transform is a special case of a subband transform, as the derivation of the fast wavelet transform reveals. What, then, does the wavelet transform contribute to image coding? As we discuss below, the chief contribution of the wavelet transform is one of perspective. The mathematical machinery used to develop the wavelet transform is quite different than that used for developing subband coders. Wavelets involve the analysis of continuous functions whereas analysis of subband decompositions is more focused on discrete time signals. The theory of wavelets has a strong spatial component whereas subbands are more focused in the frequency domain.

The subband and wavelet perspectives represent two extreme points in the analysis of this iterated filtering and downsampling process. The filters used in subband decompositions are typically designed to optimize the frequency domain behavior of a single filtering and subsampling. Because wavelet transforms involve *iterated* filtering and downsampling, the analysis of a single iteration is not quite what we want. The wavelet basis functions can be obtained by iterating the filtering and downsampling procedure an infinite number of times. Although in applications we iterate the filtering and downsampling procedure only a small number of times, examination of the properties of the basis functions provides considerable insight into the effects of iterated filtering.

A subtle but important point is that when we use the wavelet machinery, we are implicitly assuming that the values we transform are actually fine-scale scaling function coefficients rather than samples of some function. Unlike the subband framework, the wavelet framework explicitly specifies an underlying continuous-valued function from which our initial coefficients are derived. The use of continuous-valued functions allows the use of powerful analytical tools, and it leads to a number of insights that can be used to guide the filter design process. Within the continuous-valued framework we can characterize the types of functions that can be represented exactly with a limited number of wavelet coefficients. We can also address issues such as the smoothness of the basis functions. Examination of these issues has led to important new design criteria for both wavelet filters and subband decompositions.

A second important feature of the wavelet machinery is that it involves both spatial as well as frequency considerations. The analysis of subband decompositions is typically more focused on the frequency domain. Coefficients in the wavelet transform correspond to features in the underlying function in specific, well-defined locations. As we will see below, this explicit use of spatial information has proven quite valuable in motivating some of the most effective wavelet coders.

4.5 Wavelet Properties

There is an extensive literature on wavelets and their properties. See [28], [23], or [29] for an introduction. Properties of particular interest for image compression are the the accuracy of approximation , the smoothness, and the support of these bases.

The functions $\phi(x)$ and $\psi(x)$ are the building blocks from which we construct our compressed images. When compressing natural images, which tend contain locally smooth regions, it is important that these building blocks be reasonably smooth. If the wavelets possess discontinuities or strong singularities, coefficient quantization errors will cause these discontinuities and singularities to appear in decoded images. Such artifacts are highly visually objectionable, particularly in smooth regions of images. Procedures for estimating the smoothness of wavelet bases can be found in [30] and [31]. Rioul [32] has found that under certain conditions that the smoothness of scaling functions is a more important criterion than a standard frequency selectivity criterion used in subband coding.

Accuracy of approximation is a second important design criterion that has arisen from wavelet framework. A remarkable fact about wavelets is that it is possible to construct smooth, compactly supported bases that can exactly reproduce any polynomial up to a given degree. If a continuous-valued function $f(x)$ is locally equal to a polynomial, we can reproduce that portion of $f(x)$ exactly with just a few wavelet coefficients. The degree of the polynomials that can be reproduced exactly is determined by the number of *vanishing moments* of the dual wavelet $\tilde{\psi}(x)$. The dual wavelet $\tilde{\psi}(x)$ has N vanishing moments provided that $\int x^k \tilde{\psi}(x) dx = 0$ for $k = 0, \ldots, N$. Compactly supported bases for L^2 for which $\tilde{\psi}(x)$ has N vanishing moments can locally reproduce polynomials of degree $N-1$.

The number of vanishing moments also determines the rate of convergence of the approximations $A^j f$ to the original function f as the resolution goes to infinity. It has been shown that $\|f - A^j f\| \leq C 2^{-jN} \|f^{(N)}\|$ where N is the number of vanishing moments of $\tilde{\psi}(x)$ and $f^{(N)}$ is the Nth derivative of f [33, 34, 35].

The size of the support of the wavelet basis is another important design criterion. Suppose that the function $f(x)$ we are transforming is equal to polynomial of degree $N-1$ in some region. If $\tilde{\psi}$ has has N vanishing moments, then any basis function for which the corresponding dual function lies entirely in the region in which f is polynomial will have a zero coefficient. The smaller the support of $\tilde{\psi}$ is, the more zero coefficients we will obtain. More importantly, edges produce large wavelet coefficients. The larger $\tilde{\psi}$ is, the more likely it is to overlap an edge. Hence it is important that our wavelets have reasonably small support.

There is a tradeoff between wavelet support and the regularity and accuracy of approximation. Wavelets with short support have strong constraints on their regularity and accuracy of approximation, but as the support is increased they can be made to have arbitrary degrees of smoothness and numbers of vanishing moments. This limitation on support is equivalent to keeping the analysis filters short. Limiting filter length is also an important consideration in the subband coding literature, because long filters lead to ringing artifacts around edges.

5 A Basic Wavelet Image Coder

State-of-the-art wavelet coders are all derived from the transform coder paradigm. There are three basic components that underly current wavelet coders: a decorrelating transform, a quantization procedure, and an entropy coding procedure. Considerable current research is being performed on all three of these components. Before we discuss state-of-the-art coders in the next sections, we will describe a basic wavelet transform coder and discuss optimized versions of each of the components.[7]

5.1 Choice of Wavelet Basis

Deciding on the optimal wavelet basis to use for image coding is a difficult problem. A number of design criteria, including smoothness, accuracy of approximation, size of support, and filter frequency selectivity are known to be important. However, the best combination of these features is not known.

The simplest form of wavelet basis for images is a separable basis formed from translations and dilations of products of one dimensional wavelets. Using separable transforms reduces the problem of designing efficient wavelets to a one-dimensional problem, and almost all current coders employ separable transforms. Recent work of Sweldens and Kovačević [36] simplifies considerably the design of non-separable bases, and such bases may prove more efficient than separable transforms.

The prototype basis functions for separable transforms are $\phi(x)\phi(y)$, $\phi(x)\psi(y)$, $\psi(x)\phi(y)$, and $\psi(x)\psi(y)$. Each step of the transform for such bases involves two frequency splits instead of one. Suppose we have an $N \times N$ image. First each of the N rows in the image is split into a low-pass half and a high pass half. The result is an $N \times \frac{N}{2}$ sub-image and an $N \times \frac{N}{2}$ high-pass sub-image. Next each column of the sub-images is split into a low-pass and a high-pass half. The result is a four-way partition

[7] C++ source code for a coder that implements these components is available from the web site `http://www.cs.dartmouth.edu/~gdavis/wavelet/wavelet.html`.

of the image into horizontal low-pass/vertical low-pass, horizontal high-pass/vertical low-pass, horizontal low-pass/vertical high-pass, and horizontal high-pass/vertical high-pass sub-images. The low-pass/low-pass sub-image is subdivided in the same manner in the next step as is illustrated in Figure 17.

Unser [35] shows that spline wavelets are attractive for coding applications based on approximation theoretic considerations. Experiments by Rioul [32] for orthogonal bases indicate that smoothness is an important consideration for compression. Experiments by Antonini *et al* [37] find that both vanishing moments and smoothness are important, and for the filters tested they found that smoothness appeared to be slightly more important than the number of vanishing moments. Nonetheless, Vetterli and Herley [38] state that "the importance of regularity for signal processing applications is still an open question." The bases most commonly used in practice have between one and two continuous derivatives. Additional smoothness does not appear to yield significant improvements in coding results.

Villasenor *et al* [39] have systematically examined all minimum order biorthogonal filter banks with lengths ≤ 36. In addition to the criteria already mentioned, [39] also examines measures of oscillatory behavior and of the sensitivity of the coarse-scale approximations $A^j f(x)$ to translations of the function $f(x)$. The best filter found in these experiments was a 7/9-tap spline variant with less dissimilar lengths from [37], and this filter is one of the most commonly used in wavelet coders.

There is one caveat with regard to the results of the filter evaluation in [39]. Villasenor *et al* compare peak signal to noise ratios generated by a simple transform coding scheme. The bit allocation scheme they use works well for orthogonal bases, but it can be improved upon considerably in the biorthogonal case. This inefficient bit allocation causes some promising biorthogonal filter sets to be overlooked.

For biorthogonal transforms, the squared error in the transform domain is not the same as the squared error in the original image. As a result, the problem of minimizing image error is considerably more difficult than in the orthogonal case. We can reduce image-domain errors by performing bit allocation using a weighted transform-domain error measure that we discuss in section 5.5. A number of other filters yield performance comparable to that of the 7/9 filter of [37] provided that we do bit allocation with a weighted error measure. One such basis is the Deslauriers-Dubuc interpolating wavelet of order 4 [40, 41], which has the advantage of having filter taps that are dyadic rationals. Both the spline wavelet of [37] and the order 4 Deslauriers-Dubuc wavelet have 4 vanishing moments in both $\psi(x)$ and$\tilde{\psi}(x)$, and the basis functions have just under 2 continuous derivatives in the L^2 sense.

One new very promising set of filters has been developed by Balasingham and Ramstad [42]. Their design procedure combines classical filter design

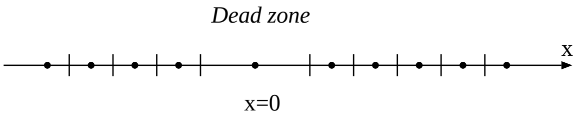


FIGURE 15. *Dead-zone quantizer, with larger encoder partition around $x = 0$ (dead zone) and uniform quantization elsewhere.*

techniques with ideas from wavelet constructions and yields filters that perform significantly better than the popular 7/9 filter set from [37].

5.2 Boundaries

Careful handling of image boundaries when performing the wavelet transform is essential for effective compression algorithms. Naive techniques for artificially extending images beyond given boundaries such as periodization or zero-padding lead to significant coding inefficiencies. For symmetrical wavelets an effective strategy for handling boundaries is to extend the image via reflection. Such an extension preserves continuity at the boundaries and usually leads to much smaller wavelet coefficients than if discontinuities were present at the boundaries. Brislawn [43] describes in detail procedures for non-expansive symmetric extensions of boundaries. An alternative approach is to modify the filter near the boundary. Boundary filters [44, 45] can be constructed that preserve filter orthogonality at boundaries. The lifting scheme [46] provides a related method for handling filtering near the boundaries.

5.3 Quantization

Most current wavelet coders employ scalar quantization for coding. There are two basic strategies for performing the scalar quantization stage. If we knew the distribution of coefficients for each subband in advance, the optimal strategy would be to use entropy-constrained Lloyd-Max quantizers for each subband. In general we do not have such knowledge, but we can provide a parametric description of coefficient distributions by sending side information. Coefficients in the high pass subbands of a wavelet transform are known *a priori* to be distributed as generalized Gaussians [27] centered around zero.

