A Wavelet-Based Analysis of Fractal Image Compression

Geoffrey M. Davis, *Member, IEEE*

Abstract— **Why does fractal image compression work? What is the implicit image model underlying fractal block coding? How can we characterize the types of images for which fractal block coders will work well? These are the central issues we address. We introduce a new wavelet-based framework for analyzing block-based fractal compression schemes. Within this framework we are able to draw upon insights from the well-established transform coder paradigm in order to address the issue of why fractal block coders work. We show that fractal block coders of the form introduced by Jacquin[1] are a Haar wavelet subtree quantization scheme. We examine a generalization of this scheme to smooth wavelets with additional vanishing moments. The performance of our generalized coder is comparable to the best results in the literature for a Jacquin-style coding scheme. Our wavelet framework gives new insight into the convergence properties of fractal block coders, and leads us to develop an unconditionally convergent scheme with a fast decoding algorithm. Our experiments with this new algorithm indicate that fractal coders derive much of their effectiveness from their ability to efficiently represent wavelet zerotrees. Finally, our framework reveals some of the fundamental limitations of current fractal compression schemes.**

Keywords— **fractal image compression, wavelets, self-quantization of subtrees, self-similarity, fractional Brownian motion**

I. Introduction

FRACTAL image compression techniques, introduced by Barnsley and Jacquin [2][3], are the product of the study of iterated function systems (IFS)[4]. These techniques involve an approach to compression quite different from standard transform coder-based methods. Transform coders model images in a very simple fashion, namely, as vectors drawn from a wide-sense stationary random process. They store images as quantized transform coefficients. Fractal block coders, as described by Jacquin, assume that "image redundancy can be efficiently exploited through *self-transformability* on a blockwise basis" [1]. They store images as contraction maps of which the images are approximate fixed points. Images are decoded by interating these maps to their fixed points.

The literature on fractal image compression has focused on three basic problems. The first problem is to determine a family of contraction maps that can be used to effectively code images [5][6]. Although a variety of families have been explored, most schemes in the literature are closely related to the block coders described by Jacquin in [1] and by Fisher in [7]. Throughout this paper, when we refer to fractal block coders, we will be referring to such Jacquin-style schemes. The second problem is to find fast and effective algorithms for associating a given image to a contraction map of which the image is an approximate fixed point[8] [9]. The third problem is to analyze the convergence properties of various families of maps and to establish error bounds for decoded images[10] [11].

In this paper we address the issues of finding effective families of maps and convergence properties of fractal schemes. More importantly, we address some much more fundamental questions. First and foremost, we seek to explain *why* fractal compression works. Toward this end, we ask, What is the implicit image model used in fractal image compression? How can we characterize the types of images for which fractal compression will work well? The theory of iterated function systems does not provide satisfactory answers.

We introduce a new wavelet-based framework for analyzing block-based fractal compression schemes. Within this framework we are able to draw upon insights from the well-established transform coder paradigm in order to address the issue of why fractal block coders work. Using the insights gained in our analysis, we obtain a generalization of fractal block coding that yields compression results that are comparable to the best reported in the literature for fractal block coders.

The main goal of the paper, however, is not to develop the best possible fractal block coder. Rather, we seek to understand the mechanisms underlying the performance of fractal block coders. Toward this end, we restrict our attention to wavelet-based coders that closely mimic the structure of the block coders introduced by Jacquin [1]. We note that comparable or slightly better results have been obtained with coders that relax these constraints and that use more elaborate quantization schemes [12] [13]. Indeed, the wavelet/zerotree coder which forms the foundation of our generalized fractal coder yields much better performance when our imposed constraints are relaxed.

We see that the fractal block coders of [1] [7] arise naturally in our wavelet-based framework as Haar quantization schemes, and we obtain a simple generalization of these schemes to smooth wavelet bases. We obtain new insight into the convergence properties of fractal block coders, and we describe an important unconditionally convergent variant of our generalized coder. Our experiments with this coder provide evidence that much of the performance of fractal block coders is due to the localization of image energy in both space and frequency. Finally, our framework reveals some of the fundamental limitations of current frac-

The author is an Assistant Professor of mathematics at Dartmouth College. E-mail: gdavis@cs.dartmouth.edu. Revised versions may be obtained from the web site *http://www.cs.dartmouth.edu/~gdavis*

tal compression schemes.

A. Related Work

The link between fractal image coding and wavelets is not a new one. The first mention of the connection was by Pentland and Horowitz in [14]. The algorithm described in [14], however, consists of a within-subband fixed vector quantizer that uses cross-scale conditioning for entropy coding vector indices, and is only loosely related to Jacquin-style schemes we examine here.

An important paper linking wavelets and fractal image coding is that of Rinaldo and Calvagno [13]. The coder in [13] uses blocks from low frequency image subbands as a vector codebook for quantizing blocks in higher frequency subbands. The main focus of [13] is to develop a new coder rather than to analyze the performance of fractal block coders in general. While the procedure in [13] is inspired by the Jacquin-style coders examined in this paper, it differs in important ways. We discuss these differences in Section V.

The link between fractal and wavelet-based coding described in Section III-B below was reported independently and nearly simultaneously by this author [15], by Krupnik, Malah, and Karnin [16], and by van de Walle [17]. This paper contains a substantial extension and generalization of the algorithms, analyses, and ideas presented in the previous three papers.

B. Outline

The balance of the paper is organized as follows. Section II gives an overview of a basic fractal block coding scheme. In Section III we introduce a wavelet-based framework for analyzing fractal block coding and show that Jacquin-style block coders are Haar subtree quantization schemes. In Section IV we introduce a simplified and generalized version of fractal block coding. Our analysis of the convergence properties of this scheme gives insight into the convergence properties of standard fractal block coders. Using our wavelet framework and a simple texture model we make Jacquin's assumption of "self-transformability" more concrete and we discuss *why* fractal block coding works for complex image features. In Section V we present experimental results and further discussion of the performance and limitations of fractal block coders.

II. Fractal Block Coders

In this section we describe a generic fractal block coding scheme based on those in [1][7], and we provide some heuristic motivation for the scheme. A more complete overview of fractal coding techniques can be found in [18][19].

A. Motivation for Fractal Coding

Transform coders are designed to take advantage of very simple structure in images, namely that values of pixels that are close together are correlated. Fractal compression is motivated by the observation that important image features, including straight edges and constant regions, are invariant under rescaling. Constant gradients are covariant under rescaling, i.e. rescaling changes the gradient by a constant factor. Scale invariance (and covariance) presents a type of structure for an image coder to exploit.

Fractal compression takes advantage of this local scale invariance by using coarse-scale image features to quantize fine-scale features. Fractal block coders perform a vector quantization (VQ) of image blocks. The vector codebook is constructed from locally averaged and subsampled isometries of larger blocks from the image. This codebook is effective for coding constant regions and straight edges due to the scale invariance of these features. The vector quantization is done in such a way that it determines a contraction map from the plane to itself of which the image to be coded is an approximate fixed point. Images are stored by saving the parameters of this map and are decoded by iterating the map to find its fixed point. An advantage of fractal block coding over VQ is that it does not require separate storage of a fixed vector codebook.

The ability of fractal block coders to represent straight edges, constant regions, and constant gradients efficiently is important, as transform coders fail to take advantage of these types of spatial structures. Indeed, recent wavelet transform based techniques that have achieved particularly good compression results have done so by augmenting scalar quantization of transform coefficients with a zerotree vector that is used to efficiently encode locally constant regions [20].

For fractal block coders to be effective, images must be composed of features at fine scales that are also present at coarser scales up to a rigid motion and an affine transform of intensities. This is the "self-transformability" assumption described by [1]. It is clear that this assumption holds for images composed of isolated straight lines and constant regions, since these features are self-similar. That it should hold when more complex features are present is much less obvious. In Section IV we use a simple texture model and our wavelet framework to provide a more detailed characterization of "self-transformable" images.

B. Mechanics of Fractal Block Coding

We now describe a simple fractal block coding scheme based on those in [1][7]. For convenience we will focus on systems based on dyadic block scalings, but we note that other scalings are possible. Let $\mathcal{I}$ be a $2^N \times 2^N$ pixel grayscale image. Let $\mathbf{B}^J_{K,L}$ be the linear "get-block" operator which when applied to $\mathcal{I}$ extracts the $2^J \times 2^J$ subblock with lower left corner at (K, L). The adjoint of this operator, $(\mathbf{B}^J_{K,L})^*$, is a "put-block" operator that inserts a $2^J \times 2^J$ image block into a $2^N \times 2^N$ all-zero image so that the lower left corner of the inserted block is at (K, L). We will use capital letters to denote block coordinates and lower case to denote individual pixel coordinates. We use a capital Greek multi-index, usually Γ, to abbreviate the block coordinates K, L and a lower-case Greek multi-index to abbreviate pixel coordinates within blocks.

We partition $\mathcal{I}$ into a set of non-overlapping $2^R \times 2^R$ *range blocks*. The goal of the compression scheme is to approximate each range block with a block from a codebook

TABLE I
BASIC NOTATION

$\mathbf{B}_\Gamma^R$	Get block of size $2^R \times 2^R$ from location Γ
$\mathbf{L}_P$	Apply isometry P to a block
$\mathbf{A}^k$	Average and subsample a block k times
$\mathbf{S}_\Gamma^{N-R}$	Get subtree with root from location Γ within scale (wavelet domain analog of $\mathbf{B}_\Gamma^R$)
$\widehat{\mathbf{A}}^k$	Truncate and rescale subtree k times (wavelet domain analog of $\mathbf{A}^k$)
$\widehat{\mathbf{L}}_P$	Rotate/reflect wavelet coefficients in a subtree (wavelet domain analog of $\mathbf{L}_P$)
$\mathbf{1}$	Square $2^N \times 2^N$ matrix of all 1's
$\mathcal{D}$	Domain pool
$\mathcal{R}$	Set of range blocks
Ω	Set of orientations HH, LH, HL
g_Γ	Gain coefficient for block/subtree Γ
h_Γ	DC offset/scaling function coefficient for block/subtree Γ

constructed from a set of $2^D \times 2^D$ *domain blocks*, where $0 < R < D \leq N$. Forming this approximation entails the construction of a contraction map from the image to itself, i.e. from the domain blocks to the range blocks, of which the image is an approximate fixed point. We store the image by storing the parameters of this map, and we recover the image by iterating the map to its fixed point. Iterated function system theory motivates this general approach to storing images, but gives little guidance on questions of implementation. The basic form of the block coder described below is the result of considerable empirical work. In Section III we see that this block-based coder arises naturally in a wavelet framework, and in Section IV we obtain greatly improved coder performance by generalizing these block-based maps to wavelet subtree-based maps.

