Combining Fractal Image Compression and Vector Quantization

Raouf Hamzaoui, Dietmar Saupe

The authors are with the Institut für Informatik, Universität Leipzig, Augustusplatz 10–11, D-04109 Leipzig, Germany. Email:hamzaoui,saupe@informatik.uni-leipzig.de. This work was done while the authors were with the Institut für Informatik, Universität Freiburg, Germany.

Abstract

In fractal image compression, the code is an efficient binary representation of a contractive mapping whose unique fixed point approximates the original image. The mapping is typically composed of affine transformations, each approximating a block of the image by another block (called domain block) selected from the same image. The search for a suitable domain block is time-consuming. Moreover, the rate-distortion performance of most fractal image coders is not satisfactory. We show how a few fixed vectors designed from a set of training images by a clustering algorithm accelerate the search for the domain blocks and improve both the rate-distortion performance and the decoding speed of a pure fractal coder, when they are used as a supplementary vector quantization codebook. We implemented two quadtree-based schemes: a fast top-down heuristic technique and one optimized with a Lagrange multiplier method. For the 8 bits per pixel (bpp) luminance part of the 512×512 Lenna image, our best scheme achieved a peak-signal-to-noise ratio of 32.50 dB at 0.25 bpp.

Keywords

Fractal coding, mean shape-gain vector quantization, clustering, quadtrees, Lagrange multipliers.

I. Introduction

The mathematical principle of fractal image compression consists of approximating an image f, seen as an element of a complete metric space (X, δ), by the unique fixed point $f_W = W(f_W)$ of a contractive mapping $W : X \to X$ [1]. The code for f is a compact representation of W. The decoding is based on the contraction mapping principle, which ensures that f_W can be generated as the limit point of the sequence of iterates $\{f_n\}_{n \geq 0}$, where $f_{n+1} = W(f_n)$ and f_0 is an arbitrary initial image. The mapping W is found by minimizing the distance δ between f and $W(f)$ in a set of many candidate contractive mappings. This approach is motivated by the *Collage Theorem* [1], which states that

$$\delta(f, f_W) \leq \frac{1}{1 - L(W)} \delta(f, W(f)),$$

where $L(W) \in [0, 1)$ is the contraction factor of W. In practice, images are digitized both spatially and in amplitude. In this context, the metric space (X, δ) is taken as the n-dimensional space $\mathbb{R}^n$ together with a distance derived from a vector norm (e.g., the L_2 norm). A fully automated scheme based on this idea is due to Jacquin [20]. Since then, several researchers have proposed modifications to the original algorithm, which improved its performance considerably.

Fractal image coders allow high compression ratios. Moreover, the decoding is simple. On the other hand, the encoding is slow, and the rate-distortion performance is inferior to that

of the state-of-the-art in image coding (e.g., zerotree-based wavelet coders). In this paper, we tackle the weaknesses of fractal image compression by combining it in a natural and efficient way with mean shape-gain vector quantization (MSGVQ) [33], [12], [15]. Previous hybrid approaches enhanced fractal image compression by combining it with transform coding (notably discrete cosine transform in [2] and wavelet transform in [6], [26]). The rate-distortion results of our scheme, although not reaching those of the state-of-the-art wavelet coders, were better than those of conventional fractal coding and conventional MSGVQ.

We now describe a generic fractal coder and introduce terminology. The support of a sampled square grey-scale digital image is partitioned into disjoint $2^n \times 2^n$ blocks $R_1, R_2, \ldots, R_{n_R}$ called *range blocks*. In the paper, a $2^n \times 2^n$ block may be seen as a two-dimensional array or as a vector in the linear vector space $\mathbb{R}^{2^{2n}}$. The vector is formed, for example, by reading the pixel intensities of the array row by row, left to right, and top to bottom. The *domain pool* is a set $\{D_1, \ldots, D_{n_D}\}$ of square blocks called *domain blocks*, which are taken from the same image as the range blocks. It may consist of all $2^{n+1} \times 2^{n+1}$ square blocks whose upper-left pixels are situated on a lattice with a fixed vertical and horizontal spacing. For each range block R_i, $i \in \{1, \ldots, n_R\}$ a transformation W_i is found, which approximates the range block R_i by the block