A much simpler quantizer that is commonly employed in practice is a uniform quantizer with a dead zone. The quantization bins, as shown in Figure 15, are of the form $[n\Delta, (n+1)\Delta)$ for $n \in \mathbb{Z}$ except for the central bin $[-\Delta, \Delta)$. Each bin is decoded to the value at its center in the simplest case, or to the centroid of the bin. In the case of asymptotically high rates, uniform quantization is optimal [47]. Although in practical regimes these dead-zone quantizers are suboptimal, they work almost as well as Lloyd-

Max coders when we decode to the bin centroids [48]. Moreover, dead-zone quantizers have the advantage that of being very low complexity and robust to changes in the distribution of coefficients in source. An additional advantage of these dead-zone quantizers is that they can be nested to produce an embedded bitstream following a procedure in [49].

5.4 Entropy Coding

Arithmetic coding provides a near-optimal entropy coding for the quantized coefficient values. The coder requires an estimate of the distribution of quantized coefficients. This estimate can be approximately specified by providing parameters for a generalized Gaussian or a Laplacian density. Alternatively the probabilities can be estimated online. Online adaptive estimation has the advantage of allowing coders to exploit local changes in image statistics. Efficient adaptive estimation procedures are discussed in [50] and [51].

Because images are not jointly Gaussian random processes, the transform coefficients, although decorrelated, still contain considerable structure. The entropy coder can take advantage of some of this structure by conditioning the encodings on previously encoded values. A coder of [49] obtains modest performance improvements using such a technique.

5.5 Bit Allocation

The final question we need to address is that of how finely to quantize each subband. As we discussed in Section 3.2, the general idea is to determine the number of bits b_j to devote to coding subband j so that the total distortion $\sum_j D_j(b_j)$ is minimized subject to the constraint that $\sum_j b_j \leq b$. Here $D_j(b)$ is the amount of distortion incurred in coding subband j with b bits. When the functions $D_j(b)$ are known in closed form we can solve the problem using the Kuhn-Tucker conditions. One common practice is to approximate the functions $D_j(b)$ with the rate-distortion function for a Gaussian random variable. However, this approximation is not very accurate at low bit rates. Better results may be obtained by measuring $D_j(b)$ for a range of values of b and then solving the constrained minimization problem using integer programming techniques. An algorithm of Shoham and Gersho [52] solves precisely this problem.

For biorthogonal wavelets we have the additional problem that squared error in the transform domain is not equal to squared error in the inverted image. Moulin [53] has formulated a multiscale relaxation algorithm which provides an approximate solution to the allocation problem for this case. Moulin's algorithm yields substantially better results than the naive approach of minimizing squared error in the transform domain.

A simpler approach is to approximate the squared error in the image by weighting the squared errors in each subband. The weight w_j for subband

j is obtained as follows: we set a single coefficient in subband j to 1 and set all other wavelet coefficients to zero. We then invert this transform. The weight w_j is equal to the sum of the squares of the values in the resulting inverse transform. We allocate bits by minimizing the *weighted* sum $\sum_j w_j D_j(b_j)$ rather than the sum $\sum_j D_j(b_j)$. Further details may be found in Naveen and Woods [54]. This weighting procedure results in substantial coding improvements when using wavelets that are not very close to being orthogonal, such as the Deslauriers-Dubuc wavelets popularized by the lifting scheme [46]. The 7/9 tap filter set of [37], on the other hand, has weights that are all nearly 1, so this weighting provides little benefit.

5.6 Perceptually Weighted Error Measures

Our goal in lossy image coding is to minimize visual discrepancies between the original and compressed images. Measuring visual discrepancy is a difficult task. There has been a great deal of research on this problem, but because of the great complexity of the human visual system, no simple, accurate, and mathematically tractable measure has been found.

Our discussion up to this point has focused on minimizing squared error distortion in compressed images primarily because this error metric is mathematically convenient. The measure suffers from a number of deficits, however. For example, consider two images that are the same everywhere except in a small region. Even if the difference in this small region is large and highly visible, the mean squared error for the whole image will be small because the discrepancy is confined to a small region. Similarly, errors that are localized in straight lines, such as the blocking artifacts produced by the discrete cosine transform, are much more visually objectionable than squared error considerations alone indicate.

There is evidence that the human visual system makes use of a multiresolution image representation; see [55] for an overview. The eye is much more sensitive to errors in low frequencies than in high. As a result, we can improve the correspondence between our squared error metric and perceived error by weighting the errors in different subbands according to the eye's contrast sensitivity in a corresponding frequency range. Weights for the commonly used 7/9-tap filter set of [37] have been computed by Watson et al in [56].

6 Extending the Transform Coder Paradigm

The basic wavelet coder discussed in Section 5 is based on the basic transform coding paradigm, namely decorrelation and compaction of energy into a few coefficients. The mathematical framework used in deriving the wavelet transform motivates compression algorithms that go beyond the traditional mechanisms used in transform coding. These important extensions are at the heart of modern wavelet coding algorithms of Sections 7 and 9. We take a moment here to discuss these extensions.

Conventional transform coding relies on energy compaction in an ordered set of transform coefficients, and quantizes those coefficients with a priority according to their order. This paradigm, while quite powerful, is based on several assumptions about images that are not always completely accurate. In particular, the Gaussian assumption breaks down for the joint distributions across image discontinuities. Mallat and Falzon [57] give the following example of how the Gaussian, high-rate analysis breaks down at low rates for non-Gaussian processes.

Let $Y[n]$ be a random N-vector defined by

$$Y[n] = \begin{cases} \mathrm{X} & \text{if } n = P \\ \mathrm{X} & \text{if } n = P + 1 (mod N) \\ 0 & \text{otherwise} \end{cases} \tag{1.29}$$

Here P is a random integer uniformly distributed between 0 and $N-1$ and X is a random variable that equals 1 or -1 each with probability $\frac{1}{2}$. X and P are independent. The vector Y has zero mean and a covariance matrix with entries

$$E\{Y[n]Y[m]\} = \begin{cases} \frac{2}{N} & \text{for } n = m \\ \frac{1}{N} & \text{for } |n-m| \in \{1, N-1\} \\ 0 & \text{otherwise} \end{cases} \tag{1.30}$$

The covariance matrix is circulant, so the KLT for this process is the simply the Fourier transform. The Fourier transform of Y is a very inefficient representation for coding Y. The energy at frequency k will be $|1+e^{2\pi i \frac{k}{N}}|^2$ which means that the energy of Y is spread out over the entire low-frequency half of the Fourier basis with some spill-over into the high-frequency half. The KLT has "packed" the energy of the two non-zero coefficients of Y into roughly $\frac{N}{2}$ coefficients. It is obvious that Y was much more compact in its original form, and could be coded better without transformation: Only two coefficients in Y are non-zero, and we need only specify the values of these coefficients and their positions.

As suggested by the example above, the essence of the extensions to traditional transform coding is the idea of selection operators. Instead of quantizing the transform coefficients in a pre-determined order of priority, the wavelet framework lends itself to improvements, through judicious choice of which elements to code. This is made possible primarily because wavelet basis elements are spatially as well as spectrally compact. In parts of the image where the energy is spatially but not spectrally compact (like the example above) one can use selection operators to choose subsets of the wavelet coefficients that represent that signal efficiently. A most notable example is the Zerotree coder and its variants (Section 7).

More formally, the extension consists of dropping the constraint of linear image approximations, as the selection operator is nonlinear. The work of DeVore *et al.* [58] and of Mallat and Falzon [57] suggests that at low rates, the problem of image coding can be more effectively addressed as a problem in obtaining a *non-linear* image approximation. This idea leads to some important differences in coder implementation compared to the linear framework. For linear approximations, Theorems 3.1 and 3.3 in Section 3.1 suggest that at low rates we should approximate our images using a fixed subset of the Karhunen-Loève basis vectors. We set a fixed set of transform coefficients to zero, namely the coefficients corresponding to the smallest eigenvalues of the covariance matrix. The non-linear approximation idea, on the other hand, is to approximate images using a subset of basis functions that are selected adaptively based on the given image. Information describing the particular set of basis functions used for the approximation, called a significance map, is sent as side information. In Section 7 we describe zerotrees, a very important data structure used to efficiently encode significance maps.

Our example suggests that a second important assumption to relax is that our images come from a single jointly Gaussian source. We can obtain better energy packing by optimizing our transform to the particular image at hand rather than to the global ensemble of images. The KLT provides efficient variance packing for vectors drawn from a single Gaussian source. However, if we have a mixture of sources the KLT is considerably less efficient. Frequency-adaptive and space/frequency-adaptive coders decompose images over a large library of different bases and choose an energy-packing transform that is adapted to the image itself. We describe these adaptive coders in Section 8.

Trellis coded quantization represents a more drastic departure from the transform coder framework. While TCQ coders operate in the transform domain, they effectively do not use scalar quantization. Trellis coded quantization captures not only correlation gain and fractional bitrates, but also the packing gain of VQ. In both performance and complexity, TCQ is essentially VQ in disguise.

The selection operator that characterizes the extension to the transform coder paradigm generates information that needs to be conveyed to the decoder as “side information”. This side information can be in the form of zerotrees, or more generally energy classes. Backward mixture estimation represents a different approach: it assumes that the side information is largely redundant and can be estimated from the causal data. By cutting down on the transmitted side information, these algorithms achieve a remarkable degree of performance and efficiency.

For reference, Table 1.1 provides a comparison of the peak signal to

TABLE 1.1. *Peak signal to noise ratios in decibels for coders discussed in the paper. Higher values indicate better performance.*

	Lena (bits/pixel)			Barbara (bits/pixel)		
Type of Coder	1.0	0.5	0.25	1.0	0.5	0.25
JPEG [59]	37.9	34.9	31.6	33.1	28.3	25.2
Optimized JPEG [60]	39.6	35.9	32.3	35.9	30.6	26.7
Baseline Wavelet [61]	39.4	36.2	33.2	34.6	29.5	26.6
Zerotree (Shapiro) [62]	39.6	36.3	33.2	35.1	30.5	26.8
Zerotree (Said & Pearlman) [63]	40.5	37.2	34.1	36.9	31.7	27.8
Zerotree (R/D optimized) [64]	40.5	37.4	34.3	37.0	31.3	27.2
Frequency-adaptive [65]	39.3	36.4	33.4	36.4	31.8	28.2
Space-frequency adaptive [66]	40.1	36.9	33.8	37.0	32.3	28.7
Frequency-adaptive + zerotrees [67]	40.6	37.4	34.4	37.7	33.1	29.3
TCQ subband [68]	41.1	37.7	34.3	–	–	–
Bkwd. mixture estimation (EQ) [69]	40.9	37.7	34.6	–	–	–

noise ratios for the coders we discuss in the paper.[8] The test images are the 512×512 Lena image and the 512×512 Barbara image. Figure 16 shows the Barbara image as compressed by JPEG, a baseline wavelet transform coder, and the zerotree coder of Said and Pearlman [63]. The Barbara image is particularly difficult to code, and we have compressed the image at a low rate to emphasize coder errors. The blocking artifacts produced by the discrete cosine transform are highly visible in the image on the top right. The difference between the two wavelet coded images is more subtle but quite visible at close range. Because of the more efficient coefficient encoding (to be discussed below), the zerotree-coded image has much sharper edges and better preserves the striped texture than does the baseline transform coder.