The range block partition is a disjoint partition of the image consisting of the blocks $\{\mathbf{B}_{K,L}^R \mathcal{I} | (K,L) \in \mathcal{R}\}$. Here $\mathcal{R} = \{(2^R m, 2^R n) | 0 \leq m, n < 2^{N-R}\}$ where $m, n \in \mathbb{Z}$. The domain blocks from which the codebook is constructed are drawn from the *domain pool*, the set $\{\mathbf{B}_{K,L}^D \mathcal{I} | (K,L) \in \mathcal{D}\}$. A variety of domain pools are used in the literature. A commonly used pool [1] is the set of all unit translates of $2^D \times 2^D$ blocks, $\mathcal{D} = \{(m,n) | 0 \leq m, n < 2^N - 2^D\}$. Some alternative domain pools that we will discuss further are the *disjoint domain pool*, $\mathcal{D} = \{(2^D m, 2^D n) | 0 \leq m, n < 2^{N-D}\}$, a disjoint tiling of $\mathcal{I}$, and the *half-overlapping domain pool*, $\mathcal{D} = \{(2^{D-1} m, 2^{D-1} n) | 0 \leq m, n < 2^{N-D+1}\}$, the union of four disjoint partitions shifted by a half block length in the x or y directions (we periodize the image at its boundaries).

Two basic operators are used for codebook construction. The "average-and-subsample" operator $\mathbf{A}$ maps a $2^J \times 2^J$ image block to a $2^{J-1} \times 2^{J-1}$ block by averaging each pixel in $\mathbf{B}_\Gamma^J \mathcal{I}$ with its neighbors and then subsampling. We define $(\mathbf{A}\mathbf{B}_\Gamma^J \mathcal{I})(k,l) = \frac{1}{4}[(\mathbf{B}_\Gamma^J \mathcal{I})(2k,2l) + (\mathbf{B}_\Gamma^J \mathcal{I})(2k+1,2l) + (\mathbf{B}_\Gamma^J \mathcal{I})(2k,2l+1) + (\mathbf{B}_\Gamma^J \mathcal{I})(2k+1,2l+1)]$ where $\mathbf{B}_\Gamma^J \mathcal{I}(k,l)$ is the pixel at coordinates (k,l) within the subblock $\mathbf{B}_\Gamma^J \mathcal{I}$. A second operator is the symmetry operator $\mathbf{L}_k$, $1 \leq k \leq 8$, which maps a square block to one of the 8 isometries obtained from compositions of reflections and 90 degree rotations.

Range block approximation is similar to shape-gain vector quantization[21]. Range blocks are quantized to a linear combination of an element from the codebook and a constant block. The codebook used for quantizing range blocks consists of averaged and subsampled isometries of domain blocks, the set $\mathcal{C} = \{\mathbf{L}_k \mathbf{A}^{D-R} \mathbf{B}_\Gamma^D \mathcal{I} : \Gamma \in \mathcal{D}, 0 \leq k \leq 8\}$. Here $\mathbf{A}^{D-R}$ denotes the operator $\mathbf{A}$ applied $D-R$ times. The contrast of the codewords in $\mathcal{C}$ is adjusted by a gain factor g, and the DC component is adjusted by adding a subblock of the $2^N \times 2^N$ matrix of ones, $\mathbf{1}$, multiplied by an offset factor h. For each range block $\mathbf{B}_\Gamma^R \mathcal{I}$ we have

$$\mathbf{B}_\Gamma^R \mathcal{I} \approx g_\Gamma \mathbf{L}_{P(\Gamma)} \mathbf{A}^{D-R} \mathbf{B}_{\Pi(\Gamma)}^D \mathcal{I} + h_\Gamma \mathbf{B}_\Gamma^R \mathbf{1}. \tag{1}$$

Here $\Pi : \mathcal{R} \to \mathcal{D}$ assigns an element from the domain pool to each range element and $P : \mathcal{R} \to \{1 \ldots 8\}$ assigns each range element a symmetry operator index. Ideally the parameters g, h, Π, and P should be chosen so that they minimize the error in the decoded image. The quantization process is complicated by the fact that the codebook used by the decoder is different from that used by the encoder, since the decoder doesn't have access to the original domain blocks. Hence errors made in quantizing range blocks are compounded because they affect the decoder codebook. These additional effects of quantization errors have proven difficult to estimate, so in practice g, h, Π, and P are chosen to minimize the l^2 approximation error in 1. This tactic gives good results in practice; we discuss the propagation of errors further in [22].

C. Decoding Fractal Coded Images

The approximations for the range blocks (1) determine a constraint on the image $\mathcal{I}$ of the form $\mathcal{I} \approx \mathbf{G}\mathcal{I} + \mathcal{H}$. Expanding $\mathcal{I}$ as a sum of range blocks we obtain

$$\begin{aligned} \mathcal{I} &= \sum_{\Gamma \in \mathcal{R}} (\mathbf{B}_\Gamma^R)^* \mathbf{B}_\Gamma^R \mathcal{I} \\ &\approx \sum_{\Gamma \in \mathcal{R}} g_\Gamma (\mathbf{B}_\Gamma^R)^* \mathbf{L}_{P(\Gamma)} \mathbf{A}^{D-R} \mathbf{B}_{\Pi(\Gamma)}^D \mathcal{I} \\ &\quad + \sum_{\Gamma \in \mathcal{R}} h_\Gamma (\mathbf{B}_\Gamma^R)^* \mathbf{B}_\Gamma^R \mathbf{1} \\ &= \mathbf{G}\mathcal{I} + \mathcal{H}. \end{aligned}$$

Provided the matrix $\mathbf{I} - \mathbf{G}$ is nonsingular, there is a unique fixed point solution $\mathcal{I}_{fp}$ satisfying

$$\mathcal{I}_{fp} = \mathbf{G}\mathcal{I}_{fp} + \mathcal{H} \tag{2}$$

given by $\mathcal{I}_{fp} = (\mathbf{I} - \mathbf{G})^{-1} \mathcal{H}$. Because $\mathbf{G}$ is a $2^{2N} \times 2^{2N}$ matrix, inverting $\mathbf{I} - \mathbf{G}$ directly is an inordinately difficult task. If and only if the eigenvalues of $\mathbf{G}$ are all less than 1 in magnitude, we can find the fixed point solution $\mathcal{I}_{fp}$ by iteratively applying (2) to an arbitrary image $\mathcal{I}_0$. Decoding of fractal coded images proceeds by forming the sequence $\mathcal{I}_n = \mathbf{G}\mathcal{I}_{n-1} + \mathcal{H} = \mathbf{G}^n \mathcal{I}_0 + \sum_{k=0}^{n-1} \mathbf{G}^k \mathcal{H}$.

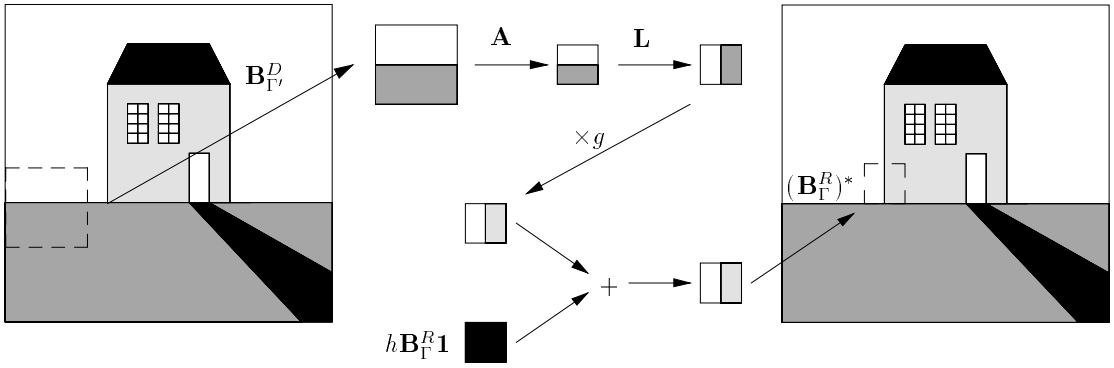

Fig. 1. We quantize the small range block $\mathbf{B}^R_\Gamma\mathcal{I}$ on the right using the codebook vector $g\mathbf{L}\mathbf{A}\mathbf{B}^D_{\Gamma'}\mathcal{I} + h\mathbf{B}^R_\Gamma\mathbf{1}$ obtained from the larger domain block on the left. $\mathbf{A}$ averages and subsamples the block, $\mathbf{L}$ rotates it, multiplication by the gain g modifies the contrast, and the addition of the offset $h\mathbf{B}^R_\Gamma\mathbf{1}$ adjusts the block DC component.

In general the image to be coded, $\mathcal{I}$, is not an exact fixed point of (2), i.e. $\mathcal{I} = \mathbf{G}\mathcal{I} + \mathcal{H} + \mathcal{E}$ where $\mathcal{E}$ is an error image. Only $\mathbf{G}$ and $\mathcal{H}$ are stored, so the difference between the decoded image $\mathcal{I}_{fp}$ and the original $\mathcal{I}$ is $\mathcal{I} - \mathcal{I}_{fp} = (\mathbf{I} - \mathbf{G})^{-1}\mathcal{E}$.