$$W_i(R_i) = s(R_i)S^{(n+1)}\tau(R_i)D(R_i) + o(R_i)\mathbf{1}. \tag{1}$$

Here $D(R_i) \in \{D_1, \ldots, D_{n_D}\}$ is a domain block, $\tau(R_i) \in \{\tau_1, \ldots, \tau_8\}$ rotates or flips the domain block (the eight isometries of the square may be used), $S^{(n+1)} : \mathbb{R}^{2^{2(n+1)}} \to \mathbb{R}^{2^{2n}}$ is a downsampling operator, which via pixel averaging shrinks a block to match the range block size, $s(R_i) \in \{s_1, \ldots, s_{n_s}\} \subset \mathbb{R}$ is a *scaling factor*, $o(R_i) \in \{o_1, \ldots, o_{n_o}\} \subset \mathbb{R}$ is an *offset*, and $\mathbf{1}$ is the block with intensity 1 at every pixel. To ensure the convergence of the decoding, the scaling factor is restricted to the interval $[-s_{max}, s_{max}]$, where $0 < s_{max} < 1$ (see [9] p. 51). The mapping W is determined by the partitioning of the image support into range blocks and by the transformations W_i, each of which is completely specified by the location of $D(R_i)$ and the respective indices of $\tau(R_i), s(R_i)$, and $o(R_i)$.

For a given range block R_i, each approximation with a tuple (s_k, o_m, D_j, τ_p) yields a *local collage error*

$$d(R_i, s_k, o_m, D_j, \tau_p) = \|R_i - (s_k S^{(n+1)}\tau_p D_j + o_m \mathbf{1})\|_2^2$$

and a bit rate

$$r(R_i, s_k, o_m, D_j, \tau_p) = \begin{cases} \lceil \log_2 n_s \rceil + \lceil \log_2 n_o \rceil + \lceil \log_2 n_D \rceil + 3 & \text{if } s_k \neq 0 \\ \lceil \log_2 n_s \rceil + \lceil \log_2 n_o \rceil & \text{otherwise,} \end{cases}$$

since in the latter case the information for the domain and the isometry is redundant. According to the Collage Theorem, R_i and $W_i(R_i)$ should be as close as possible. For this reason, W_i is commonly determined by finding a tuple $(s(R_i), o(R_i), D(R_i), \tau(R_i))$ minimizing the local collage error.

An extension to this scheme is obtained by allowing the blocks to have a variable size. One can use, for example, a *quadtree partition* where the image support is hierarchically partitioned into square blocks of various sizes. In Fisher's implementation [10], one starts with a partition into square blocks of a given maximal size and recursively splits those square blocks that do not fulfill the *root mean square* (rms) error criterion

$$[\frac{1}{2^{2n}} d(R_i, s(R_i), o(R_i), D(R_i), \tau(R_i))]^{1/2} \leq t,$$

where t is a given threshold.

There is a strong similarity between fractal image compression and vector quantization (VQ). At the encoder, both methods use a vector codebook in which best matches are found for the blocks of the original image. In VQ, the codebook blocks are typically generated by a clustering algorithm from a set of training images. These blocks are needed at the decoder to construct an approximation of the original image. In fractal image compression, the codebook (the domain pool) is formed from the original image. However, this codebook is not explicitly known by the decoder, which receives only the positions of the domain blocks selected in the encoding. Lepsøy *et al.* [23], [36], [24] pointed out that since a range block is encoded by an affine transformation of a domain block (see equation (1)), the actual codebook in fractal image compression is a *product code codebook* similar to the one used in MSGVQ. They showed [24] that for the same product codebook size, the rate-distortion performance of standard MSGVQ is better than that of the generic fractal image compression scheme. They concluded that the domain pool, though image adaptive, is inferior to a fixed trained VQ codebook. In [15], we showed the benefits of incorporating negative gains and the eight isometries of the square in quadtree-based MSGVQ. Indeed, the isometries enhance the rate-distortion performance by enlarging the vector codebook without additional CPU memory requirements.

As in VQ, the time consuming part in fractal encoding is the searching for the best matches in the codebook. In [13], we drastically reduced the complexity of the encoding by combining a distance-based clustering scheme with the mean classification of Fisher *et al.* [19], [10]. Cluster centers were computed either adaptively from the original image or from a set of training images. In this paper, we show that not only the encoding speed, but also both the rate-distortion performance and the decoding speed of fractal image compression can be improved by considering the cluster centers as an integral part of the coder. When the cluster centers are computed from a set of training images, we use them as a MSGVQ vector codebook and encode a range block with a cluster center instead of a domain block if the cluster center provides a satisfactory approximation.

The rest of the paper is organized as follows. Section II presents our clustering-based fast encoding scheme [13]. Section III compares our hybrid scheme to pure fractal coding for top-down quadtree partitioning [17]. Section IV explains how the rate-distortion performance of the hybrid scheme can be significantly improved by a Lagrange multiplier method, which jointly optimizes the quadtree partition and the coding of the image blocks. In the last section, we summarize, discuss the results and compare them to the state-of-the-art in image coding.