7 Zerotree Coding

The rate-distortion analysis of the previous sections showed that optimal bitrate allocation is achieved when the signal is divided into subbands such that each subband contains a "white" signal. It was also shown that for typical signals of interest, this leads to narrower bands in the low frequencies and wider bands in the high frequencies. Hence, wavelet transforms have very good energy compaction properties.

This energy compaction leads to efficient utilization of scalar quantizers. However, a cursory examination of the transform in Figure 17 shows that a significant amount of structure is present, particularly in the fine scale coefficients. Wherever there is structure, there is room for compression, and

[8] More current numbers may be found on the web at
`http://www.icsl.ucla.edu/~ipl/psnr_results.html`

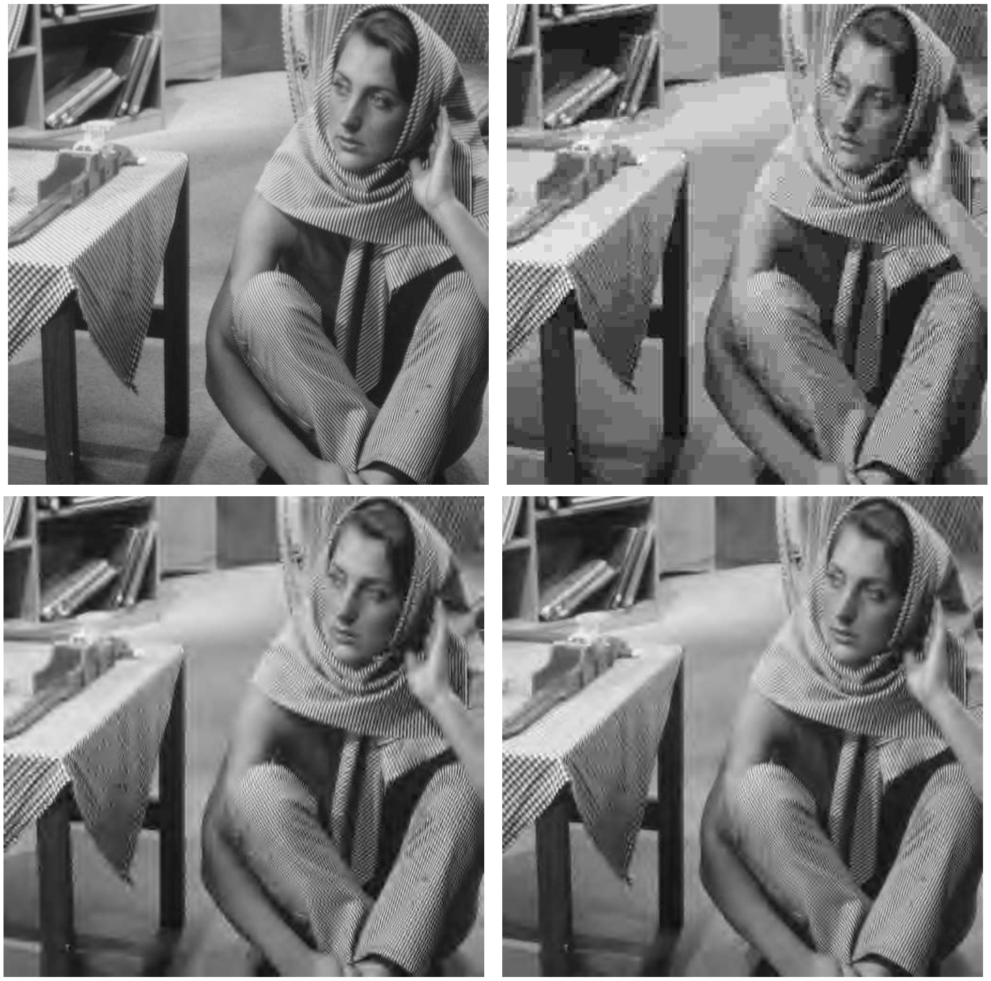


FIGURE 16. *Results of different compression schemes for the* 512×512 *Barbara test image at 0.25 bits per pixel. Top left: original image. Top right: baseline JPEG, PSNR = 24.4 dB. Bottom left: baseline wavelet transform coder [61], PSNR = 26.6 dB. Bottom right: Said and Pearlman zerotree coder, PSNR = 27.6 dB.*

advanced wavelet compression algorithms all address this structure in the higher frequency subbands.

One of the most prevalent approaches to this problem is based on exploiting the relationships of the wavelet coefficients across bands. A direct visual inspection indicates that large areas in the high frequency bands have little or no energy, and the small areas that have significant energy are similar in shape and location, across different bands. These high-energy areas stem from poor energy compaction close to the edges of the original image. Flat and slowly varying regions in the original image are well-described by the low-frequency basis elements of the wavelet transform (hence leading to high energy compaction). At the edge locations, however, low-frequency basis elements cannot describe the signal adequately, and some of the energy leaks into high-frequency coefficients. This happens similarly at all scales, thus the high-energy high-frequency coefficients representing the edges in the image have the same shape.

Our *a priori* knowledge that images of interest are formed mainly from flat areas, textures, and edges, allows us to take advantage of the resulting cross-band structure. Zerotree coders combine the idea of cross-band correlation with the notion of coding zeros jointly (which we saw previously in the case of JPEG), to generate very powerful compression algorithms.

The first instance of the implementation of zerotrees is due to Lewis and Knowles [70]. In their algorithm the image is represented by a tree-structured data construct (Figure 18). This data structure is implied by a dyadic discrete wavelet transform (Figure 19) in two dimensions. The root node of the tree represents the scaling function coefficient in the lowest frequency band, which is the parent of three nodes. Nodes inside the tree correspond to wavelet coefficients at a scale determined by their height in the tree. Each of these coefficients has four children, which correspond to the wavelets at the next finer scale having the same location in space. These four coefficients represent the four phases of the higher resolution basis elements at that location. At the bottom of the data structure lie the leaf nodes, which have no children.

Note that there exist three such quadtrees for each coefficient in the low frequency band. Each of these three trees corresponds to one of three filtering orderings: there is one tree consisting entirely of coefficients arising from horizontal high-pass, vertical low-pass operation (HL); one for horizontal low-pass, vertical high-pass (LH), and one for high-pass in both directions (HH).

The zerotree quantization model used by Lewis and Knowles was arrived at by observing that often when a wavelet coefficient is small, its children on the wavelet tree are also small. This phenomenon happens because significant coefficients arise from edges and texture, which are local. It is not difficult to see that this is a form of conditioning. Lewis and Knowles took this conditioning to the limit, and assumed that insignificant parent nodes always imply insignificant child nodes. A tree or subtree that contains (or

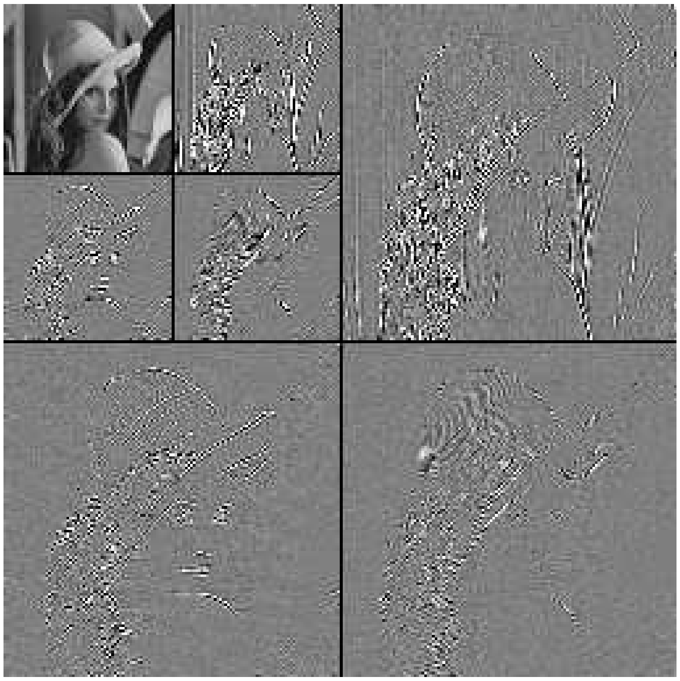


FIGURE 17. *Wavelet transform of the image "Lena."*

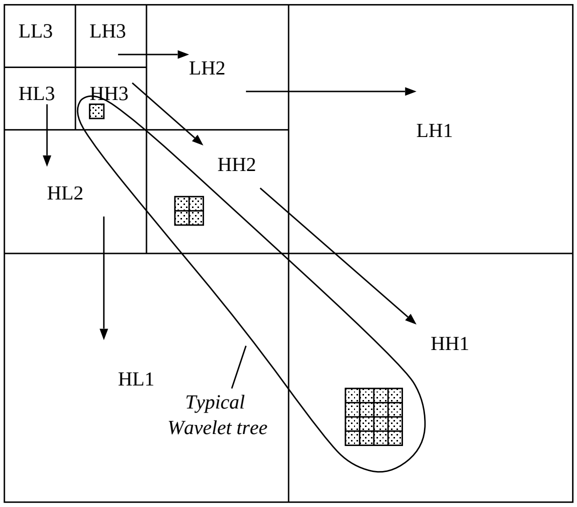


FIGURE 18. *Space-frequency structure of wavelet transform*

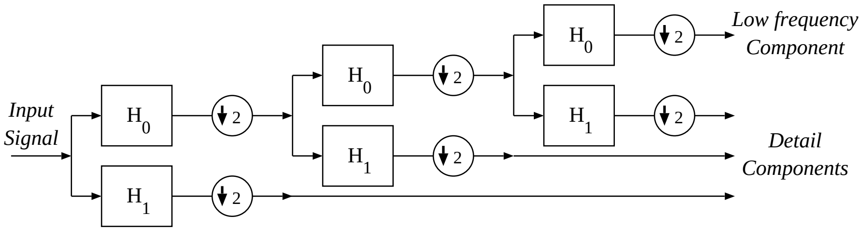


FIGURE 19. *Filter bank implementing a discrete wavelet transform*

is assumed to contain) only insignificant coefficients is known as a zerotree.