Our goal in coding is to minimize $\mathcal{I} - \mathcal{I}_{fp}$ with respect to some norm given a bit budget for storing $\mathbf{G}$ and $\mathcal{H}$. Given a vector norm $\|\bullet\|$ we construct a matrix norm by defining $\|\mathbf{G}\| = \max_{\|x\|=1} \|\mathbf{G}\mathbf{x}\|$. With respect to these norms we have

$$\|\mathcal{I} - \mathcal{I}_{fp}\| \leq \|(\mathbf{I} - \mathbf{G})^{-1}\|\|\mathcal{E}\| \leq \frac{\|\mathcal{E}\|}{1 - \|\mathbf{G}\|} \tag{3}$$

provided $\|\mathbf{G}\| < 1$, i.e. provided $\mathbf{G}$ is contractive with respect to the given norm. The bound (3) is referred to in fractal compression literature as the *collage theorem bound*.

The collage theorem bound is more useful as a motivator than as a practical numerical bound. Although we typically try to minimize the l^2 error between $\mathcal{I}$ and $\mathcal{I}_{fp}$, bounding $\|\mathbf{G}\|_2 < 1$ is quite difficult [23]. We can bound $\|\mathbf{G}\|_\infty < 1$ by constraining the gains $|g_\Gamma| < 1$ [1]. Although this ensures that the decoding process will converge, numerical experiments have found the collage bound to be too pessimistic by orders of magnitude [24]. Moreover, it fails to predict the convergence of block coding schemes when some gains $|g_\Gamma| > 1$. In section IV we discuss schemes with convergence properties that are independent of the g_Γ.

III. A Wavelet Framework

A. Notation

The wavelet transform is a natural tool for analyzing fractal block coders since wavelet bases possess the same type of dyadic self-similarity that fractal coders seek to exploit. In particular, the Haar wavelet basis possesses a regular block structure that is aligned with the range block partition of the image. We show below that the maps generated by fractal block coders reduce to a simple set of equations in the wavelet transform domain.

Separable 2-D biorthogonal wavelet bases consist of translates and dyadic scalings of a set of separable wavelets $\psi_{LH}(x,y)$, $\psi_{HL}(x,y)$, and $\psi_{HH}(x,y)$ together with translates of a scaling function $\phi(x,y)$. We will use the subscript ω to represent one of the three orientations in $\Omega = \{LH, HL, HH\}$. We will limit our attention to symmetrical (or antisymmetrical) bases. The discrete wavelet transform of a $2^N \times 2^N$ image $\mathcal{I}$ expands the image into a linear combination of the basis functions in the set $\mathcal{W}_J = \{\phi^J_{k,l} | 0 \leq k,l < 2^J\} \cup \{\psi^j_{\omega,k,l} | \omega \in \Omega; J \leq j < N; 0 \leq k,l < 2^j\}$. We will use a single lower-case Greek multi-index, usually γ, to abbreviate the orientation and translation subscripts of ϕ and ψ. The coefficients for the basis functions $\phi^j_{k,l}$ and $\psi^j_{\omega,k,l}$ are given by $\langle\tilde{\phi}^j_{k,l}, \mathcal{I}\rangle$ and $\langle\tilde{\psi}^j_{\omega,k,l}, \mathcal{I}\rangle$, respectively, where $\tilde{\phi}^j_{k,l}$ and $\tilde{\psi}^j_{\omega,k,l}$ are dual scaling functions and wavelets.

An important property of wavelet basis expansions, particularly Haar expansions, is that they preserve the spatial localization of image features. For example, the coefficient of the Haar scaling function $\phi^J_{k,l}$ is proportional to the average value of an image in the $2^J \times 2^J$ block of pixels with lower left corner at $2^J k, 2^J l$. The wavelet coefficients associated with this region are organized into three quadtrees. We call this union of three quadtrees a *wavelet subtree*. Coefficients forming such a subtree are shaded in each of the transforms in Figure 2. At the root of a wavelet subtree are the coefficients of the wavelets $\psi^J_{\omega,k,l}$, where $\omega \in \Omega$. These coefficients correspond to the block's coarse-scale information. Each wavelet coefficient $\langle\tilde{\psi}^j_{\omega,k,l}, \mathcal{I}\rangle$ in the tree has four children that correspond to the same spatial location and the same orientation. The children consist of the coefficients of the wavelets of the next finer scale, $\psi^{j+1}_{\omega,2k,2l}$, $\psi^{j+1}_{\omega,2k+1,2l}$, $\psi^{j+1}_{\omega,2k,2l+1}$, and $\psi^{j+1}_{\omega,2k+1,2l+1}$. A wavelet subtree consists of the coefficients of the roots, together with all of their descendents in all three orientations. The scaling function $\phi^J_{k,l}$ is localized in the same region as the subtree with roots given by $\psi^J_{\omega,k,l}$, and we refer to this $\phi^J_{k,l}$ as the scaling function associated with the subtree.

B. A Wavelet Analog of Fractal Block Coding

We now describe a wavelet-based analog of fractal block coding introduced in [15]. Fractal block coders approximate a set of $2^R \times 2^R$ range blocks using a set of $2^D \times 2^D$ domain blocks. The wavelet analog of an image block, a set of pixels associated with a small region in space, is a wavelet subtree together with its associated scaling func-

tion coefficient. We define a linear "get-subtree" operator $\mathbf{S}^J_{K,L} : \mathbb{R}^{2^{2N}} \to \mathbb{R}^{2^{2(N-J)}-1}$ which extracts from an image the subtree whose root level consists of the coefficients of $\psi^J_{\omega,K,L}$ for all ω. We emphasize that when we discuss wavelet subtrees in this paper, we will primarily be discussing trees of coefficients of *all 3 orientations* as opposed to more commonly used subtrees of a fixed orientation.

The adjoint of $\mathbf{S}^J_{K,L}$ is a "put-subtree" operator which inserts a given subtree into an all-zero image so that the root of the inserted subtree corresponds to the coefficients $\psi^J_{\omega,K,L}$ for $\omega \in \Omega$. For the Haar basis, subblocks and their corresponding subtrees and associated scaling function coefficients contain identical information, i.e. the transform of a range block $\mathbf{B}^R_\Gamma \mathcal{I}$ yields the coefficients of subtree $\mathbf{S}^{N-R}_\Gamma \mathcal{I}$ and the scaling function coefficient $\langle \tilde{\phi}^{N-R}_\Gamma, \mathcal{I} \rangle$. For the remainder of this section we will take our wavelet basis to be the Haar basis. The actions of the get-subtree and put-subtree operators are illustrated in Figure 2.

The linear operators used in fractal block coding have simple behavior in the transform domain. We first consider the wavelet analog $\widehat{\mathbf{A}}$ of the average-and-subsample operator $\mathbf{A}$. Averaging and subsampling the finest-scale Haar wavelets sets them to 0. The local averaging has no effect on coarser scale Haar wavelets, and subsampling ψ^j_γ yields the Haar wavelet at the next finer scale, ψ^{j+1}_γ, multiplied by $\frac{1}{2}$. Similarly, averaging and subsampling the scaling function ϕ^j_γ yields $\frac{1}{2}\phi^{j+1}_\gamma$ for $j < N-1$ and 0 for $j = N-1$. The action of the averaging and subsampling operator thus consists of a shifting of coefficients from coarse-scale to fine, a multiplication by $\frac{1}{2}$, and a truncation of the finest-scale coefficients. The operator $\widehat{\mathbf{A}}$ prunes the leaves of a subtree and shifts all remaining coefficients to the next finer scale. The action of $\widehat{\mathbf{A}}$ is illustrated in Figure 2.

For symmetrical wavelets, horizontal/vertical block reflections correspond to a horizontal/vertical reflection of the set of wavelet coefficients within each scale of a subtree. Similarly, 90 degree block rotations correspond to 90 degree rotations of the set of wavelet coefficients within each scale and a switching of the ψ_{LH} coefficients with ψ_{HL} coefficients. Hence the wavelet analogs $\widehat{\mathbf{L}}_k$ of the block symmetry operators $\mathbf{L}_k$ permute wavelet coefficients within each scale. Figure 2 illustrates the action of a symmetry operator on a subtree. Note that the Haar basis is the only orthogonal basis we consider here, since it is the only compactly supported symmetrical wavelet basis[25]. When we generalize to non-Haar bases, we must use biorthogonal bases to obtain both symmetry and compact support.

The approximation (1) leads to a similar relation for subtrees in the Haar wavelet transform domain,

$$\mathbf{S}^{N-R}_\Gamma \mathcal{I} \approx g_\Gamma \widehat{\mathbf{L}}_{P(\Gamma)} \widehat{\mathbf{A}}^{D-R} \mathbf{S}^{N-D}_{\Pi(\Gamma)} \mathcal{I}. \tag{4}$$

We refer to this quantization of subtrees using other subtrees as the *self-quantization* of $\mathbf{S}^{N-R}_\Gamma \mathcal{I}$. The offset terms h_Γ from (1) affect only the scaling function coefficients because the left hand side of (4) is orthogonal to the subblocks of $\mathbf{1}$. Breaking up the subtrees into their constituent wavelet coefficients, we obtain a system of equations for the coefficients of the ψ^j_γ in $\mathbf{S}^{N-R}_\Gamma \mathcal{I}$,

$$\langle \tilde{\psi}^j_\gamma, \mathcal{I} \rangle \approx \frac{g_\Gamma}{2^{D-R}} \langle \tilde{\psi}^{j-(D-R)}_{\gamma'}, \mathcal{I} \rangle = \frac{g_\Gamma}{2^{D-R}} \langle \mathbf{T} \tilde{\psi}^j_\gamma, \mathcal{I} \rangle. \tag{5}$$

Here $\mathbf{T}$ is the map induced by the domain block selection followed by averaging, subsampling, and rotating. We obtain a similar relation for the scaling function coefficients,

$$\begin{aligned} \langle \tilde{\phi}^{N-R}_\Gamma, \mathcal{I} \rangle &\approx \frac{g_\Gamma}{2^{D-R}} \langle \tilde{\phi}^{N-D}_{\Pi(\Gamma)}, \mathcal{I} \rangle + h_\Gamma \\ &= \frac{g_\Gamma}{2^{D-R}} \langle \mathbf{T} \tilde{\phi}^{N-R}_\Gamma, \mathcal{I} \rangle + h_\Gamma \end{aligned} \tag{6}$$

From the system (5) and (6) we see that, roughly speaking, the fractal block quantization process constructs a map from coarse-scale wavelet coefficients to fine. It is important to note that the operator $\mathbf{T}$ in (5) and (6) does *not* necessarily map elements of $\mathcal{W}_{N-D}$ to elements of $\mathcal{W}_{N-D}$, since translation of domain blocks by distances smaller than 2^D leads to non-integral translates of the wavelets in their corresponding subtrees. We discuss this notion of a map from coarse to fine scales in greater detail in Section IV-E.