II. Fast encoding with clustering

The time complexity of the encoding is a major drawback of fractal image compression. For each $2^n \times 2^n$ range block R_i, one has to find a tuple $(s(R_i), o(R_i), D(R_i), \tau(R_i))$ that minimizes the local collage error $d(R_i, s_k, o_m, D_j, \tau_p)$ over all $k \in \{1, \ldots, n_s\}$, $m \in \{1, \ldots, n_o\}$, $j \in \{1, \ldots, n_D\}$, and $p \in \{1, \ldots, 8\}$. Several authors [31], [3], [9] simplify this combinatorial problem by considering for a given domain-isometry pair (D_j, τ_p) only the scaling factor-offset pair yielded by least-squares minimization. This is done as follows. Given j and p, let $C_{j,p}$ denote the codebook vector $S^{(n+1)}(\tau_p D_j)$. Assuming that $C_{j,p}$ is not in the linear span of $\mathbf{1}$, first solve the least-squares problem

$$e(R_i, C_{j,p}) = \min_{s,o \in \mathbb{R}} d(R_i, s, o, D_j, \tau_p). \tag{2}$$

This gives

$$s = \frac{\frac{1}{2^n \times 2^n} \langle R_i - \mu(R_i)\mathbf{1}, C_{j,p} - \mu(C_{j,p})\mathbf{1} \rangle}{\nu(C_{j,p})} \tag{3}$$

and

$$o = \mu(R_i) - s\mu(C_{j,p}). \tag{4}$$

Here $\mu(B)$ and $\nu(B)$ denote the mean and the variance of a block B, respectively, and $\langle , \rangle$ denotes the inner product. Then quantize the solutions s and o in $\{s_1, \ldots, s_{n_s}\}$ and $\{o_1, \ldots, o_{n_o}\}$, respectively. This yields a scaling factor $\hat{s}$ and an offset $\hat{o}$. Finally, for each pair (D_j, τ_p) compute the collage error for $\hat{s}$ and $\hat{o}$ only. We say then that the range block R_i is *compared* to the block $\tau_p D_j$.

Other supplementary techniques for reducing the time complexity of the encoding have been devised. They are essentially based on:

- **Classification.** After a preprocessing step where ranges and domains are classified, for a given range, only domains of a "similar" class are examined [20], [19], [10], [5], [4], [35], [25].
- **Domain pool reduction.** Domain blocks with low intensity variance are discarded from the domain pool [41]. For a given range, the domain candidates are restricted to those domains whose *spatial locations* are close to the range location [32], [45], [18].
- **Nearest neighbor search.** The ranges and domains are assigned feature vectors, and for a given range only domains that are close in the feature space are considered [39], [40].
- **Multiresolution approaches.** Range blocks and domain blocks are downsampled by pixel averaging to a coarse resolution where the search can be carried out more rapidly [7], [29].
- **Fast Fourier Transform method** [43]. This is an acceleration technique based on the cross-correlation property of the Fourier transform.

A comprehensive review of these methods can be found in [40], [42]. In the following, we describe the classification technique of Fisher *et al.* [19], [10] and the nearest neighbor search approach [39], [40], which are used in the remaining of the paper.

A. Classification by mean and variance

As demonstrated in Figure 1, for every $2n \times 2n$ block B there exists a unique isometry $\tau_B \in \{\tau_1, \ldots, \tau_8\}$ having the following property. If we denote by B_i^*, $i = 1, 2, 3, 4$ the average pixel intensities of the four quadrants (upper left, upper right, lower left, and lower right) of the block $\tau_B B$, then these averages are ordered in one of the three *canonical classes*

$$\text{Class 1: } B_1^* \geq B_2^* \geq B_3^* \geq B_4^*,$$
$$\text{Class 2: } B_1^* \geq B_2^* \geq B_4^* \geq B_3^*,$$
$$\text{Class 3: } B_1^* \geq B_4^* \geq B_2^* \geq B_3^*.$$

Once the isometry τ_B has been determined, there are 24 possible orderings of the variances of the 4 quadrants in $\tau_B B$ which define 24 subclasses for each canonical class. Thus, there are 72 classes in all for B.

Now in (2), for a given range R and domain D, instead of testing all eight isometries τ_p, Fisher *et al.* suggest to use only $\tau_R^{-1}\tau_D$ and $\tau_{-R}^{-1}\tau_D$, which amounts to encoding $\tau_R R$ by $\tau_D D$ and $\tau_{-R}R$ by $\tau_D D$. The optimal scaling factor is expected to be nonnegative in the first case and nonpositive in the second case. This reduces the domain-isometry pairs from $8n_D$ to $2n_D$ with almost no quality loss. Furthermore, one can limit the number of domains examined by searching in a variable number of classes as follows.