Lewis and Knowles used the following algorithm for the quantization of wavelet coefficients: Quantize each node according to an optimal scalar quantizer for the Laplacian density. If the node value is insignificant according to a pre-specified threshold, ignore all its children. These ignored coefficients will be decoded as zeros at the decoder. Otherwise, go to each of its four children and repeat the process. If the node was a leaf node and did not have a child, go to the next root node and repeat the process.

Aside from the nice energy compaction properties of the wavelet transform, the Lewis and Knowles coder achieves its compression ratios by joint coding of zeros. For efficient run-length coding, one needs to first find a conducive data structure, e.g. the zig-zag scan in JPEG. Perhaps the most significant contribution of this work was to realize that wavelet domain data provide an excellent context for run-length coding: not only are large run lengths of zeros generated, but also there is no need to transmit the length of zero runs, because they are assumed to automatically terminate at the leaf nodes of the tree. Much the same as in JPEG, this is a form of joint vector/scalar quantization. Each individual (significant) coefficient is quantized separately, but the symbols corresponding to small coefficients in fact are representing a vector consisting of that element and the zero run that follows it to the bottom of the tree.

While this compression algorithm generates subjectively acceptable images, its rate-distortion performance falls short of baseline JPEG, which at the time was often used for comparison purposes. The lack of sophistication in the entropy coding of quantized coefficients somewhat disadvantages this coder, but the main reason for its mediocre performance is the way it generates and recognizes zerotrees. As we have noted, whenever a coefficient is small, it is *likely* that its descendents are also insignificant. However, the Lewis and Knowles algorithm assumes that small parents *always* have small descendents, and therefore suffers large distortions when this does not hold because it zeros out large coefficients. The advantage of this method is that the detection of zerotrees is automatic: zerotrees are determined by measuring the magnitude of known coefficients. No side information is required to specify the locations of zerotrees, but this simplicity is obtained at the cost of reduced performance. More detailed analysis of this tradeoff gave rise to the next generation of zerotree coders.

7.1 The Shapiro and Said-Pearlman Coders

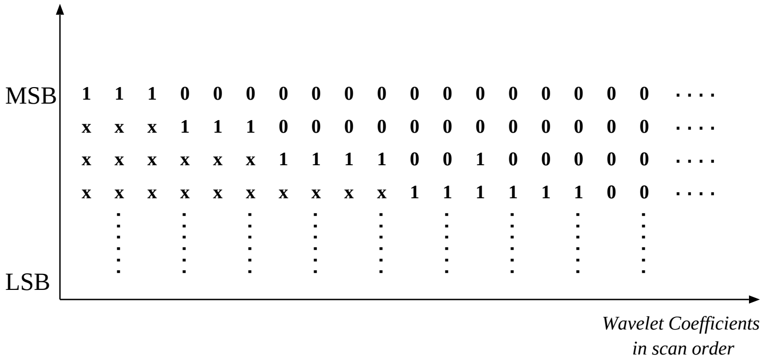
The Lewis and Knowles algorithm, while capturing the basic ideas inherent in many of the later coders, was incomplete. It had all the intuition that lies at the heart of more advanced zerotree coders, but did not efficiently specify significance maps, which is crucial to the performance of wavelet coders.

A significance map is a binary function whose value determines whether each coefficient is significant or not. If not significant, a coefficient is assumed to quantize to zero. Hence a decoder that knows the significance map needs no further information about that coefficient. Otherwise, the coefficient is quantized to a non-zero value. The method of Lewis and Knowles does not generate a significance map from the actual data, but uses one implicitly, based on *a priori* assumptions on the structure of the data. On the infrequent occasions when this assumption does not hold, a high price is paid in terms of distortion. The methods to be discussed below make use of the fact that, by using a small number of bits to correct mistakes in our assumptions about the occurrences of zerotrees, we can reduce the coded image distortion considerably.

The first algorithm of this family is due to Shapiro [71] and is known as the embedded zerotree wavelet (EZW) algorithm. Shapiro's coder was based on transmitting both the non-zero data and a significance map. The bits needed to specify a significance map can easily dominate the coder output, especially at lower bitrates. However, there is a great deal of redundancy in a general significance map for visual data, and the bitrates for its representation can be kept in check by conditioning the map values at each node of the tree on the corresponding value at the parent node. Whenever an insignificant parent node is observed, it is highly likely that the descendents are also insignificant. Therefore, most of the time, a "zerotree" significance map symbol is generated. But because p, the probability of this event, is close to 1, its information content, $-p \log p$, is very small. So most of the time, a very small amount of information is transmitted, and this keeps the average bitrate needed for the significance map relatively small.

Once in a while, one or more of the children of an insignificant node will be significant. In that case, a symbol for "isolated zero" is transmitted. The likelihood of this event is lower, and thus the bitrate for conveying this information is higher. But it is essential to pay this price to avoid losing significant information down the tree and therefore generating large distortions.

In summary, the Shapiro algorithm uses three symbols for significance maps: zerotree, isolated zero, or significant value. But using this structure, and by conditionally entropy coding these symbols, the coder achieves very



1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	....
x	x	x	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	....
x	x	x	x	x	x	1	1	1	1	0	0	1	0	0	0	0	0	....
x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	0	0	....

FIGURE 20. *Bit plane profile for raster scan ordered wavelet coefficients.*

good rate-distortion performance.

In addition, Shapiro's coder also generates an embedded code. Coders that generate embedded codes are said to have the *progressive transmission* or *successive refinement* property. Successive refinement consists of first approximating the image with a few bits of data, and then improving the approximation as more and more information is supplied. An embedded code has the property that for two given rates $R_1 > R_2$, the rate-R_2 code is a prefix to the rate-R_1 code. Such codes are of great practical interest for the following reasons:

- The encoder can easily achieve a precise bitrate by continuing to output bits when it reaches the desired rate.
- The decoder can cease decoding at any given point, generating an image that is the best representation possible with the decoded number of bits. This is of practical interest for broadcast applications where multiple decoders with varying computational, display, and bandwidth capabilities attempt to receive the same bitstream. With an embedded code, each receiver can decode the passing bitstream according to its particular needs and capabilities.
- Embedded codes are also very useful for indexing and browsing, where only a rough approximation is sufficient for deciding whether the image needs to be decoded or received in full. The process of screening images can be speeded up considerably by using embedded codes: after decoding only a small portion of the code, one knows if the target image is present. If not, decoding is aborted and the next image is requested, making it possible to screen a large number of images quickly. Once the desired image is located, the complete image is decoded.

Shapiro's method generates an embedded code by using a bit-slice approach (see Figure 20). First, the wavelet coefficients of the image are indexed into a one-dimensional array, according to their order of importance. This order places lower frequency bands before higher frequency bands since they have more energy, and coefficients within each band appear in a raster scan order. The bit-slice code is generated by scanning this one-dimensional array, comparing each coefficient with a threshold T. This initial scan provides the decoder with sufficient information to recover the most significant bit slice. In the next pass, our information about each coefficient is refined to a resolution of $T/2$, and the pass generates another bit slice of information. This process is repeated until there are no more slices to code.

Figure 20 shows that the upper bit slices contain a great many zeros because there are many coefficients below the threshold. The role of zerotree coding is to avoid transmitting all these zeros. Once a zerotree symbol is transmitted, we know that all the descendent coefficients are zero, so no information is transmitted for them. In effect, zerotrees are a clever form of run-length coding, where the coefficients are ordered in a way to generate longer run lengths (more efficient) as well as making the runs self-terminating, so the length of the runs need not be transmitted.

The zerotree symbols (with high probability and small code length) can be transmitted again and again for a given coefficient, until it rises above the sinking threshold, at which point it will be tagged as a significant coefficient. After this point, no more zerotree information will be transmitted for this coefficient.

To achieve embeddedness, Shapiro uses a clever method of encoding the sign of the wavelet coefficients with the significance information. There are also further details of the priority of wavelet coefficients, the bit-slice coding, and adaptive arithmetic coding of quantized values (entropy coding), which we will not pursue further in this review. The interested reader is referred to [71] for more details.

Said and Pearlman [72] have produced an enhanced implementation of the zerotree algorithm, known as Set Partitioning in Hierarchical Trees (SPHIT). Their method is based on the same premises as the Shapiro algorithm, but with more attention to detail. The public domain version of this coder is very fast, and improves the performance of EZW by 0.3-0.6 dB. This gain is mostly due to the fact that the original zerotree algorithms allow special symbols only for single zerotrees, while in reality, there are other sets of zeros that appear with sufficient frequency to warrant special symbols of their own. In particular, the Said-Pearlman coder provides symbols for combinations of parallel zerotrees.

Davis and Chawla [73] have shown that both the Shapiro and the Said and Pearlman coders are members of a large family of tree-structured significance mapping schemes. They provide a theoretical framework that explains in more detail the performance of these coders and describe an algorithm for selecting a member of this family of significance maps that is optimized for a given image or class of images.

7.2 Zerotrees and Rate-Distortion Optimization

In the previous coders, zerotrees were used only when they were detected in the actual data. But consider for the moment the following hypothetical example: assume that in an image, there is a wide area of little activity, so that in the corresponding location of the wavelet coefficients there exists a large group of insignificant values. Ordinarily, this would warrant the use of a big zerotree and a low expenditure of bitrate over that area. Suppose, however, that there is a one-pixel discontinuity in the middle of the area, such that at the bottom of the would-be zerotree, there is one significant coefficient. The algorithms described so far would prohibit the use of a zerotree for the entire area.

Inaccurate representation of a single pixel will change the average distortion in the image only by a small amount. In our example we can gain significant coding efficiency by ignoring the single significant pixel so that we can use a large zerotree. We need a way to determine the circumstances under which we should ignore significant coefficients in this manner.

The specification of a zerotree for a group of wavelet coefficient is a form of quantization. Generally, the values of the pixels we code with zerotrees are non-zero, but in using a zerotree we specify that they be decoded as zeros. Non-zerotree wavelet coefficients (significant values) are also quantized, using scalar quantizers. If we saves bitrate by specifying larger zerotrees, as in the hypothetical example above, the rate that was saved can be assigned to the scalar quantizers of the remaining coefficients, thus quantizing them more accurately. Therefore, we have a choice in allocating the bitrate among two types of quantization. The question is, if we are given a unit of rate to use in coding, where should it be invested so that the corresponding reduction in distortion is maximized?