IV. Self-Quantization of Subtrees

A. Generalization to non-Haar bases

We obtain a wavelet-based analog of fractal compression by replacing the Haar basis used in (5) and (6) with a symmetric biorthogonal wavelet basis. This change of basis brings a number of benefits. Smooth wavelet bases eliminate the sharp discontinuities at range block boundaries caused by quantization errors. These artifacts are especially objectionable because the eye is particularly sensitive to horizontal and vertical lines. Moreover, bases with a higher number of vanishing moments than the Haar better approximate the K-L basis for the fractional Brownian motion texture model described below, and they therefore improve coder performance in these textured regions. Figure 5 compares images coded with Haar and smooth spline bases. We see both an increase in overall compressed image fidelity with the spline basis as well as a dramatic reduction in block boundary artifacts.

B. Self-Quantization of Subtrees

We now introduce a simplification of the coding scheme that facilitates our analysis of the convergence properties of these generalized fractal block coders. We store an image $\mathcal{I}$ by storing the parameters in the relations (5) and (6). We must store one constant h_Γ for each scaling function. Image decoding is greatly simplified if we store the scaling function coefficients directly rather than storing the h_Γ's. We then only have to recover the wavelet coefficients when decoding. Also, because we know how quantization errors for the scaling function coefficients will affect the decoded image, we have greater control over the final decoded error than we do with the h_Γ's. We call this modified scheme in which we use (4) to quantize wavelet coefficients and we store scaling function coefficients directly the *self-quantization of subtrees* (SQS) scheme.

We can encode the scaling function coefficients more efficiently in our SQS scheme than we can the h_Γ's in the

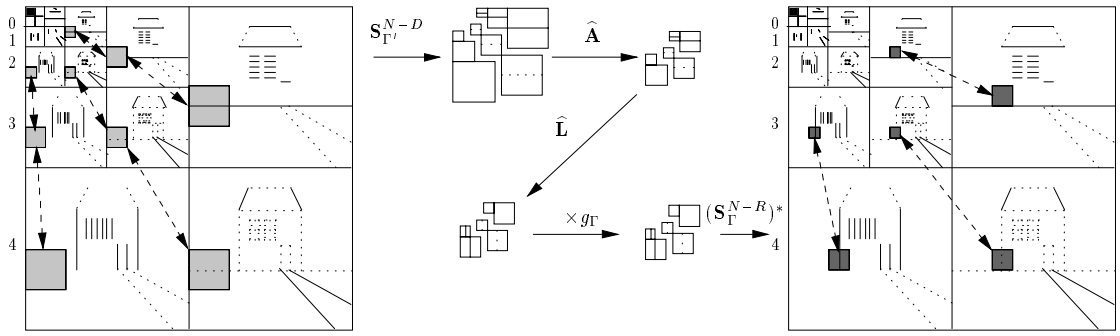

Fig. 2. We approximate the darkly shaded range subtree $\mathbf{S}_\Gamma^{N-R}\mathcal{I}$ using the codebook element $g_\Gamma\widehat{\mathbf{L}}\widehat{\mathbf{A}}\mathbf{S}_{\Gamma'}^{N-D}\mathcal{I}$ derived from the lightly shaded domain subtree $\mathbf{S}_{\Gamma'}^{N-D}\mathcal{I}$. $\widehat{\mathbf{A}}$ truncates the finest scale coefficients of the domain subtree, and $\widehat{\mathbf{L}}$ rotates it. We store coefficients in subbands of scale 2 and lower and the encodings of all subtrees with roots in scale subband 3. Note that in our usage a subtree contains coefficients from all three orientations, HL, LH, and HH.

original scheme. The scaling function coefficients associated with the range blocks are correlated and contain considerable redundancy. We can exploit this redundancy by computing the wavelet transform of these coefficients and storing these coarse-scale wavelet coefficients. Although we could employ a similar strategy with the h_Γ's, it is less likely to yield significant coding gains. The reason is that the h_Γ's are formed from linear combinations of pairs of scaling function coefficients from different parts of the image. These scaling function coefficient pairs are not in general correlated because the l^2 codeword selection criterion that determines the pairings does not take the values of these coefficients into account. Combining these dissimilar coefficients destroys any spatial structure that may be present in the h_Γ's and makes them difficult to code.

C. Convergence for non-Haar Bases

The first issue we must address is that of convergence, since our generalized encoding scheme is pointless if we cannot decode our images. Our wavelet analog of (1) gives rise to a relation for $\mathcal{I}$ similar to (2),

$$(\mathbf{W}\mathcal{I}) \approx \mathbf{G}_\mathbf{W}(\mathbf{W}\mathcal{I}) + \mathcal{H}_W\,. \tag{7}$$

Here $\mathbf{W}\mathcal{I}$ is the discrete wavelet transform of $\mathcal{I}$. We first examine the convergence properties of the unmodified wavelet analog of fractal block coding in (1). We assume a simple non-adaptive scheme.

Equations (5) and (6) express the coefficient of each wavelet and scaling function $w_k \in \mathcal{W}_{N-R}$ in terms of the coefficient of a translated wavelet or scaling function at a coarser scale, $\mathbf{T}w_j$. This translated function $\mathbf{T}w_j$ will not in general be an element of the basis $\mathcal{W}_{N-R}$. Expanding the inner product $\mathbf{T}w_j$ over the basis $\mathcal{W}_{N-R}$, we obtain $\mathbf{T}w_j = \sum_{w_k\in\mathcal{W}_{N-R}}\langle\tilde{w}_k, \mathbf{T}w_j\rangle w_k$.

The entries of the matrix $\mathbf{G}_\mathbf{W}$ in the row corresponding to the basis function $w \in \mathcal{W}_{N-R}$ will thus be given by $\mathbf{G}_{j,k} = 2^{R-D}g_{\Gamma_j}\langle\tilde{w}_k, \mathbf{T}w_j\rangle$ for $w_j, w_k \in \mathcal{W}_R$. The iterative scheme will converge if the magnitudes of the eigenvalues of $\mathbf{G}_\mathbf{W}$ are all less than one. A sufficient condition for convergence is that $\|\mathbf{G}_\mathbf{W}\|_\infty < 1$, and this condition will be satisfied provided that for all Γ we have $|g_\Gamma| < 2^{D-R}\min_{w'}\left(\sum_{w\in\mathcal{W}_{N-R}}|\langle\tilde{w}, w'\rangle|\right)^{-1}$. Here the minimum is taken over all wavelets contained in the subtrees in the domain pool and over all scaling functions associated with these subtrees. This result can be extended to adaptive coders, but it becomes decomposition dependent.

We can obtain a similar sufficient condition for convergence for our scheme in which we store scaling function coefficients directly, and this condition is more readily extended to adaptive coders. We first consider the structure of the matrix $\mathbf{G}_\mathbf{W}$ in (7) for the modified scheme. We first order the coefficients of the image vectors $\mathbf{W}\mathcal{I}$ and $\mathcal{H}_W$ so that they are grouped into scaling function coefficients, $(\mathbf{W}\mathcal{I})_\phi$ and $\mathcal{H}_\phi$, and wavelet coefficients, $(\mathbf{W}\mathcal{I})_\psi$ and $\mathcal{H}_\psi$. The matrix relation (7) becomes, in block form,

$$\begin{pmatrix}(\mathbf{W}\mathcal{I})_\phi\\(\mathbf{W}\mathcal{I})_\psi\end{pmatrix} = \begin{pmatrix}\mathbf{G}_{\phi\phi} & \mathbf{G}_{\phi\psi}\\ \mathbf{G}_{\psi\phi} & \mathbf{G}_{\psi\psi}\end{pmatrix}\begin{pmatrix}(\mathbf{W}\mathcal{I})_\phi\\(\mathbf{W}\mathcal{I})_\psi\end{pmatrix} + \begin{pmatrix}\mathcal{H}_\phi\\\mathcal{H}_\psi\end{pmatrix}.$$

All coefficients in $\mathcal{H}_\psi$ are zero in both the unmodified and modified schemes, since the information in $\mathcal{H}_W$ depends only on the scaling function coefficients of the image. The SQS modification removes the implicit dependence of the scaling function coefficients $(\mathbf{W}\mathcal{I})_\phi$ on other coefficients in the image, so $\mathbf{G}_{\phi\phi}$ and $\mathbf{G}_{\phi\psi}$ have all zero entries. We thus have $\|\mathbf{G}_W\|_\infty < 1$ provided $\|\mathbf{G}_{\psi\psi}\|_\infty < 1$, and we can ensure that this condition will be satisfied by restricting the gains g_Γ. Upper bounds for these g_Γ's can be obtained numerically for various domain pools. Table II below lists upper bounds for g_Γ that will ensure that $\|\mathbf{G}_{\psi\psi}\|_\infty < 1$ for the 7/9 tap spline variant basis of [26]. We assume a 512×512 image with a domain pool consisting of all unit translates of blocks of size $2^D\times2^D$ or smaller. We further assume that range blocks of size $2^R\times2^R$ are quantized using domain blocks of size $2^{R+1}\times2^{R+1}$. These bounds apply to the adaptive schemes described in section V provided the maximum domain block size satisfies the limits in the table.