- One-class search: R is compared to

1. all blocks $\tau_R^{-1}\tau_D D$ such that R and D have the same canonical class and the same variance subclass.
2. all blocks $\tau_{-R}^{-1}\tau_D D$ such that $-R$ and D have the same canonical class and the same variance subclass.

- Three-class search: R is compared to

1. all blocks $\tau_R^{-1}\tau_D D$ such that R and D have the same variance subclass.
2. all blocks $\tau_{-R}^{-1}\tau_D D$ such that $-R$ and D have the same variance subclass.

- Twenty-four-class search: R is compared to

1. all blocks $\tau_R^{-1}\tau_D D$ such that R and D have the same canonical class.
2. all blocks $\tau_{-R}^{-1}\tau_D D$ such that $-R$ and D have the same canonical class.

- Seventy-two-class search: Here all domains are checked. Thus, R is compared to all blocks $\tau_R^{-1}\tau_D D$ and all blocks $\tau_{-R}^{-1}\tau_D D$. This is called *full search.*

Although this classification provides a variable number of classes, there is no natural notion of neighborhood between classes, so that one cannot search in a "near" class if a good match is not found in the selected class. This problem occurs in particular when a class is empty.

B. Nearest neighbor search

Saupe [39] showed that the problem of finding a domain-isometry pair minimizing the least-squares error in (2) can be replaced by a nearest neighbor search in multi-dimensional Euclidean space. It is more convenient to see the problem from that angle because finding closest neighbors is a well studied problem for which many efficient methods exist [12]. In the following, we only restate the main result. Details can be found in [39], [40].

Let $n \geq 2$ and let O be the orthogonal projection operator which maps $\mathbb{R}^n$ onto the orthogonal complement $\mathcal{C}^\perp$, where $\mathcal{C}$ is the linear span of $\mathbf{1}$. Let $\mathcal{X} = \mathbb{R}^n \setminus \mathcal{C}$. For $Z \in \mathcal{X}$, let

$$\phi(Z) = \frac{OZ}{||OZ||} = \frac{Z - \mu(Z)\mathbf{1}}{\|Z - \mu(Z)\mathbf{1}\|}.$$

Define the function $\Delta : \mathcal{X} \times \mathcal{X} \to [0, \sqrt{2}]$ by

$$\Delta(R, C) = \min\left(||\phi(R) + \phi(C)||, ||\phi(R) - \phi(C)||\right).$$

Then we have [39]:

Theorem 1: For a range block R and a codebook block $C_{j,p}$ both in $\mathcal{X}$, the least-squares error $e(R, C_{j,p})$ is given by

$$e(R, C_{j,p}) = \langle R, \phi(R)\rangle^2 \, g(\Delta(R, C_{j,p}))$$

where $g(\Delta) = \Delta^2(1 - \frac{\Delta^2}{4})$.

Since g is an increasing function in $[0, \sqrt{2}]$, it follows from the theorem that we can replace the computation and minimization of $8n_D$ least-squares errors $e(R, C_{j,p})$ by the search for a nearest neighbor of the *feature vector* $\phi(R) \in \mathbb{R}^n$ in the set of $2 \times 8n_D$ feature vectors $\pm\phi(C_{j,p}) \in \mathbb{R}^n$.

C. Distance-based clustering

In this section, we present a fast fractal encoding scheme introduced in [13]. We combine clustering of the feature vectors of the codebook blocks in multi-dimensional Euclidean space with the mean classification of Fisher *et al.*, yielding a classification with a notion of distance which is not given in traditional classification schemes.

We define clusters as *Voronoi* (nearest neighbor) *cells* in Euclidean multi-dimensional space. Each cell $\Omega_i \subset \mathbb{R}^n$, $i = 1, \ldots, n_m$ is represented by a *center* $\mathbf{m}_i \in \mathbb{R}^n$ and given by

$$\Omega_i = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{m}_i\| \leq \|\mathbf{x} - \mathbf{m}_k\| \text{ for all } k = 1, \ldots, n_m\}.$$

To ensure that the cells are disjoint we add the tie-breaking rule: If $\arg\min_i \|\mathbf{x} - \mathbf{m}_i\|$ is not unique, then assign $\mathbf{x}$ to the cell with the smallest index. We also denote the $k-$th nearest neighbor of $\mathbf{x}$ in the set of centers $\{\mathbf{m}_1, \ldots, \mathbf{m}_{n_m}\}$ by $\Omega^k(\mathbf{x})$; ties are decided as before.