This question, in the context of zerotree wavelet coding, was addressed by Xiong et al. [74], using well-known bit allocation techniques [1]. The central result for optimal bit allocation states that, in the optimal state, the slope of the operational rate-distortion curves of all quantizers are equal. This result is intuitive and easy to understand. The slope of the operational rate-distortion function for each quantizer tells us how many units of distortion we add/eliminate for each unit of rate we eliminate/add. If one of the quantizers has a smaller R-D slope, meaning that it is giving us less distortion reduction for our bits spent, we can take bits away from this quantizer (i.e. we can reduce its step size) and give them to the other, more efficient quantizers. We continue to do so until all quantizers have an equal slope.

Obviously, specification of zerotrees affects the quantization levels of non-zero coefficients because total available rate is limited. Conversely, specifying quantization levels will affect the choice of zerotrees because it affects the incremental distortion between zerotree quantization and scalar quantization. Therefore, an iterative algorithm is needed for rate-distortion optimization. In phase one, the uniform scalar quantizers are fixed, and optimal zerotrees are chosen. In phase two, zerotrees are fixed and the quantization level of uniform scalar quantizers is optimized. This algorithm is guaranteed to converge to a local optimum [74].

There are further details of this algorithm involving prediction and description of zerotrees, which we leave out of the current discussion. The advantage of this method is mainly in performance, compared to both EZW and SPHIT (the latter only slightly). The main disadvantages of this method are its complexity, and perhaps more importantly, that it does not generate an embedded bitstream.

8 Frequency, space-frequency adaptive coders

8.1 Wavelet Packets

The wavelet transform does a good job of decorrelating image pixels in practice, especially when images have power spectra that decay approximately uniformly and exponentially. However, for images with non-exponential rates of spectral decay and for images which have concentrated peaks in the spectra away from DC, we can do considerably better.

Our analysis of Section 3.5 suggests that the optimal subband decomposition for an image is one for which the spectrum in each subband is approximately flat. The octave-band decomposition produced by the wavelet transform produces nearly flat spectra for exponentially decaying spectra. The Barbara test image shown in Figure 16 contains a narrow-band component at high frequencies that comes from the tablecloth and the striped clothing. Fingerprint images contain similar narrow-band high frequency components.

The best basis algorithm, developed by Coifman and Wickerhauser [75], provides an efficient way to find a fast, wavelet-like transform that provides a good approximation to the Karhunen-Loève transform for a given image. As with the wavelet transform, we start by assuming that a given signal corresponds to a sum of fine-scale scaling functions. The transform performs a change of basis, but the new basis functions are not wavelets but rather *wavelet packets* [76].

Like wavelets, wavelet packets are formed from translated and dilated linear combinations of scaling functions. However, the recurrence relations they satisfy are different, and the functions form an overcomplete set. Consider a signal of length 2^N. The wavelet basis for such a signal consists of a scaling function and $2^N - 1$ translates and dilates of the wavelet $\psi(x)$. Wavelet packets are formed from translates and dilates of 2^N different prototype functions, and there are $N2^N$ different possible functions that can be used to form a basis.

Wavelet packets are formed from recurrence relations similar to those for wavelets and generalize the theoretical framework of wavelets. The simplest wavelet packet $\pi_0(x)$ is the scaling function $\phi(x)$. New wavelet packets $\pi_j(x)$ for $j > 0$ are formed by the recurrence relations

$$\begin{aligned} \pi_{2j}(x) &= \sum_k h_k \pi_j(2x-k) \qquad (1.31) \\ \pi_{2j+1}(x) &= \sum_k g_k \pi_j(2x-k). \qquad (1.32) \end{aligned}$$

where the h_k and g_k are the same as those in the recurrence equations (4.3) and (4.3).

The idea of wavelet packets is most easily seen in the frequency domain. Recall from Figure 14 that each step of the wavelet transform splits the current low frequency subband into two subbands of equal width, one high-pass and one low-pass. With wavelet packets there is a new degree of freedom in the transform. Again there are N stages to the transform for a signal of length 2^N, but at each stage we have the option of splitting the low-pass subband, the high-pass subband, both, or neither. The high and low pass filters used in each case are the same filters used in the wavelet transform. In fact, the wavelet transform is the special case of a wavelet packet transform in which we always split the low-pass subband. With this increased flexibility we can generate 2^N possible different transforms in 1-D. The possible transforms give rise to all possible dyadic partitions of the frequency axis. The increased flexibility does not lead to a large increase in complexity; the worst-case complexity for a wavelet packet transform is $O(N \log N)$.

8.2 Frequency Adaptive Coders

The *best basis algorithm* is a fast algorithm for minimizing an additive cost function over the set of all wavelet packet bases. Our analysis of transform coding for Gaussian random processes suggests that we select the basis that maximizes the transform coding gain. The approximation theoretic arguments of Mallat and Falzon [57] suggest that at low bit rates the basis that maximizes the number of coefficients below a given threshold is the best choice. The best basis paradigm can accommodate both of these choices. See [77] for an excellent introduction to wavelet packets and the best basis algorithm. Ramchandran and Vetterli [65] describe an algorithm for finding the best wavelet packet basis for coding a given image using rate-distortion criteria.

An important application of this wavelet-packet transform optimization is the FBI Wavelet/Scalar Quantization Standard for fingerprint compression. The standard uses a wavelet packet decomposition for the transform stage of the encoder [78]. The transform used is fixed for all fingerprints, however, so the FBI coder is a first-generation linear coder.

The benefits of customizing the transform on a per-image basis depend considerably on the image. For the Lena test image the improvement in peak signal to noise ratio is modest, ranging from 0.1 dB at 1 bit per pixel to 0.25 dB at 0.25 bits per pixel. This is because the octave band partitions of the spectrum of the Lena image are nearly flat. The Barbara image (see Figure 16), on the other hand, has a narrow-band peak in the spectrum at high frequencies. Consequently, the PSNR increases by roughly 2 dB over the same range of bitrates [65]. Further impressive gains are obtained by combining the adaptive transform with a zerotree structure [67].

8.3 Space-Frequency Adaptive Coders

The best basis algorithm is not limited only to adaptive segmentation of the frequency domain. Related algorithms permit joint time and frequency segmentations. The simplest of these algorithms performs adapted frequency segmentations over regions of the image selected through a quadtree decomposition procedure [79, 80]. More complicated algorithms provide combinations of spatially varying frequency decompositions and frequency varying spatial decompositions [66]. These jointly adaptive algorithms work particularly well for highly nonstationary images.

The primary disadvantage of these spatially adaptive schemes are that the pre-computation requirements are much greater than for the frequency adaptive coders, and the search is also much larger. A second disadvantage is that both spatial and frequency adaptivity are limited to dyadic partitions. A limitation of this sort is necessary for keeping the complexity manageable, but dyadic partitions are not in general the best ones.

9 Utilizing Intra-band Dependencies

The development of the EZW coder motivated a flurry of activity in the area of zerotree wavelet algorithms. The inherent simplicity of the zerotree data structure, its computational advantages, as well as the potential for generating an embedded bitstream were all very attractive to the coding community. Zerotree algorithms were developed for a variety of applications, and many modifications and enhancements to the algorithm were devised, as described in Section 7.

With all the excitement incited by the discovery of EZW, it is easy to automatically assume that zerotree structures, or more generally inter-band dependencies, should be the focal point of efficient subband image compression algorithms. However, some of the best performing subband image coders known today are not based on zerotrees. In this section, we explore two methods that utilize intra-band dependencies. One of them uses the concept of Trellis Coded Quantization (TCQ). The other uses both

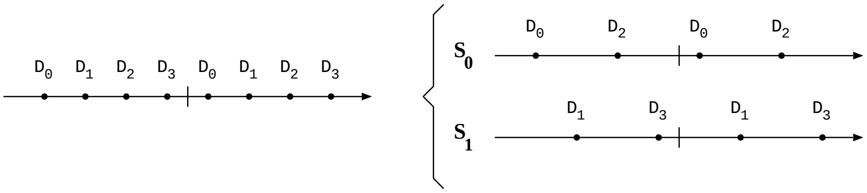


FIGURE 21. *TCQ sets and supersets*

inter- and intra-band information, and is based on a recursive estimation of the variance of the wavelet coefficients. Both of them yield excellent coding results.

9.1 Trellis coded quantization

Trellis Coded Quantization (TCQ) [81] is a fast and effective method of quantizing random variables. Trellis coding exploits correlations between variables. More interestingly, it can use non-rectangular quantizer cells that give it quantization efficiencies not attainable by scalar quantizers. The central ideas of TCQ grew out of the ground-breaking work of Ungerboeck [82] in trellis coded modulation. In this section we describe the operational principles of TCQ, mostly through examples. We will briefly touch upon variations and improvements on the original idea, especially at the low bitrates applicable in image coding. In Section 9.2, we review the use of TCQ in multiresolution image compression algorithms.

The basic idea behind TCQ is the following: Assume that we want to quantize a stationary, memoryless uniform source at the rate of R bits per sample. Performing quantization directly on this uniform source would require an optimum scalar quantizer with 2^N reproduction levels (symbols). The idea behind TCQ is to first quantize the source more finely, with 2^{R+k} symbols. Of course this would exceed the allocated rate, so we cannot have a free choice of symbols at all times.

In our example we take $k = 1$. The scalar codebook of 2^{R+1} symbols is partitioned into subsets of 2^{R-1} symbols each, generating four sets. In our example $R = 2$; see Figure 21. The subsets are designed such that each of them represents reproduction points of a coarser, rate-$(R - 1)$ quantizer. The four subsets are designated D_0, D_1, D_2, and D_3. Also, define $S_0 = D_0 \bigcup D_2$ and $S_1 = D_1 \bigcup D_3$, where S_0 and S_1 are known as *supersets*.

Obviously, the rate constraint prohibits the specification of an arbitrary symbol out of 2^{R+1} symbols. However, it is possible to exactly specify, with R bits, one element out of either S_0 or S_1. At each sample, assuming we know which one of the supersets to use, one bit can be used to determine the active subset, and $R - 1$ bits to specify a codeword from the subset. The choice of superset is determined by the state of a finite state machine, described by a suitable trellis. An example of such a trellis, with eight

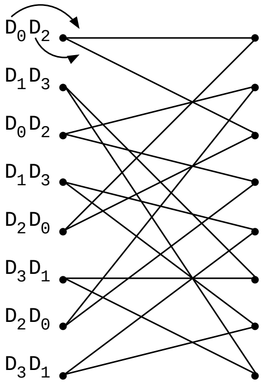


FIGURE 22. *An 8-state TCQ trellis with subset labeling. The bits that specify the sets within the superset also dictate the path through the trellis.*

states, is given in Figure 22. The subsets $\{D_0, D_1, D_2, D_3\}$ are also used to label the branches of the trellis, so the same bit that specifies the subset (at a given state) also determines the next state of the trellis.