We have thus shown that provided a suitable bound is imposed on the gains g_Γ, we can extend fractal block coding techniques to non-Haar wavelet subtrees with *arbitrary* domain pools. We emphasize that this bound is a sufficient

TABLE II
BOUND ON $|g_\Gamma|$ TO ENSURE $\|G_W\|_\infty < 1$

Largest domain block	Bound for $\|g_\Gamma\|$	
	Haar basis	Spline basis
64×64	0.140	0.193
32×32	0.174	0.201
16×16	0.222	0.231
8×8	0.333	0.256
4×4	0.500	0.431

but not a necessary condition for convergence. Indeed, in the next section we describe some special cases of domain pools for which decoding is unconditionally convergent.

D. Unconditional Convergence

Because we store the scaling function coefficients as coarse-scale wavelet coefficients, the matrix $\mathbf{G_W}$ becomes a map from $\mathcal{W}_0$ to $\mathcal{W}_0$. The wavelets and scaling functions $\mathbf{T}w$ that make up the translated domain blocks do not in general belong to the basis $\mathcal{W}_0$, and this leads to a complicated matrix $\mathbf{G_W}$. If, however, we restrict the domain pool to the disjoint domain pool, $\mathcal{D} = \{(2^D m, 2^D n) | 0 \leq m, n < 2^{N-D}\}$, the matrix $\mathbf{G_W}$ simplifies considerably. In this case the coefficients in the domain subtrees all correspond to wavelets in the basis $\mathcal{W}_0$. From (5) we see that each wavelet coefficient of scale $N - R$ and finer depends on exactly one coarser-scale wavelet coefficient. The matrix $\mathbf{G_W}$ is thus a map from coarse scales to fine.

The rows of the matrix $\mathbf{G_W}$ for this domain pool contain a single nonzero entry with value $\frac{g_\Gamma}{2^{D-R}}$. We order the vector of coefficients from coarse to fine, so $\mathbf{G_W}$ will be a strictly lower triangular matrix with all zeros on the diagonal. Hence, all eigenvalues of $\mathbf{G_W}$ are zero, and the reconstruction procedure for this domain pool converges unconditionally in a finite number of steps. This particular $\mathbf{G_W}$ is a special case of an *R-scale-extending* map, a map for which each wavelet coefficient of scale $j \geq N - R$ in the range is dependent only on the coefficients of wavelets in the domain from the same basis of scales coarser than j.

Theorem 1 (Reconstruction Theorem) Let $\mathcal{I}$ be a $2^N \times 2^N$ image for which the scaling function coefficients $\langle \tilde{\phi}_\gamma^{N-R}, \mathcal{I} \rangle$ are known. If $\mathcal{I}$ is the fixed point of a linear R-scale-extending map $\mathbf{M}$, then we can recover $\mathcal{I}$ from these scaling function coefficients using R applications of the map $\mathbf{M}$.

Proof: By applying the wavelet transform to the image $\mathcal{I}_R = \sum_\gamma \langle \tilde{\phi}_\gamma^{N-R}, \mathcal{I} \rangle \phi_\gamma^{N-R}$, we obtain all the coarse-scale wavelet coefficients $\langle \tilde{\psi}_\gamma^j, \mathcal{I} \rangle$ for $j < N - R$. We can now obtain the wavelet coefficients $\langle \tilde{\psi}_\gamma^{N-R}, \mathcal{I} \rangle$ by applying the map $\mathbf{M}$, since these coefficients depend only on the coefficients we already know. Each time we apply the map $\mathbf{M}$ we obtain the wavelet coefficients at the next finer scale, so by induction the result is proved. ■

The intuition behind this proof can be seen in Figure 3. The shaded coefficients represent coefficients that are stored by the SQS coder. Each range subtree is quantized to a domain subtree with root at a coarser scale. When we apply the map $\mathbf{G_W}$ to the image, information is carried from the stored shaded section to the unshaded section. Each application of the map $\mathbf{G_W}$ transfers known coarse-scale information to the next finer scale, so we recover the image coefficients one scale at a time.

The disjoint domain pool illustrated for the 1-D case in Figure 3 is a particularly simple scale-extending map. We also obtain an scale-extending map when we use an orthogonal basis with the half-overlapping domain pool, $\mathcal{D} = \{(2^{D-1} m, 2^{D-1} n) | 0 \leq m, n < 2^{N-D+1}\}$. For this domain pool, all domain subtree coefficients correspond to wavelets in $\mathcal{W}$ except for the root coefficients, which correspond to half-integer translates of $\psi_{\omega,\Gamma}^{N-D}$. The map $\mathbf{G_W}$ will be scale-extending provided we can show that these half-integer translates $\psi(x - \frac{k}{2})$ are orthogonal to the the finer scale wavelets $2^{j/2}\psi(2^j x - n)$ for $j > 0$. This can be seen by noting that $\langle \psi(x - \frac{k}{2}), \psi(2^j x - n) \rangle = \langle \psi(x), \psi(2^j x - n + 2^{j-1} k) \rangle = 0$. Hence the map $\mathbf{G_W}$ will still be scale-extending.

For the Haar basis this half-overlapping domain pool corresponds to the set of domain blocks which share boundaries with range blocks of the next finer scale. This particular restricted domain pool has been studied for standard fractal block coders in [27] and [28]. The above theorem generalizes the results of [27] and [28] and shows clearly why these results hold.

Our convergence proof yields a fast algorithm for decoding SQS-coded images. For the disjoint domain pool each fine-scale wavelet coefficient depends on only one other coefficient. The cascading of information from coarse scales to fine thus requires only $O(1)$ operations per pixel. We also obtain a fast decoding scheme for orthogonal bases with the half-overlapping domain pool since in this case the matrix $\mathbf{G_W}$ has a sparse block structure.

The above reconstruction theorem generalizes to allow adaptive image encoding. Using the disjoint domain pool, we can recover an image using a fast algorithm provided that for each self-quantized subtree we store its associated scaling function coefficient. Equivalently, we can recover an image provided we know all coarse-scale wavelet coefficients not contained in the range subtrees.

E. Discussion

Standard fractal compression schemes entail the quantization of "fine-scale" features using "coarse-scale" features. The above theorem shows we can make this notion of scale rigorous when using the disjoint domain pool. The scale of a particular image feature is determined by the detail space it occupies in a multiresolution analysis of the image. It is because the detail space of resolution 2^J is not invariant under translations smaller than 2^{N-J} pixels that we have convergence problems when we expand the domain pool to include fine translates of domain subtrees. When we approximate range subtrees using fine translates of domain subtrees, we introduce dependencies of fine-scale wavelet coefficients on coefficients from the same or finer scales. Information no longer flows strictly from coarse to fine un-

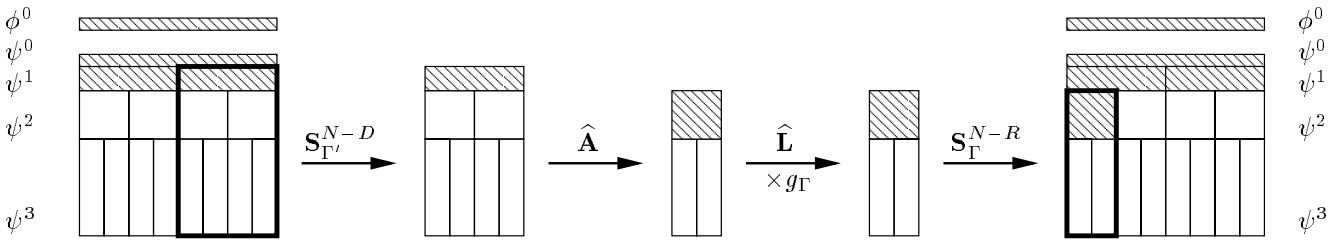

Fig. 3. The above figure illustrates how information is carried from coarse scales to fine via an application of a scale-extending map in 1-D. The shaded coefficients are stored by the self-quantization of subtrees coder, and below this level all range subtrees are quantized using domain subtrees. The range subtree on the right has been quantized using a codeword constructed from the larger domain subtree on the left. The operator $\widehat{\mathbf{A}}$ shifts the stored coarse information in the domain subtree to the next finer scale. Each range subtree is quantized in a similar fashion, so each application of the scale-extending map recovers one scale of information.

der the map $\mathbf{G_W}$. Dependency loops from fine-scales to fine-scales permit the growth of unstable eigenvectors unless these loops are damped by restricting the magnitudes of the gains g_Γ.

While unconditionally convergent domain pools allow efficient decoding algorithms, they are by means necessary for image coding. We describe experiments with more densely sampled domain pools in Section V. Although decoding instabilities potentially exist when scaling factors are allowed to exceed the values in Table II, we do not observe any such instabilities in our experiments.

F. Fractal Block Coding of Textures

In section II-A we motivated the codebook used by fractal block coders by emphasizing the scale-invariance of isolated straight edges, constant regions, and constant gradients. More complex image structures lack this deterministic self-similarity, however. How can we explain fractal block coders' ability to compress images containing complex structures? Why should the codebook constructed from the domain blocks be an effective one for regions that are not self-similar? We address these questions by examining fractal coding of textures. We model textures as fractional Brownian motion (fBm) processes as proposed by Pentland [29].

The fractional Brownian motion texture model captures an essential feature of natural images, the fact that their power spectra decay according to a power law. Fractional Brownian motion processes have spectral decay rates ranging from f^{-1} to f^{-3}, where f is frequency. Here f^{-2} corresponds to ordinary Brownian motion. Measurements of spectral decay in natural images show decay rates between f^{-2} and f^{-3}. Field[30] hypothesizes that image contrast is invariant across scale, which implies that image luminance power spectra decay like f^{-2}. His measurements of the spectra of natural images show an overall decay rate of roughly $f^{-2.2}$.