Suppose that the cluster centers are available. We can speed up the encoding as follows. In a preprocessing step, we determine the cluster of each feature vector $\pm\phi(C_{j,p})$, $j = 1, \ldots, n_D$, $p = 1, \ldots, 8$. In a one-cluster search, we compare a range block R only to the blocks $\tau_p D_j$ for which at least one of the feature vectors $\pm\phi(C_{j,p})$ is in the cluster whose center is $\Omega^1(\phi(R))$. This strategy

is motivated by the remark following Theorem 1 and the observation that the nearest feature vectors to $\phi(R)$ are likely to be in this cluster. We can search in more clusters by considering the next closest centers to $\phi(R)$. This produces more accurate collage codings at the expense of increased time.

It is reasonable to construct the cluster centers such that they approximate the probability density function of the stochastic source producing the feature vectors. Kohonen's Self-Organizing Map (SOM) [21] is particularly appropriate, since it can rapidly generate a high quality clustering.

Because of time and storage constraints it would be unwise to cluster all $2 \times 8n_D$ feature vectors $\pm\phi(C_{j,p})$. A solution to this problem is given by incorporating the concept of canonical classes of Fisher *et al.* Thus, as proposed by Fisher *et al.*, we use only the isometries $\tau_R^{-1}\tau_{D_j}$ and $\tau_{-R}^{-1}\tau_{D_j}$. In the first case, the best domain candidates for range R are determined by finding the nearest neighbors of $\phi(\tau_R R)$ in the set of feature vectors $\{\pm\phi(C_j),\ j = 1, \ldots, n_D\}$. Here the codebook vectors C_j are defined by

$$C_j = S^{(n+1)}\tau_{D_j} D_j,\ j = 1, \ldots, n_D.$$

But since $\tau_R R$ and $\tau_{D_j} D_j$ are both in canonical orientation, the scaling factor is likely to be nonnegative. Thus, as justified by the following proposition, we can ignore the feature vectors $-\phi(C_j)$.

Proposition 2: With the notations of Theorem 1 we have

$$\Delta(R, C_{j,p}) = \|\phi(R) - \phi(C_{j,p})\| \text{ if and only if } s \geq 0,$$

where s is the scaling factor obtained by least-squares optimization in equation (3).

Proof: We have $\Delta(R, C_{j,p}) = \|\phi(R) - \phi(C_{j,p})\|$ if and only if $\langle \phi(R), \phi(C_{j,p}) \rangle \geq 0$ or, equivalently, $\langle R - \mu(R)\mathbf{1}, C_{j,p} - \mu(C_{j,p})\mathbf{1} \rangle \geq 0$, which is equivalent to $s \geq 0$ from (3). ■

In the second case, with a similar argument, we obtain the best domain candidates by finding the nearest neighbors of $\phi(\tau_{-R}(-R))$ in the set of feature vectors $\{\phi(C_j),\ j = 1, \ldots, n_D\}$.

Our encoding algorithm is as follows. Let q be a search depth.

1. Construct a set of centers $\{\mathbf{m}_1, \ldots, \mathbf{m}_{n_m}\}$ by using the feature vectors $\phi(C_j)$ as training vectors in the SOM algorithm.
2. Find the cluster of each feature vector $\phi(C_j)$.
3. Three options can be used:

Option 1: Compare R to each block $\tau_R^{-1}\tau_{D_j}D_j$ such that $\phi(C_j)$ is in the union of the q clusters whose centers are $\Omega^k(\phi(\tau_R R)),\ k=1,\ldots,q$.

Option 2: Compare R to the blocks selected by Option 1 and also to each block $\tau_{-R}^{-1}\tau_{D_j}D_j$ such that $\phi(C_j)$ is in the union of the q clusters whose centers are $\Omega^k(\phi(\tau_{-R}(-R))),\ k=1,\ldots,q$.

Option 3: Determine the subset of the q smallest norms in the set

$$\{\|\phi(\tau_R R)-\mathbf{m}_k\|, \|\phi(\tau_{-R}(-R))-\mathbf{m}_k\|,\ k=1,\ldots,n_m\}.$$

For all k such that $\|\phi(\tau_R R)-\mathbf{m}_k\|$ is in this subset, compare R to each block $\tau_R^{-1}\tau_{D_j}D_j$ for which $\phi(C_j)\in\Omega_k$. For all k such that $\|\phi(\tau_{-R}(-R))-\mathbf{m}_k\|$ is in this subset compare R to each block $\tau_{-R}^{-1}\tau_{D_j}D_j$ for which $\phi(C_j)\in\Omega_k$. This option inspects only half as many clusters as Option 2. It is slower than Option 1; but it provides a better collage error.