Encoding is achieved by spending one bit per sample on specifying the path through the trellis, while the remaining $R-1$ bits specify a codeword out of the active subset. It may seem that we are back to a non-optimal rate-R quantizer (either S_0 or S_1). So why all this effort? The answer is that we have more codewords than a rate-R quantizer, because there is some freedom of choosing from symbols of either S_0 or S_1. Of course this choice is not completely free: the decision made at each sample is linked to decisions made at past and future sample points, through the permissible paths of the trellis. But it is this additional flexibility that leads to the improved performance. Availability of both S_0 and S_1 means that the reproduction levels of the quantizer are, in effect, allowed to "slide around" and fit themselves to the data, subject to the permissible paths on the trellis.

Before we continue with further developments of TCQ and subband coding, we should note that in terms of both efficiency and computational requirements, TCQ is much more similar to VQ than to scalar quantization. Since our entire discussion of transform coding has been motivated by an attempt to avoid VQ, what is the motivation for using TCQ in subband coding, instead of standard VQ? The answer lies in the recursive structure of trellis coding and the existence of a simple dynamic programming method, known as the Viterbi algorithm [83], for finding the TCQ codewords. Although it is true that block quantizers, such as VQ, are asymptotically as efficient as TCQ, the process of approaching the limit is far from trivial for VQ. For a given realization of a random process, the code vectors generated by the VQ of size $N-1$ have no clear relationship to those with vector dimension N. In contrast, the trellis encoding algorithm increases the dimensionality of the problem automatically by increasing the length of the trellis.

The standard version of TCQ is not particularly suitable for image coding, because its performance degrades quickly at low rates. This is due partially to the fact that one bit per sample is used to encode the trellis alone, while interesting rates for image coding are mostly below one bit per sample. Entropy constrained TCQ (ECTCQ) improves the performance of TCQ at low rates. In particular, a version of ECTCQ due to Marcellin [84] addresses two key issues: reducing the rate used to represent the trellis (the so-called "state entropy"), and ensuring that zero can be used as an output codeword with high probability. The codebooks are designed using the algorithm and encoding rule from [85].

9.2 TCQ subband coders

In a remarkable coincidence, at the 1994 International Conference in Image Processing, three research groups [86, 87, 88] presented similar but independently developed image coding algorithms. The main ingredients of the three methods are subband decomposition, classification and optimal rate allocation to different subsets of subband data, and entropy-constrained TCQ. These works have been brought together in [68]. We briefly discuss the main aspects of these algorithms.

Consider a subband decomposition of an image, and assume that the subbands are well represented by a *non-stationary* random process X, whose samples X_i are taken from distributions with variances σ_i^2. One can compute an "average variance" over the entire random process and perform conventional optimal quantization. But better performance is possible by sending overhead information about the variance of each sample, and quantizing it optimally according to its own p.d.f.

This basic idea was first proposed by Chen and Smith [89] for adaptive quantization of DCT coefficients. In their paper, Chen and Smith proposed to divide all DCT coefficients into four groups according to their "activity level", i.e. variance, and code each coefficient with an optimal quantizer designed for its group. The question of how to partition coefficients into groups was not addressed, however, and [89] arbitrarily chose to form groups with equal population.[9]

However, one can show that equally populated groups are not a always

[9] If for a moment, we disregard the overhead information, the problem of partitioning the coefficients bears a strong resemblance to the problem of best linear transform. Both operations, namely the linear transform and partitioning, conserve energy. The goal in both is to minimize overall distortion through optimal allocation of a finite rate. Not surprisingly, the solution techniques are similar (Lagrange multipliers), and they both generate sets with maximum separation between low and high energies (maximum arithmetic to geometric mean ratio).

a good choice. Suppose that we want to classify the samples into J groups, and that all samples assigned to a given class $i \in \{1, ..., J\}$ are grouped into a source X_i. Let the total number of samples assigned to X_i be N_i, and the total number of samples in all groups be N. Define $p_i = N_i/N$ to be the probability of a sample belonging to the source X_i. Encoding the source X_i at rate R_i results in a mean squared error distortion of the form [90]

$$D_i(R_i) = \epsilon_i^2 \, \sigma_i^2 \, 2^{-2R_i} \tag{1.33}$$

where ϵ_i is a constant depending on the shape of the pdf. The rate allocation problem can now be solved using a Lagrange multiplier approach, much in the same way as was shown for optimal linear transforms, resulting in the following optimal rates:

$$R_i = \frac{R}{J} + \frac{1}{2} \log_2 \frac{\epsilon_i^2 \, \sigma_i^2}{\prod_{j=1}^{J} (\epsilon_j^2 \, \sigma_j^2)^{p_j}} \tag{1.34}$$

where R is the total rate and R_i are the rates assigned to each group. *Classification gain* is defined as the ratio of the quantization error of the original signal X, divided by that of the optimally bit-allocated classified version.

$$G_c = \frac{\epsilon^2 \, \sigma^2}{\prod_{j=1}^{J} (\epsilon_j^2 \, \sigma_j^2)^{p_j}} \tag{1.35}$$

One aims to maximize this gain over $\{p_i\}$. It is not unexpected that the optimization process can often yield non-uniform $\{p_i\}$, resulting in unequal population of the classification groups. It is noteworthy that non-uniform populations not only have better classification gain in general, but also lower overhead: Compared to a uniform $\{p_i\}$, any other distribution has smaller entropy, which implies smaller side information to specify the classes.

The classification gain is defined for X_i taken from one subband. A generalization of this result in [68] combines it with the conventional *coding gain* of the subbands. Another refinement takes into account the side information required for classification. The coding algorithm then optimizes the resulting expression to determine the classifications. ECTCQ is then used for final coding.

Practical implementation of this algorithm requires attention to a great many details, for which the interested reader is referred to [68]. For example, the classification maps determine energy levels of the signal, which are related to the location of the edges in the image, and are thus related in different subbands. A variety of methods can be used to reduce the overhead information (in fact, the coder to be discussed in the next section makes the management of side information the focus of its efforts) Other issues include alternative measures for classification, and the usage of arithmetic coded TCQ. The coding results of the ECTCQ based subband coding are some of the best currently available in the literature, although the computational complexity of these algorithms is also considerably greater than the other methods presented in this paper.

9.3 Mixture Modeling and Estimation

A common thread in successful subband and wavelet image coders is modeling of image subbands as random variables drawn from a mixture of distributions. For each sample, one needs to detect which p.d.f. of the mixture it is drawn from, and then quantize it according to that pdf. Since the decoder needs to know which element of the mixture was used for encoding, many algorithms send side information to the decoder. This side information becomes significant, especially at low bitrates, so that efficient management of it is pivotal to the success of the image coder.

All subband and wavelet coding algorithms discussed so far use this idea in one way or another. They only differ in the constraints they put on side information so that it can be coded efficiently. For example, zerotrees are a clever way of indicating side information. The data is assumed from a mixture of very low energy (zero set) and high energy random variables, and the zero sets are assumed to have a tree structure.

The TCQ subband coders discussed in the last section also use the same idea. Different classes represent different energies in the subbands, and are transmitted as overhead. In [68], several methods are discussed to compress the side information, again based on geometrical constraints on the constituent elements of the mixture (energy classes).

A completely different approach to the problem of handling information overhead is explored in [69, 91]. These two works were developed simultaneously but independently. The version developed in [69] is named Estimation Quantization (EQ) by the authors, and is the one that we present in the following. The title of [91] suggests a focus on entropy coding, but in fact the underlying ideas of the two are remarkably similar. We will refer to the the aggregate class as *backward mixture-estimation encoding* (BMEE).

BMEE models the wavelet subband coefficients as non-stationary generalized Gaussian, whose non-stationarity is manifested by a slowly varying variance (energy) in each band. Because the energy varies slowly, it can be predicted from causal neighboring coefficients. Therefore, unlike previous methods, BMEE does not send the bulk of mixture information as overhead, but attempts to recover it at the decoder from already transmitted data, hence the designation "backward". BMEE assumes that the causal neighborhood of a subband coefficient (including parents in a subband tree) has the same energy (variance) as the coefficient itself. The estimate of energy is found by applying a maximum likelihood method to a training set formed by the causal neighborhood.

Similar to other recursive algorithms that involve quantization, BMEE has to contend with the problem of stability and drift. Specifically, the decoder has access only to quantized coefficients, therefore the estimator of energy at the encoder can only use quantized coefficients. Otherwise, the estimates at the encoder and decoder will vary, resulting in drift problems. This presents the added difficulty of estimating variances from *quantized* causal coefficients. BMEE incorporates the quantization of the coefficients into the maximum likelihood estimation of the variance.

The quantization itself is performed with a dead-zone uniform quantizer (see Figure 15). This quantizer offers a good approximation to entropy constrained quantization of generalized Gaussian signals. The dead-zone and step sizes of the quantizers are determined through a Lagrange multiplier optimization technique, which was introduced in the section on optimal rate allocation. This optimization is performed offline, once each for a variety of encoding rates and shape parameters, and the results are stored in a look-up table. This approach is to be credited for the speed of the algorithm, because no optimization need take place at the time of encoding the image.

Finally, the backward nature of the algorithm, combined with quantization, presents another challenge. All the elements in the causal neighborhood may sometimes quantize to zero. In that case, the current coefficient will also quantize to zero. This degenerate condition will propagate through the subband, making all coefficients on the causal side of this degeneracy equal to zero. To avoid this condition, BMEE provides for a mechanism to send side information to the receiver, whenever all neighboring elements are zero. This is accomplished by a preliminary pass through the coefficients, where the algorithm tries to "guess" which one of the coefficients will have degenerate neighborhoods, and assembles them to a set. From this set, a generalized Gaussian variance and shape parameter is computed and transmitted to the decoder. Every time a degenerate case happens, the encoder and decoder act based on this extra set of parameters, instead of using the backward estimation mode.

The BMEE coder is very fast, and especially in the low bitrate mode (less than 0.25 bits per pixel) is extremely competitive. This is likely to motivate a re-visitation of the role of side information and the mechanism of its transmission in wavelet coders.

10 Future Trends

Current research in image coding is progressing along a number of fronts. At the most basic level, a new interpretation of the wavelet transform has appeared in the literature. This new theoretical framework, called the lifting scheme [41], provides a simpler and more flexible method for designing wavelets than standard Fourier-based methods. New families of nonseparable wavelets constructed using lifting have the potential to improve coders. One very intriguing avenue for future research is the exploration of the nonlinear analogs of the wavelet transform that lifting makes possible.

The area of classification and backward estimation based coders is an active one. Several research groups are reporting promising results [92, 93].