Although Fractional Brownian motion processes are not deterministically self-similar, they are *statistically* self-similar, i.e. the statistics of scaled subsets are identical to the statistics of the original set. Flandrin [31] has shown that the wavelet transform coefficients of a fractional Brownian motion process are stationary sequences with a self-similar covariance structure. This means that the codebook constructed from domain subtrees will possess the same second order statistics as the set of range subtrees. Hence for fBm textured regions, the quantization in (4) involves matching two random vectors drawn from sources with the same second order statistics.

Obtaining a close match between pairs of high dimensional random vectors is an extremely difficult task unless the distribution of these vectors is such that the vectors are highly clustered. Fractal coders can avoid this difficult high-dimensional problem to some extent by adaptively using small range blocks when necessary. Adaptation alone does not explain the performance of fractal block coders in complex regions, however. In numerical experiments we find that although the quantized range blocks/subtrees tend to be smaller in textured regions, they are still considerably larger than the trivial case.

Why should such clustering occur in natural images? The answer lies in the fact that the Haar transform acts as an approximate Karhunen-Loève (K-L) transform for ordinary Brownian motion, concentrating the energy in the coarse-scale coefficients. The result is that for Brownian motion processes, the Haar subtrees are clustered around the low-dimensional subspace consisting of subtrees with all-zero fine-scale coefficients. Moreover, because of the statistical self-similarity of Brownian motion, the second order statistics of these clusters are the same (up to a constant factor) for range and domain subtrees. Matching random subtrees that lie near this low-dimensional subspace is a much easier problem than matching arbitrary random subtrees. Statistical self-similarity alone is not enough to enable fractal coders to perform effectively, however. The clustering effects of the Haar transform for a statistically self-similar process with an *increasing* power spectrum are negligible. Thus, our texture model suggests that fractal block coders owe much of their performance in complex regions to the decaying power spectra of these regions.

The Haar transform is a less effective approximate K-L transform for fBm processes with rates of spectral decay corresponding more closely to observed values. When the decay is $O(f^{-\beta})$ for $2 < \beta < 3$, the autocorrelation function for a coefficient lag of n decays as $|n|^{\beta-3}$ for n large [31]. Tewfik and Kim[32] have shown that for such fBm's, transforms using bases with larger numbers of vanishing moments yield much better approximations to the K-L trans-

form. Our texture model therefore motivates the use of bases with additional vanishing moments. Indeed, numerical experiments described in Section V show more effective subtree quantization when using wavelets with higher numbers of vanishing moments.

An important observation is that quantization in our texture model entails matching pairs of random vectors. The process of matching random subtrees is comparable to quantizing a random vector $\mathbf{x}$ with a density function $p(\mathbf{x})$ using a quantizer with bins distributed according to the same density $p(\mathbf{x})$. For high resolution entropy constrained quantization, the optimal distribution of quantizer bins is very nearly uniform [33] [34]. As we have seen from our texture model, the distribution of code vectors used by fractal coders is far from the near-optimal uniform distribution. Codewords for our fractal block scheme will be unnecessarily densely distributed in high probability regions and too sparsely distributed in low probability regions. This conjecture is borne out in numerical experiments described in Section V. We find a tight clustering of codewords around the all-zero subtree, which leads to an inefficient codebook.

V. Results

A. Implementation

Our self-quantization of subtrees scheme possesses a structure similar to that of the space-frequency coder described in [35]. We have two basic methods for quantizing data: we have a set of coarse-scale wavelet coefficients that we quantize using a set of scalar quantizers, and we have a set of range subtrees that we self-quantize using codewords generated from domain subtrees. Given a partition of our data into range subtrees and coarse-scale coefficients, the determination of a near-optimal quantization of each set of data is a straightforward problem. The problem is finding the most effective partition of the data. We employ an algorithm that optimizes the allocation of bits between a set of scalar quantizers and a set of subtree quantizers following [35].

The source code for our implementation and scripts for generating the figures are available from the web site http://www.cs.dartmouth.edu/~gdavis/fractal/fractal.html.

The implementation is based on the public domain Wavelet Image Compression Construction Kit, available from http://www.cs.dartmouth.edu/~gdavis/wavelet/wavelet.html.

B. SQS vs. Fractal Block Coders

Figure 4 compares the peak signal to noise ratios of the 512×512 Lena image compressed by two fractal block coders, by our self-quantization of subtrees (SQS) scheme, and by a wavelet transform coder. Images compressed at roughly 64:1 by the various methods are shown in Figure 5 to illustrate the artifacts they generate.

The bottommost line in Figure 4, 'Fractal Quadtree', was produced by the quadtree block coder listed in the appendix of [18]. The command line used to generate the data was "enc -t XX -m 3 -M 7 -w 512 -d 1 -D 0 -f lena.raw lena.tXX", where XX ranged from 1 to 20. These parameters dictate that the encoding uses the disjoint domain pool to encode range blocks from size 4×4 to 64×64 and that gains are quantized uniformly between -1 and 1 using 5 bits per gain factor. We used a small domain pool here for comparison with the SQS schemes, which use an equivalent pool. As we discuss below, the performance of this quadtree scheme improves when larger domain pools are used. Allowing gains of magnitude 1.2 to 1.4 also yields marginally better encodings of the Lena image, but convergence is no longer guaranteed. We have restricted the gains to levels required to ensure image independent convergence. The use of this coder is primarily to provide a point of reference, since this coder is well-known in the area of fractal coding.

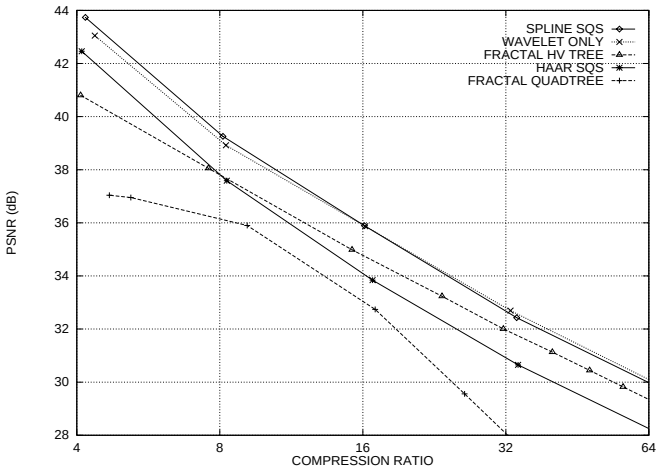

Fig. 4. PSNR's as a function of compression ratio for the 512×512 Lena image using fractal block coding, our self-quantization of subtrees (SQS) scheme, and a baseline wavelet transform coder.

The next line, 'Haar SQS', was generated by our adaptive SQS scheme using the Haar basis. We use the disjoint domain pool for coding range subtrees corresponding to blocks with sizes from 4×4 to 64×64. As we see from Figure 5, the SQS scheme produces dramatically improved results compared to the quadtree scheme, although both schemes use exactly the same domain pool. A large part of this improvement is attributable to the fact that the quadtree coder uses no entropy coding, whereas the SQS coder uses an adaptive arithmetic coder. However, a significant fraction of the bitstream consists of domain block offset indices, for which arithmetic coding is of little help. Much of the gain for SQS is because our improved understanding of how various bits contribute to final image fidelity enables us to partition bits more efficiently between wavelet coefficients and subtrees. Further gains come from storing the coarse-scale image information as quantized wavelet coefficients rather than as a set of h_Γ's. Finally, some of the improvement is also attributable to SQS's ability to use a greater range of gain factors due to its unconditional convergence.

Fisher notes that the performance of quadtree coders is significantly improved by enlarging the domain pool [7]. The third line from the bottom of Figure 4, 'Fractal HV Tree', was produced by a fractal block encoding of rect-

Fig. 5. The leftmost 512×512 Lena image has been compressed at 60.6:1 (PSNR = 24.9 dB) using a disjoint domain pool and the quadtree coder from [18]. The center image has been compressed at 68.2:1 (PSNR = 28.0 dB) using our self-quantization of subtrees (SQS) scheme with the Haar basis. Our SQS scheme uses exactly the same domain pool as the quadtree scheme, but our analysis of the SQS scheme enables us to make much more efficient use of bits. The rightmost image has been compressed at 65.6:1 (PSNR = 29.9 dB) using a smooth wavelet basis. Blocking artifacts have been completely eliminated.

angular range blocks using rectangular domain blocks [6]. The use of rectangular blocks introduces an additional degree of freedom in the construction of the domain pool and gives increased flexibility to the partitioning of the image. This strategy uses an enormous domain pool. The reconstructed images in [6] show the coding to be of high quality and in fact, the authors claim that their algorithm gives the best results of any fractal block coder in the literature (we note that these results have been since superseded by hybrid transform-based coders such as those of [12] and [13]). The computational requirements for this scheme are quite large due to the size of the domain pool and the increased freedom in partitioning the image. Image encoding times were as high as 46 CPU-hours on a Silicon Graphics Personal IRIS 4D/35. In contrast, the SQS encodings required roughly 90 minutes apiece on a 133 MHz Intel Pentium PC.

The top line in Figure 4, 'Spline SQS', illustrates an alternative method for improving compressed image fidelity: changing bases. The Haar basis performs poorly for image compression because quantization of the coefficients introduces blocking artifacts into the decoded image. Switching to a smooth basis eliminates these artifacts. Our fractional Brownian motion texture model predicts that a basis with more vanishing moments than the Haar will perform better as an approximate K-L basis for the texture data and therefore will provide better encodings. The line 'Spline SQS' was generated using the 7-9 tap biorthogonal filter set from [26]. The domain pool was the same as for the Haar SQS scheme and the quadtree block coder. As can be seen in Figure 5, there is a substantial improvement in perceived image quality over the Haar SQS scheme. The blocking artifacts, a hallmark of fractal block coding schemes, have been completely eliminated, and the PSNR has increased by 1 to 2 dB over the Haar SQS scheme.

For comparison, the fourth line in Figure 4 shows the performance of the wavelet transform portion of the SQS coder alone. This baseline wavelet scheme is identical to the SQS coder except that no subtrees are self-quantized.