Figure 2 shows the peak-signal-to-noise ratio (PSNR) and the encoding time as a function of the number of classes (respectively clusters) searched for the 512 × 512 Lenna image. The encoding time was measured on an SGI Indigo2 with an R4400 processor running at 150 MHz. The curves correspond to Option 2 (Both), Option 3 (Best), and the classification by mean and variance of Fisher *et al.* (Fisher). The range blocks had the fixed size (4 × 4). The domain pool consisted of 4096 disjoint 8 × 8 blocks. For a fair comparison with the classification of Fisher *et al.*, we used 72 cluster centers, which were designed with the SOM package [22]. We formed the initial centers from the set $\mathcal{T}=\{\phi(C_j), j=1,\ldots,n_D\}$ by the linear initialization procedure [21], [22], and we took the training vectors randomly from $\mathcal{T}$. More details on the implementation can be found in [13], [14]. Our acceleration technique clearly outperformed the classification of Fisher *et al.* For example, we obtained an encoding fidelity of 37.05 dB in 95 seconds with Option 2 whereas it took 758 seconds to reach 37.04 dB with the scheme of Fisher *et al.*

When the image partitioning is based on a quadtree scheme [10], up to four range sizes may be used (e.g., 4 × 4, 8 × 8, 16 × 16, and 32 × 32), leading to feature vectors of dimension as high as 1024. This can cause both a computational and a storage problem if the design of the centers and the subsequent clustering were to be done in such high dimensions. To cope with this difficulty, we first reduce the size of the blocks to 4 × 4 and then compute both the feature vectors and the cluster centers in the space $\mathbb{R}^{16}$. In this way, we have only one set of centers at the lowest dimension, which we use for classifying blocks of all sizes. However, once we select the domain-isometry candidates for a given range block, we compute the least-squares error at the real size. More precisely, assume that a set of cluster centers was designed in $\mathbb{R}^{16}$. Let n_{D^n}

be the number of domains D_j^n of size $2^{n+1} \times 2^{n+1}$:

In a preprocessing step: For all $j = 1, \ldots, n_{D^n},\ n = 2, 3, \ldots, n_{max}$

1. Compute the 16-dimensional vectors

$$C_j^n = S^{(3)} \cdots S^{(n)} S^{(n+1)} (\tau_{D_j^n} D_j^n).$$

2. Cluster the feature vectors $\phi(C_j^n) \in \mathbb{R}^{16}$.

Then for search depth q and Option 1 (the extension to the other options is straightforward), to encode a range block R^n of size $2^n \times 2^n,\ n \in \{3, \ldots, n_{max}\}$:

1. Compute $r^n = S^{(3)} \cdots S^{(n)} (\tau_{R^n} R^n)$.

2. Compute $\phi(r^n)$.

3. Compare R^n to each block $\tau_{R^n}^{-1} \tau_{D_j^n} D_j^n$ such that $\phi(C_j^n)$ is in the union of the q clusters whose centers are $\Omega^k(\phi(r^n)),\ k = 1, \ldots, q$.

The low dimensional cluster centers can be designed by the SOM from, for example, the set of feature vectors $\{\phi(C_j^n);\ n = 2, \ldots, n_{max},\ j = 1, \ldots, n_{D^n}\}$.

The rationale of the above algorithm lies in our expectation that if the low dimensional version of a domain block cannot provide a good match for the low dimensional version of the range block, then the same holds for the blocks at the real size. This is motivated by the following theorem.

Theorem 3: Let R and D be two image blocks of size $2^n \times 2^n$ and $2^{n+1} \times 2^{n+1}$, respectively. Then $\min_{s,o \in \mathbb{R}} \frac{1}{2^n} \|R - (sS^{(n+1)} D + o\mathbf{1})\| \geq \min_{s,o \in \mathbb{R}} \frac{1}{2^{n-1}} \|S^{(n)} R - (sS^{(n)} S^{(n+1)} D + o\mathbf{1})\|$.

Proof: See Proposition 7.2 in [7]. ■

Table 1 shows that searching in a few clusters with the dimension reduction algorithm yields PSNR values close to those obtained by inspecting all 72 clusters. In the experiments, the domain pool spacing was equal to eight pixels.

Since the centers are computed with a VQ algorithm, we may reasonably assume that our classification will also be successful if the training vectors are taken from a set of training images that do not include the test image. This approach eliminates the preprocessing time needed to construct adaptive centers [25]. Table 2 confirms that classification with fixed centers is an alternative to classification with adaptive centers. Here we encoded the 512×512 Boat image with Option 2, a single range size (4×4), and 4096 domain blocks of size 8×8. The 72 fixed centers were designed with the SOM from a set of 9 images that did not include the test image.