One very promising research direction is the development of coded images that are robust to channel noise via joint source and channel coding. See for example [94], [95] and [96].

The adoption of wavelet based coding to video signals presents special challenges. One can apply 2-D wavelet coding in combination to temporal prediction (motion estimated prediction), which will be a direct counterpart of current DCT-based video coding methods. It is also possible to consider the video signal as a three-dimensional array of data and attempt to compress it with 3-D wavelet analysis. This approach presents difficulties that arise from the fundamental properties of the discrete wavelet transform. The discrete wavelet transform (as well as any subband decomposition) is a space-varying operator, due to the presence of decimation and interpolation. This space variance is not conducive to compact representation of video signals, as described below.

Video signals are best modeled by 2-D projections whose position in consecutive frames of the video signal varies by unknown amounts. Because vast amounts of information are repeated in this way, one can achieve considerable gain by representing the repeated information only once. This is the basis of motion compensated coding. However, since the wavelet representation of the same 2-D signal will vary once it is shifted[10], this redundancy is difficult to reproduce in the wavelet domain. A frequency domain study of the difficulties of 3-D wavelet coding of video is presented in [97], and leads to the same insights. Some attempts have also been made on applying 3-D wavelet coding on the residual 3-D data after motion compensation, but have met with indifferent success.

11 Summary and Conclusion

Image compression is governed by the general laws of information theory and specifically rate-distortion theory. However, these general laws are nonconstructive and the more specific techniques of quantization theory are needed for the actual development of compression algorithms.

Vector quantization can theoretically attain the maximum achievable coding efficiency. However, VQ has three main impediments: computational complexity, delay, and the curse of dimensionality. Transform coding techniques, in conjunction with entropy coding, capture important gains of VQ,

[10] Unless the shift is exactly by a correct multiple of M samples, where M is the downsampling rate

while avoiding most of its difficulties.

Theoretically, the Karhunen-Loéve transform is optimal for Gaussian processes. Approximations to the K-L transform, such as the DCT, have led to very successful image coding algorithms such as JPEG. However, even if one argues that image pixels can be individually Gaussian, they cannot be assumed to be jointly Gaussian, at least not across the image discontinuities. Image discontinuities are the place where traditional coders spend the most rate, and suffer the most distortion. This happens because traditional Fourier-type transforms (e.g., DCT) disperse the energy of discontinuous signals across many coefficients, while the compaction of energy in the transform domain is essential for good coding performance.

The discrete wavelet transform provides an elegant framework for signal representation in which both smooth areas and discontinuities can be represented compactly in the transform domain. This ability comes from the multi-resolution properties of wavelets. One can motivate wavelets through spectral partitioning arguments used in deriving optimal quantizers for Gaussian processes. However, the usefulness of wavelets in compression goes beyond the Gaussian case.

State of the art wavelet coders assume that image data comes from a source with fluctuating variance. Each of these coders provides a mechanism to express the local variance of the wavelet coefficients, and quantizes the coefficients optimally or near-optimally according to that variance. The individual wavelet coders vary in the way they estimate and transmit this variances to the decoder, as well as the strategies for quantizing according to that variance.

Zerotree coders assume a two-state structure for the variances: either negligible (zero) or otherwise. They send side information to the decoder to indicate the positions of the non-zero coefficients. This process yields a non-linear image approximation rather than the linear truncated KLT-based approximation motivated by our Gaussian model. The set of zero coefficients are expressed in terms of wavelet trees (Lewis & Knowles, Shapiro, others) or combinations thereof (Said & Pearlman). The zero sets are transmitted to the receiver as overhead, as well as the rest of the quantized data. Zerotree coders rely strongly on the dependency of data across scales of the wavelet transform.

Frequency-adaptive coders improve upon basic wavelet coders by adapting transforms according to the local inter-pixel correlation structure within an image. Local fluctuations in the correlation structure and in the variance can be addressed by spatially adapting the transform and by augmenting the optimized transforms with a zerotree structure.

Other wavelet coders use dependency of data within the bands (and sometimes across the bands as well). Coders based on Trellis Coded Quantization (TCQ) partition coefficients into a number of groups, according to their energy. For each coefficient, they estimate and/or transmit the group information as well as coding the value of the coefficient with TCQ, according to the nominal variance of the group. Another newly developed class of coders transmit only minimal variance information while achieving impressive coding results, indicating that perhaps the variance information is more redundant than previously thought.

While some of these coders may not employ what might strictly be called a wavelet transform, they all utilize a multi-resolution decomposition, and use concepts that were motivated by wavelet theory. Wavelets and the ideas arising from wavelet analysis have had an indelible effect on the theory and practice of image compression, and are likely to continue their dominant presence in image coding research in the near future.

Acknowledgments: G. Davis thanks the Digital Signal Processing group at Rice University for their generous hospitality during the writing of this paper. This work has been supported in part by a Texas Instruments Visiting Assistant Professorship at Rice University and an NSF Mathematical Sciences Postdoctoral Research Fellowship.

12 References

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

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

[3] R. M. Gray, J. C. Kieffer, and Y. Linde, "Locally optimal block quantizer design," *Information and Control*, vol. 45, pp. 178–198, May 1980.

[4] K. Zeger, J. Vaisey, and A. Gersho, "Globally optimal vector quantizer design by stochastic relaxation," *IEEE Transactions on Signal Processing*, vol. 40, pp. 310–322, Feb. 1992.

[5] T. Cover and J. Thomas, *Elements of Information Theory.* New York: John Wiley & Sons, Inc., 1991.

[6] A. Gersho, "Asymptotically optimal block quantization," *IEEE Transactions on Information Theory*, vol. IT-25, pp. 373–380, July 1979.

[7] L. F. Toth, "Sur la representation d'une population infinie par un nombre fini d'elements," *Acta Math. Acad. Scient. Hung.*, vol. 10, pp. 299–304, 1959.

[8] P. L. Zador, "Asymptotic quantization error of continuous signals and the quantization dimension," *IEEE Transactions on Information Theory*, vol. IT-28, pp. 139–149, Mar. 1982.

[9] J. Y. Huang and P. M. Schultheiss, "Block quantization of correlated Gaussian random variables," *IEEE Transactions on Communications*, vol. CS-11, pp. 289–296, September 1963.

[10] R. A. Horn and C. R. Johnson, *Matrix Analysis.* New York: Cambridge University Press, 1990.

[11] N. Ahmed, T. Natarajan, and K. R. Rao, "Descrete cosine transform," *IEEE Transactions on Computers*, vol. C-23, pp. 90–93, January 1974.

[12] K. Rao and P. Yip, *The Discrete Cosine Transform.* New York: Academic Press, 1990.

[13] W. Pennebaker and J. Mitchell, *JPEG still image data compression standard.* New York: Van Nostrad Reinhold, 1993.

[14] Y. Arai, T. Agui, and M. Nakajima, "A fast DCT-SQ scheme for images," *Transactions of the IEICE*, vol. 71, p. 1095, November 1988.

[15] E. Feig and E. Linzer, "Discrete cosine transform algorithms for image data compression," in *Electronic Imaging '90 East*, (Boston, MA), pp. 84–87, October 1990.

[16] A. Croisier, D. Esteban, and C. Galand, "Perfect channel splitting by use of interpolation/decimation/tree decomposition techniques," in *Proc. Int. Symp. on Information, Circuits and Systems*, (Patras, Greece), 1976.

[17] P. Vaidyanathan, "Theory and design of M-channel maximally decimated quadrature mirror filters with arbitrary M, having perfect reconstruction property," *IEEE Trans. ASSP*, vol. ASSP-35, pp. 476–492, April 1987.

[18] M. J. T. Smith and T. P. Barnwell, "A procedure for desiging exact reconstruction fiolter banks for tree structured subband coders," in *Proc. ICASSP*, (San Diego, CA), pp. 27.1.1–27.1.4, March 1984.

[19] M. J. T. Smith and T. P. Barnwell, "Exact reconstruction techniques for tree-structured subband coders," *IEEE Trans. ASSP*, vol. ASSP-34, pp. 434–441, June 1986.

[20] M. Vetterli, "Splitting a signal into subband channels allowing perfect reconstruction," in *Proc. IASTED Conf. Appl. Signal Processing*, (Paris, France), June 1985.

[21] M. Vetterli, "Filter banks allowing perfect reconstruction," *Signal Processing*, vol. 10, pp. 219–244, April 1986.

[22] P. Vaidyanathan, *Multirate systems and filter banks.* Englewood Cliffs, NJ: Prentice Hall, 1993.

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

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

[25] B. Girod, F. Hartung, and U. Horn, "Subband image coding," in *Subband and wavelet transforms: design and applications*, Boston, MA: Kluwer Academic Publishers, 1995.

[26] P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. COM-31, Apr. 1983.

[27] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, pp. 674–693, July 1989.

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

[29] C. S. Burrus, R. A. Gopinath, and H. Guo, *Introduction to Wavelets and Wavelet Transforms: A Primer.* Englewood Cliffs, NJ: Prentice-Hall, 1997.

[30] I. Daubechies, *Ten Lectures on Wavelets.* Philadelphia, PA: SIAM, 1992.

[31] O. Rioul, "Simple regularity criteria for subdivision schemes," *SIAM J. Math. Analysis*, vol. 23, pp. 1544–1576, Nov. 1992.

[32] O. Rioul, "Regular wavelets: a discrete-time approach," *IEEE Transactions on Signal Processing*, vol. 41, pp. 3572–3579, Dec. 1993.

[33] G. Strang and G. Fix, "A Fourier analysis of the finite element variational method," *Constructive Aspects of Functional Analysis*, pp. 796–830, 1971.

[34] W. Sweldens and R. Piessens, "Quadrature formulae and asymptotic error expansions for wavelet approximations of smooth functions," *SIAM Journal of Numerical Analysis*, vol. 31, pp. 1240–1264, Aug. 1994.

[35] M. Unser, "Approximation power of biorthogonal wavelet expansions," *IEEE Transactions on Signal Processing*, vol. 44, pp. 519–527, Mar. 1996.

[36] J. Kovačević and W. Sweldens, "Interpolating filter banks and wavelets in arbitrary dimensions," tech. rep., Lucent Technologies, Murray Hill, NJ, 1997.

[37] M. Antonini, M. Barlaud, and P. Mathieu, “Image Coding Using Wavelet Transform,” *IEEE Trans. Image Proc.*, vol. 1, pp. 205–220, Apr. 1992.

[38] M. Vetterli and C. Herley, “Wavelets and filter banks: Theory and design,” *IEEE Trans. Acoust. Speech Signal Proc.*, vol. 40, no. 9, pp. 2207–2232, 1992.