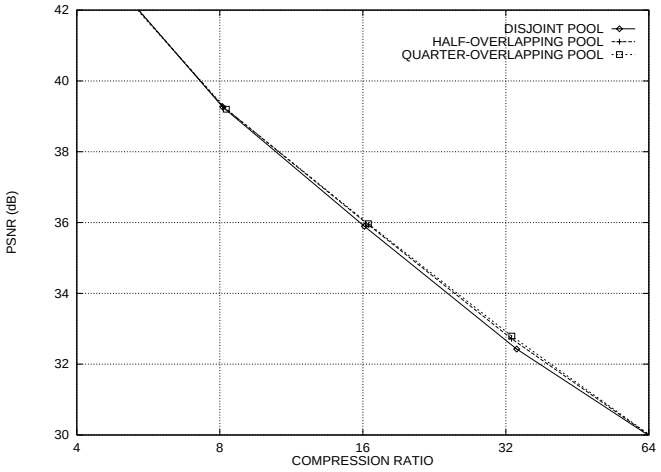

Fig. 6. Self-quantization of subtrees (SQS) coder performance for the 512×512 Lena image using the disjoint domain pool, the half-overlapping domain pool, and the quarter-overlapping domain pool. The spline variant basis of [26] was used in all cases.

It uses the same scalar quantizers and the same Lagrange multiplier bit allocation algorithm as SQS. We see that self-quantization of subtrees yields a modest improvement over the baseline transform coder at high bit rates. Our results below indicate that this improvement is due largely to the ability of self-quantization to efficiently represent smooth regions.

C. Larger Domain Pools

Experiments in [7] show the performance of fractal block coders improves when larger domain pools are used. Figure 6 shows PSNR's as a function of compression ratio for the 512×512 Lena image SQS encoded using a disjoint domain pool at each scale, a half-overlapping domain pool at each scale, and a quarter-overlapping domain pool, $\mathcal{D} = \{(2^{D-2}m, 2^{D-2}n) | 0 \leq m, n < 2^{N-D+2}\}$, at each scale. We see that increasing the domain pool size yields a slight improvement in coder performance. Increasing the domain pool size results in a considerable increase in com-

putational complexity. Using the half-overlapping domain pool increases the quantization search complexity by a factor of four over the disjoint pool, and using the quarter-overlapping pool increases the the complexity by a factor of sixteen over the disjoint pool. When we use these overlapping domain pools we no longer are guaranteed decoder convergence in a finite number iterations since our basis is biorthogonal. This adds additional complexity to the decoding process since convergence requires additional iterations. In our experiments we restricted the SQS gain factors to $[-2, 2]$. Although these limits are too large to guarantee convergence, we saw no evidence of convergence problems with the larger pools.

The use of larger domain pool allows more accurate quantization of subtrees. The cost of storing the quantization parameters increases, though, so there is a tradeoff. While image encodings improve slightly for the Lena image with increased codebook size, they decrease slightly for the standard "mandrill" test image. Increased codebook size does not necessarily lead to performance gains, since adding additional translates of domain subtrees to the pool leads to duplicate or near-duplicate codewords in the codebook. Image features invariant under translation, including the straight edges and constant regions that motivated fractal coding in the first place, give rise to such duplicate codewords. Duplicate codewords increase the cost of code words but contribute nothing to the reduction of distortion. The result is an inefficient codebook. This problem becomes more acute as the domain pool shifts become finer.

The reason that the use of larger domain pools yields such different results for block-based coders and SQS coders has to do with the relative efficiency with which block-based schemes and SQS schemes store block offsets and scaling function coefficients. When the codebook is small, very few large subtrees can be self-quantized accurately. Adaptive block coders quantize primarily small blocks and must spend a relatively large fraction of their bit budgets coding the associated DC values. As discussed in section IV-B, the method used by standard fractal block coders to encode these DC values is inefficient. When the codebook is enlarged, larger blocks can be self-quantized, and fewer DC coefficients need to be coded. The shift in bits from the inefficient DC quantization to the more efficient block quantization results in improved performance. Our SQS coder encodes both subtrees and coarse-scale coefficients efficiently, so increasing the domain pool does not yield a similar improvement.

D. Zerotrees

Recent wavelet-based image coders [20] [35] have shown that zerotrees, wavelet subtrees whose coefficients are all nearly zero, are a common feature of natural images. The use of zerotrees allows coders to take advantage of the localization of image energy in space. Zerotrees are trivially self-similar, so they can be encoded relatively cheaply via self-quantization. We conjecture that much of fractal coders' effectiveness is due to their ability to effectively represent zerotrees.

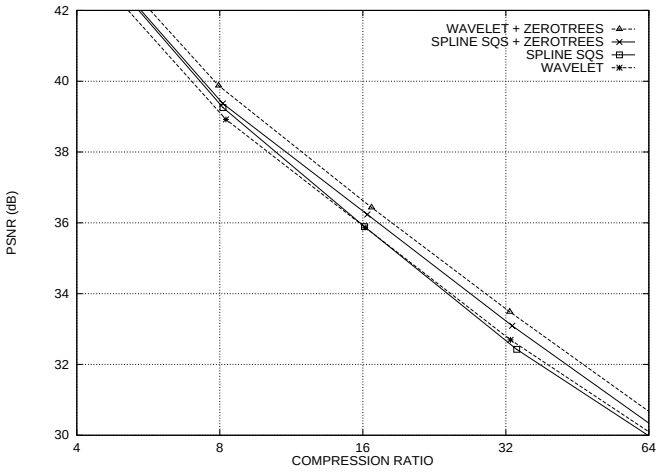

Fig. 7. Baseline wavelet coder performance vs. self-quantization of subtrees (SQS) with the disjoint domain pool for the 512×512 Lena image. PSNR's are shown for both unmodified and zerotree-enhanced versions of these coders. The spline variant basis of [26] was used in all cases.

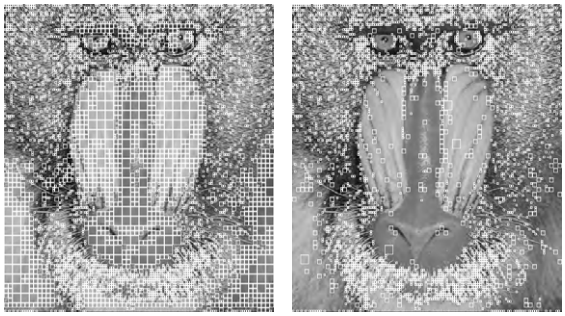

Fig. 8. The white squares in above images correspond to the self-quantized subtrees used in compressing the 512×512 mandrill image. The squares in the image on the left correspond to the support of the self-quantized subtrees used in a standard self-quantization of subtrees (SQS) scheme with a compression ratio of 8.2:1 (PSNR = 28.0 dB). The squares in the image on the right correspond to the support of the self-quantized subtrees used in a zerotree-enhanced SQS scheme with a compression ratio of 8.4:1 (PSNR = 28.0 dB).

We test this hypothesis by examining the results of incorporating a separate inexpensive zerotree codeword into our codebook. We implement this zerotree quantization in a manner similar to [35] by adding a low-cost codeword for an all-zero range block to our SQS codebook. Although zerotree quantization of subtrees is less accurate in general than self-quantization, zerotrees are much cheaper to code.

In our experiments the addition of zerotrees to our codebook results in a modest increase in the performance of our coder. Figure 7 shows the results of the zerotree enhancement for the Lena image. Images compressed with and without zerotrees look similar, but they differ dramatically in the sets of subtrees that are self-quantized. The white boxes in the first image in Figure 8 show the range subtrees that are self-quantized when no zerotrees are used. 58% of all coefficients in the image belong to self-quantized subtrees. Self-quantization takes place primarily along locally straight edges and locally smooth regions, with some sparse self-quantization in the textured fur. This is consis-

tent with our analysis of the fBm texture model.

The second image in Figure 8 shows the self-quantized subtrees in a zerotree-enhanced SQS coder. Only 23% of the coefficients are contained in self-quantized subtrees, and these coefficients are primarily from regions containing locally straight edges. Most of the self-quantized subtrees in the first image can be closely approximated by zerotrees; we obtain similar results for other test images.

Adding zerotrees to our baseline wavelet coder leads to a significant performance improvement, as can be seen in Figure 7. In fact, the performance of the wavelet coder with zerotrees is superior to or roughly equivalent to that of the zerotree-enhanced SQS scheme for all images tested.

On the whole, once the zerotree codeword is added to the codebook, self-quantization actually *diminishes* coder performance. The reason is simple: the gains from self-quantization do not balance out the increased side information costs.

Self-quantization of subtrees is more effective than scalar wavelet quantization for coding self-similar features. As we observed above, most of these self-quantized subtrees are zerotrees or can be closely approximated by zerotrees. The benefits of self-quantization are due largely to its relative effectiveness at coding zerotrees. Adding a zerotree codeword to the baseline wavelet coder provides it with an even cheaper way to quantize these zerotrees.

Both the SQS coder and the wavelet coder must transmit side information to indicate for each subtree what kind of quantization was used. The SQS coder's greater flexibility in quantization results in increased side information costs. For example, the side information cost for the mandrill image in Figure 8 for the zerotree-enhanced SQS coder is 2937 bytes. The side information cost for the zerotree-enhanced wavelet coder at a similar compression ratio is only 1940 bytes. The increased side information costs eat up the benefits that self-quantization provides for straight edges. The result is that self-quantization yields constant or diminished coder performance.

It is important to observe that the structure of the subtrees we are using significantly reduces the performance of the zerotree-enhanced wavelet coder. In adherence to the Jacquin-style block coder framework, we have limited our attention to subtrees containing components from all three subband orientations at each scale. The zerotree/wavelet coder of Xiong et al. [35] obtains PSNR's of over 1 dB better than those reported here through the use of oriented zerotrees. The Rinaldo-Calvagno [13] coder treats coefficients with different orientations separately, so it too obtains some advantage over our zerotree structure. However, the code words in Rinaldo-Calvagno coder do not extend across multiple scales and as a result they are unable to take advantage of the fact that edges and smooth regions have structures that persist across scales. Whether using oriented range and domain subtrees yields a substantial improvement in performance remains a topic for future research.