III. The hybrid scheme

A. Uniform partition

We first assume that the image is partitioned into square blocks of fixed size. Using the SOM with linear initialization [21], we design a small set of fixed centers from several training images as explained in Section II-C. It can be shown that these centers have zero mean [14]. Then we turn the centers into unit vectors $\mathbf{m}_1, \ldots, \mathbf{m}_{n_m}$ by normalization. Thus, we have $\phi(\mathbf{m}_p) = \mathbf{m}_p$ for all $p = 1, \ldots, n_m$. Since the design of the centers is independent of the test image, the centers can be made available at the decoder.

Suppose now that we want to encode a range block R of size $2^n \times 2^n$. For clarity, we explain a one-cluster search with Option 1. In a preprocessing step, we determine the cluster of each feature vector $\phi(C_j)$ corresponding to the codebook vector $C_j = S^{(n+1)}(\tau_{D_j} D_j),\ j \in \{1, \ldots, n_D\}$. Then, we determine the cluster center $\mathbf{m}_c = \Omega^1\{\phi(\tau_R R)\}$. Next, we consider approximating the range block by this center. Thus, we compute the least-squares error

$$\min_{(s,o)\in\mathbb{R}\times\mathbb{R}} \|\tau_R R - (s\,\mathbf{m}_c + o\,\mathbf{1})\|_2^2.$$

Note that Theorem 1 ensures that $\mathbf{m}_c$ is the center that can best approximate $\tau_R R$ in the least-squares sense. The values s and o associated to the solution are quantized, giving a scaling factor $\hat{s}(R)$, an offset $\hat{o}(R)$, and an error

$$d(R, \hat{s}(R), \hat{o}(R), \mathbf{m}_c, \tau_R^{-1}) = \|R - (\hat{s}(R)\,\tau_R^{-1}\mathbf{m}_c + \hat{o}(R)\,\mathbf{1})\|_2^2.$$

Now we test if the rms error given by this approximation is less than a given threshold α, that is, if

$$\frac{1}{2^n} d(R, \hat{s}(R), \hat{o}(R), \mathbf{m}_c, \tau_R^{-1})^{1/2} \leq \alpha. \tag{5}$$

If the inequality is satisfied, we do not consider the domain blocks, and the range block is encoded by the center as in oriented mean shape-gain VQ (OMSGVQ) [15], that is,

$$R \approx \hat{s}(R)\,\tau_R^{-1}\mathbf{m}_c + \hat{o}(R)\,\mathbf{1}.$$

If inequality (5) is not satisfied, then we compare the approximation given by the center to the approximation given by the best domain candidate in the set of domains associated to center $\mathbf{m}_c$. This is done by testing whether the inequality

$$d(R, \hat{s}(R), \hat{o}(R), \mathbf{m}_c, \tau_R^{-1})^{1/2} \leq (1+\epsilon) d(R, s(R), o(R), D(R), \tau_R^{-1}\tau_{D(R)})^{1/2} \tag{6}$$

holds. Here ϵ is a given small error bound and $D(R)$ is the domain block providing the best collage error among all domain blocks whose feature vectors $\phi(C_j)$ are in the cluster with center $\mathbf{m}_c$. When inequality (6) is satisfied, we know that the approximation given by the domain block cannot be much better than the approximation given by the center. Therefore we also encode the range block by the center $\mathbf{m}_c$. It is advantageous to still prefer the center to the domain block for two reasons. First, since there are fewer centers than domain blocks, the possible loss in quality will be compensated by a reduction in rate. Second, in fractal image compression, the reconstruction error at the decoder is typically larger than the collage error. This does not happen when we reconstruct the range block from the cluster center.

If neither condition (5) nor condition (6) is satisfied, then we encode the range block by the domain block $D(R)$ as in conventional fractal image compression.

In this way, the encoding of a range block R is specified by:

- One bit to indicate if a center or a domain block is used.
- The index c of the cluster center or the position of the domain $D(R)$.
- One of the two isometries τ_R^{-1} or $\tau_R^{-1}\tau_{D(R)}$.
- A scaling factor and an offset.

If the scaling factor is zero, the bit indicating the encoding method, the index of the center (respectively the position of the domain), and the isometry are not needed. As in Section 1, the search can be extended to neighboring clusters. By considering $\tau_{-R}(-R)$, Option 2 and Option 3 can also be used as well, yielding negative scaling factors.