[39] J. Villasenor, B. Belzer, and J. Liao, “Wavelet filter evaluation for image compression,” *IEEE Transactions on image processing*, vol. 2, pp. 1053–1060, Aug. 1995.

[40] G. Deslauriers and S. Dubuc, “Symmetric iterative interpolation processes,” *Constructive Approximation*, vol. 5, no. 1, pp. 49–68, 1989.

[41] W. Sweldens, “The lifting scheme: A new philosophy in biorthogonal wavelet constructions,” in *Wavelet Applications in Signal and Image Processing III* (A. F. Laine and M. Unser, eds.), pp. 68–79, Proc. SPIE 2569, 1995.

[42] I. Balasingham and T. A. Ramstad, “On the relevance of the regularity constraint in subband image coding,” in *Proc. Asilomar Conference on Signals, Systems, and Computers*, (Pacific Grove), 1997.

[43] C. M. Brislawn, “Classification of nonexpansive symmetric extension transforms for multirate filter banks,” *Applied and Comp. Harmonic Analysis*, vol. 3, pp. 337–357, 1996.

[44] C. Herley and M. Vetterli, “Orthogonal time-varying filter banks and wavelets,” in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 1, pp. 391–394, May 1993.

[45] C. Herley, “Boundary filters for finite-length signals and time-varying filter banks,” *IEEE Trans. Circuits and Systems II*, vol. 42, pp. 102–114, Feb. 1995.

[46] W. Sweldens and P. Schröder, “Building your own wavelets at home,” Tech. Rep. 1995:5, Industrial Mathematics Initiative, Mathematics Department, University of South Carolina, 1995.

[47] H. Gish and J. N. Pierce, “Asymptotically efficient quantizing,” *IEEE Transactions on Information Theory*, vol. IT-14, pp. 676–683, Sept. 1968.

[48] N. Farvardin and J. W. Modestino, “Optimum quantizer performance for a class of non-Gaussian memoryless sources,” *IEEE Transactions on Information Theory*, vol. 30, pp. 485–497, May 1984.

[49] D. Taubman and A. Zakhor, “Multirate 3-D subband coding of video,” *IEEE Trans. Image Proc.*, vol. 3, Sept. 1994.

[50] T. Bell, J. G. Cleary, and I. H. Witten, *Text Compression.* Englewood Cliffs, NJ: Prentice Hall, 1990.

[51] D. L. Duttweiler and C. Chamzas, “Probability estimation in arithmetic and adaptive-Huffman entropy coders,” *IEEE Transactions on Image Processing*, vol. 4, pp. 237–246, Mar. 1995.

[52] Y. Shoham and A. Gersho, “Efficient bit allocation for an arbitrary set of quantizers,” *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 36, pp. 1445–1453, Sept. 1988.

[53] P. Moulin, “A multiscale relaxation algorithm for SNR maximization in nonorthogonal subband coding,” *IEEE Transactions on Image Processing*, vol. 4, pp. 1269–1281, Sept. 1995.

[54] J. W. Woods and T. Naveen, “A filter based allocation scheme for subband compression of HDTV,” *IEEE Trans. Image Proc.*, vol. IP-1, pp. 436–440, July 1992.

[55] B. A. Wandell, *Foundations of Vision.* Sunderland, MA: Sinauer Associates, 1995.

[56] A. Watson, G. Yang, J. Soloman, and J. Villasenor, “Visual thresholds for wavelet quantization error,” in *Proceedings of the SPIE*, vol. 2657, pp. 382–392, 1996.

[57] S. Mallat and F. Falzon, “Understanding image transform codes,” in *Proc. SPIE Aerospace Conf.*, (orlando), Apr. 1997.

[58] R. A. DeVore, B. Jawerth, and B. J. Lucier, “Image Compression through Wavelet Transform Coding,” *IEEE Trans. Info. Theory*, vol. 38, pp. 719–746, Mar. 1992.

[59] W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard.* New York: Van Nostrand Reinhold, 1992.

[60] M. Crouse and K. Ramchandran, “Joint thresholding and quantizer selection for decoder-compatible baseline JPEG,” in *Proc. ICASSP*, May 1995.

[61] G. M. Davis, “The wavelet image compression construction kit.” http://www.cs.dartmouth.edu/∼gdavis/wavelet/wavelet.html, 1996.

[62] J. Shapiro, “Embedded image coding using zerotrees of wavelet coefficients,” *IEEE Transactions on Signal Processing*, vol. 41, pp. 3445–3462, Dec. 1993.

[63] A. Said and W. A. Pearlman, “A new, fast, and efficient image codec based on set partitioning in hierarchichal trees,” *IEEE Trans. Circuits and Systems for Video Technology*, vol. 6, pp. 243–250, June 1996.

[64] Z. Xiong, K. Ramchandran, and M. T. Orchard, “Space-frequency quantization for wavelet image coding,” *preprint*, 1995.

[65] K. Ramchandran and M. Vetterli, “Best wavelet packet bases in a rate-distortion sense,” *IEEE Transactions on Image Processing*, vol. 2, no. 2, pp. 160–175, 1992.

[66] 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*, Sept. 1997.

[67] Z. Xiong, K. Ramchandran, M. Orchard, and K. Asai, “Wavelet packets-based image coding using joint space-frequency quantization,” *Preprint*, 1996.

[68] R. L. Joshi, H. Jafarkhani, J. H. Kasner, T. R. Fisher, N. Farvardin, and M. W. Marcellin, “Comparison of different methods of classification in subband coding of images,” *IEEE Transactions on Image Processing*, submitted.

[69] 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 *Proc. Data Compression Conference*, (Snowbird, Utah), pp. 221–230, 1997.

[70] A. S. Lewis and G. Knowles, “Image compression using the 2-d wavelet transform,” *IEEE Transactions on Image Processing*, vol. 1, pp. 244–250, April 1992.

[71] J. Shapiro, “Embedded image coding using zero-trees of wavelet coefficients,” *IEEE Transactions on Signal Processing*, vol. 41, pp. 3445–3462, December 1993.

[72] A. Said and W. A. Pearlman, “A new, fast, and efficient image codec based on set partitioning in hierarchical trees,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, pp. 243–250, June 1996.

[73] G. M. Davis and S. Chawla, “Image coding using optimized significance tree quantization,” in *Proc. Data Compression Conference* (J. A. Storer and M. Cohn, eds.), pp. 387–396, Mar. 1997.

[74] Z. Xiong, K. Ramchandran, and M. T. Orchard, “Space-frequency quantization for wavelet image coding,” *IEEE Transactions on Image Processing*, vol. 6, pp. 677–693, May 1997.

[75] R. R. Coifman and M. V. Wickerhauser, “Entropy based algorithms for best basis selection,” *IEEE Transactions on Information Theory*, vol. 32, pp. 712–718, Mar. 1992.

[76] R. R. Coifman and Y. Meyer, “Nouvelles bases orthonormées de $l^2(\mathbf{r})$ ayant la structure du système de Walsh,” Tech. Rep. Preprint, Department of Mathematics, Yale University, 1989.

[77] M. V. Wickerhauser, *Adapted Wavelet Analysis from Theory to Software.* Wellesley, MA: A. K. Peters, 1994.

[78] C. J. I. Services, *WSQ Gray-Scale Fingerprint Image Compression Specification (ver. 2.0).* Federal Bureau of Investigation, Feb. 1993.

[79] C. Herley, J. Kovačević, 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*, vol. 41, pp. 3341–3359, Dec. 1993.

[80] J. R. Smith and S. F. Chang, “Frequency and spatially adaptive wavelet packets,” in *Proc. ICASSP*, May 1995.

[81] M. W. Marcellin and T. R. Fischer, “Trellis coded quantization of memoryless and Gauss-Markov sources,” *IEEE Transactions on Communications*, vol. 38, pp. 82–93, January 1990.

[82] G. Ungerboeck, “Channel coding with multilevel/phase signals,” *IEEE Transactions on Information Theory*, vol. IT-28, pp. 55–67, January 1982.

[83] G. D. Forney, “The Viterbi algorithm,” *Proceedings of the IEEE*, vol. 61, pp. 268–278, March 1973.

[84] M. W. Marcellin, “On entropy-constrained trellis coded quantization,” *IEEE Transactions on Communications*, vol. 42, pp. 14–16, January 1994.

[85] P. A. Chou, T. Lookabaugh, and R. M. Gray, “Entropy-constrained vector quantization,” *IEEE Transactions on Information Theory*, vol. 37, pp. 31–42, January 1989.

[86] H. Jafarkhani, N. Farvardin, and C. C. Lee, “Adaptive image coding based on the discrete wavelet transform,” in *Proc. IEEE Int. Conf. Image Proc. (ICIP)*, vol. 3, (Austin, TX), pp. 343–347, November 1994.

[87] R. L. Joshi, T. Fischer, and R. H. Bamberger, "Optimum classification in subband coding of images," in *Proc. IEEE Int. Conf. Image Proc. (ICIP)*, vol. 2, (Austin, TX), pp. 883–887, November 1994.

[88] J. H. Kasner and M. W. Marcellin, "Adaptive wavelet coding of images," in *Proc. IEEE Int. Conf. Image Proc. (ICIP)*, vol. 3, (Austin, TX), pp. 358–362, November 1994.

[89] W. H. Chen and C. H. Smith, "Adaptive coding of monochrome and color images," *IEEE Transactions on Communications*, vol. COM-25, pp. 1285–1292, November 1977.

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

[91] C. Chrysafis and A. Ortega, "Efficient context based entropy coding for lossy wavelet image compression," in *Proc. Data Compression Conference*, (Snowbird, Utah), pp. 241–250, 1997.

[92] Y. Yoo, A. Ortega, and B. Yu, "Progressive classification and adaptive quantization of image subbands." Preprint, 1997.

[93] D. Marpe and H. L. Cycon, "Efficient pre-coding techniques for wavelet-based image compression." Submitted to PCS, Berlin, 1997.

[94] P. G. Sherwood and K. Zeger, "Progressive image coding on noisy channels," in *Proc. Data Compression Conference*, (Snowbird, UT), pp. 72–81, Mar. 1997.

[95] S. L. Regunathan, K. Rose, and S. Gadkari, "Multimode image coding for noisy channels," in *Proc. Data Compression Conference*, (Snowbird, UT), pp. 82–90, Mar. 1997.

[96] J. Garcia-Frias and J. D. Villasenor, "An analytical treatment of channel-induced distortion in run length coded image subbands," in *Proc. Data Compression Conference*, (Snowbird, UT), pp. 52–61, Mar. 1997.

[97] A. Nosratinia and M. T. Orchard, "A multi-resolution framework for backward motion compensation," in *Proc. SPIE Symposium on Electronic Imaging*, (San Jose, CA), February 1995.