VI. Conclusion

We began this paper with the question, Why do fractal block coders work? The wavelet framework we have presented makes the answer much more clear. Up to the DC component, the block quantization performed by fractal block coders is equivalent to the self-quantization of a Haar subtree. We have shown that

- Self-quantization is effective for quantizing isolated straight edges and zerotrees because these features are self-similar. A significant fraction of subtrees in natural images are well-approximated by zerotrees, suggesting that fractal coders' ability to encode zerotrees cheaply is a major source of their effectiveness.
- Effective self-quantization of textures requires that these textures possess a rapidly decaying power spectrum. Because of the inefficient distribution of code words used by self-quantization, transform coding is more effective than self-quantization for coding textures.
- The use of smooth wavelet bases with 2 or more vanishing moments for self-quantization results in a substantial improvement in coder performance over Haar-based schemes. The improvement can be seen both in PSNR and in subjective image quality. Smooth bases eliminate blocking artifacts, and the extra vanishing moments lead to better transform energy packing properties in textured regions.

A fundamental weakness of fractal block coders is that the coders possess no control over the codebook. Codewords are too densely clustered around the very common all-zero subtree and too sparsely distributed elsewhere. This dense clustering of near-zerotrees increases codeword cost but contributes very little to image fidelity.

Some authors have addressed the problem of codebook inefficiencies by augmenting fractal codebooks [36]. While this codebook supplementation adds codewords in the sparse regions, it does not address the problem of overly dense clustering of code words around zero. At 0.25 bits per pixel, over 80 percent of all coefficients in the 512×512 Lena image are assigned to zerotrees by our zerotree-augmented wavelet coder. Hence only about 20 percent of the fractal coder's codewords are significantly different from a zerotree. This redundancy is costly, since when using self-quantization we pay a substantial number of bits to differentiate between these essentially identical zero code words. Relatively little attention has been paid to this problem of redundancy. A codebook pruning strategy of Signes [37] is a promising first attempt. An alternative strategy would be to adaptively eliminate from the domain pool any subtrees that are subsets of larger subtrees that have been quantized to zero.

Our analysis suggests that the primary advantage that fractal block coders have over simple wavelet transform coders is their ability to efficiently represent zerotrees. Zerotree-augmented wavelet coders share this ability and are not burdened with the codebook inefficiencies inherent to the fractal block coders we have described. Moreover, the computational complexity of zerotree-augmented wavelet coders is an order of magnitude lower than that of SQS coders. Addressing this problem of codebook ineffi-

ciency is a topic for future research.

Acknowledgments

This work was supported in part by a National Science Foundation Postdoctoral Research Fellowship and by DARPA as administered by the AFOSR under contract DOD F4960-93-1-0567.

References

[1] Arnaud Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations", *IEEE Trans. Image Proc.*, vol. 1, no. 1, pp. 18–30, Jan. 1992.

[2] Michael F. Barnsley and Arnaud Jacquin, "Application of recurrent iterated function systems to images", *Proc. SPIE*, vol. 1001, pp. 122–131, 1988.

[3] A. Jacquin, "Fractal image coding based on a theory of iterated contractive image transformations", in *Proc. SPIE Visual Comm. and Image Proc.*, 1990, pp. 227–239.

[4] M. F. Barnsley and S. Demko, "Iterated function systems and the global construction of fractals", *Proc. Royal Society of London*, vol. A399, pp. 243–275, 1985.

[5] F. Davoine, E. Bertin, and J-M. Chassery, "From rigidity to adaptive tesselations for fractal image compression: comparative studies", in *IEEE IMDSP*, Sept. 1993.

[6] Yuval Fisher and Spencer Menlove, "Fractal encoding with HV partitions", in *Fractal Compression: Theory and Application to Digital Images*, Yuval Fisher, Ed. Springer Verlag, New York, 1994.

[7] Yuval Fisher, "Fractal image compression with quadtrees", in *Fractal Compression: Theory and Application to Digital Images*, Yuval Fisher, Ed. Springer Verlag, New York, 1994.

[8] E. W. Jacobs, Y. Fisher, and R. D. Boss, "Image compression: a study of the iterated transform method", *Signal Processing*, vol. 29, no. 3, pp. 251–263, Dec. 1992.

[9] Dietmar Saupe, "Accelerating fractal image compression by multi-dimensional nearest neighbor search", in *Proc. Data Compression Conference, Snowbird, Utah*, James A. Storer and Martin Cohn, Eds. IEEE Computer Society, Mar. 1995, pp. 222–231.

[10] B. Hurtgen and T. Hain, "On the convergence of fractal transforms", in *Proc. ICASSP*, 1994, vol. 5, pp. 561–564.

[11] John Kominek, "Convergence of fractal encoded images", in *Proc. Data Compression Conference, Snowbird, Utah*, James A. Storer and Martin Cohn, Eds. IEEE Computer Society, Mar. 1995, pp. 242–251.

[12] Kai Uwe Barthel, Jörg Schüttermeyer, Thomas Voyé, and Peter Noll, "A new image coding technique unifying fractal and transform coding", in *IEEE ICIP*, Austin, Texas, Nov. 1994.

[13] Roberto Rinaldo and Giancarlo Calvagno, "Image coding by block prediction of multiresolution subimages", *IEEE Transactions on Image Processing*, vol. 4, no. 7, pp. 909–920, July 1995.

[14] Alex Pentland and Bradley Horowitz, "A practical approach to fractal-based image compression", in *Proc. Data Compression Conference, Snowbird, Utah*, James A. Storer and Martin Cohn, Eds. IEEE Computer Society, Mar. 1991, pp. 176–185.

[15] Geoffrey M. Davis, "Self-quantization of wavelet subtrees: a wavelet-based theory of fractal image compression", in *Proc. Data Compression Conference, Snowbird, Utah*, James A. Storer and Martin Cohn, Eds. IEEE Computer Society, Mar. 1995, pp. 232–241.

[16] H. Krupnik, D. Malah, and E. Karnin, "Fractal representation of images via the discrete wavelet transform", in *IEEE 18th Conv. of EE in Israel*, Tel-Aviv, Mar. 1995.

[17] Axel van de Walle, "Merging fractal image compression and wavelet transform methods", in *Fractal Image Coding and Analysis: a NATO ASI Series Book*, Yuval Fisher, Ed. Springer Verlag, New York, 1996.

[18] Yuval Fisher, *Fractal Compression: Theory and Application to Digital Images*, Springer Verlag, New York, 1994.

[19] Arnaud Jacquin, "Fractal image coding: a review", *Proc. IEEE*, vol. 81, no. 10, pp. 1451–1465, Oct. 1993.

[20] J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients", *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3445–3462, Dec. 1993.

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

[22] Geoffrey M. Davis, "Self-Quantization of Wavelet Subtrees", in *Proc. SPIE Wavelet Applications in Signal and Image Proc. III, San Diego*, Andrew F. Laine and Michael A. Unser, Eds., July 1995, vol. 2569, pp. 294–307.

[23] Lars M. Lundheim, "A discrete framework for fractal signal modeling", in *Fractal Compression: Theory and Application to Digital Images*, Yuval Fisher, Ed. Springer Verlag, New York, 1994.

[24] Yuval Fisher, Bill Jacobs, and Roger Boss, "Fractal image compression using iterated transforms", in *Image and Text Compression*, J. Storer, Ed., pp. 35–61. Kluwer Academic, 1992.

[25] Martin Vetterli and Jelena Kovačević, *Wavelets and Subband Coding*, Prentice Hall, Englewood Cliffs, NJ, 1995.

[26] M. Antonini, M. Barlaud, and P. Mathieu, "Image Coding Using Wavelet Transform", *IEEE Trans. Image Proc.*, vol. 1, no. 2, pp. 205–220, Apr. 1992.

[27] G. E. Øien, Z. Baharav, S. Lepsøy, and E. Karnin, "A new improved collage theorem with applications to multiresolution fractal image coding", in *Proc. ICASSP*, 1994.

[28] Z. Baharav, D. Malah, and E. Karnin, "Hierarchical interpretation of fractal image coding and its application to fast decoding", in *Proc. Digital Signal Processing Conference*, Cyprus, July 1993.

[29] Alex Pentland, "Fractal-based description of natural scenes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 661–673, 1984.

[30] David J. Field, "Scale-invariance and self-similar 'wavelet' transforms: an analysis of natural scenes and mammalian visual systems", in *Wavelets, Fractals, and Fourier Transforms*, M. Farge, J. C. R. Hunt, and J. C. Vassilicos, Eds. Oxford University Press, Oxford, 1993.

[31] Patrick Flandrin, "Wavelet analysis and synthesis of fractional Brownian motion", *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 910–917, Mar. 1992.

[32] A. H. Tewfik and M. Kim, "Correlation structure of the discrete wavelet coefficients of fractional Brownian motion", *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 904–909, Mar. 1992.

[33] Allen Gersho, "Asymptotically optimal block quantization", *IEEE Transactions on Information Theory*, vol. IT-25, no. 4, pp. 373–380, July 1979.

[34] Paul L. Zador, "Asymptotic quantization error of continuous signals and the quantization dimension", *IEEE Transactions on Information Theory*, vol. IT-28, no. 2, pp. 139–149, Mar. 1982.

[35] Zixiang Xiong, Kannan Ramchandran, and Michael T. Orchard, "Space-frequency quantization for wavelet image coding", *preprint*, 1995.

[36] Mohammad Gharavi-Alkhansari and Thomas Huang, "Generalized image coding using fractal-based methods", in *Proc. ICIP*, 1994, pp. 440–443.

[37] Julien Signes, "Geometrical interpretation of IFS based image coding", in *Fractal Image Coding and Analysis: a NATO ASI Series Book*, Yuval Fisher, Ed. Springer Verlag, New York, 1996.