The decoding of the hybrid scheme is by iteration starting from an arbitrary image as in a conventional fractal decoder. It is easy to see that if the range blocks encoded by domain blocks have scaling factors with absolute value less than one, then the decoding of the hybrid scheme is convergent, even when the scaling factors corresponding to the centers are arbitrary [14].

We now list the benefits of the hybrid scheme, when compared to the distance-based pure fractal scheme of Section 1.

1. The encoding is faster because the search for a domain block is started only if the center cannot provide an acceptable approximation (condition (6) is tested *only* if condition (5) is not fulfilled).

2. The compression ratio can be improved by a judicious choice of the ratio of the number of cluster centers to the number of domain blocks. If n_R is the number of range blocks and n_1

is the number of range blocks VQ encoded, then the hybrid scheme uses fewer bits than the pure fractal scheme if $n_1 > \frac{n_R}{p-k}$, where $p = \log_2 n_D$ and $k = \log_2 n_m$. (In the derivation of the inequality, one bit per range is included to specify the encoding method (fractal or VQ).)

3. The decoding is faster. Indeed, it is not necessary to recompute at each iteration the intensities of the pixels in the range blocks encoded by centers because these intensities are invariant after the first iteration. Moreover, we expect convergence in fewer iterations.

We now compare the performance of the distance-based fractal coder and the hybrid scheme. With Kohonen's SOM [22] we designed a set of 256 fixed centers corresponding to the nodes of a 16×16 square array. We generated the training vectors from the nine 512×512 grey-scale images Airplane, Eltoro, Baboon, Medicine, Crowd, Barbara, Bridge, Man, and Couple [14]. For the two schemes, we considered both positive and negative scaling factors (Option 2). Ranges had the fixed size (4×4), and the domain pool spacing was equal to 8. Thus we had 16384 range blocks and 4096 disjoint 8×8 domain blocks. We opted for a four-cluster search, which provides an acceptable trade-off between speed and fidelity. We spent respectively five and seven bits for the scaling factor and the offset. All coefficients were uniformly quantized. The maximum value of the scaling factor was set to 1 for fractal coding and to 250 for VQ coding. Table 3 shows coding results for the 512×512 Boat image. Here we used Lempel-Ziv coding (with GNU gzip) of the bit stream corresponding to the extra bit per range specifying the encoding method (VQ or fractal). The times were measured on an SGI Indigo2 running an MIPS R4400 150 MHz processor. In comparison, the distance-based fractal coder required 57 seconds for the encoding, had a compression ratio of 4.74:1 and provided a PSNR of 36.29 dB. Note that a full search (here a 256-cluster search) would increase the PSNR of the pure fractal scheme to 36.52 dB. On the other hand, encoding all range blocks with VQ centers yielded a PSNR of only 35.81 dB. Table 4 compares the convergence of the decoding of the two schemes when the iterations are started from a black image. Convergence occurred after five iterations for the hybrid scheme and nine iterations for the pure fractal scheme.

The experiments confirm that for a wide range of the thresholds α and ϵ, the hybrid scheme outperforms the pure fractal scheme: the compression ratio is higher, the PSNR is better, and both the encoding and the decoding are faster.

B. Quadtree partition

We now explain how to combine the hybrid scheme with Fisher's top-down quadtree scheme [10]. First, we design additional sets of VQ centers independently for each range size. Thus, for block size $2^n \times 2^n,\ n \in \{2, 3, \ldots, n_{max}\}$, we have n_m centers (the number of centers need not be the same for each size) $\mathbf{m}_1^n, \ldots, \mathbf{m}_{n_m}^n$, each of which having zero mean and norm one. To accelerate the clustering, we downsample the high dimensional centers ($n > 2$), the range blocks, and the domain blocks to the lowest dimension. Thus, the feature vectors of the ranges and domains are 16-dimensional.

The core of the quadtree algorithm for a q-cluster search is as follows. For simplicity, we discuss Option 1 only; the extension to Option 2 and Option 3 is straightforward. Let R^n be a square block of size $2^n \times 2^n,\ n \in \{3, \ldots, n_{max}\}$. We start with $n = n_{max}$. For some given rms thresholds α and t, if

$$\frac{1}{2^n} d(R^n, \hat{s}(R^n), \hat{o}(R^n), \mathbf{m}_c^n, \tau_{R^n}^{-1})^{1/2} \leq \alpha \tag{7}$$

then R^n is encoded with cluster center $\mathbf{m}_c^n$. Here $\mathbf{m}_c^n$ is the center of dimension $2^n \times 2^n$ that yields the smallest rms error among the q centers whose feature vectors $\phi(S^{(3)} \cdots S^{(n)} \mathbf{m}_{p_i}^n),\ i = 1, \ldots, q$ are the q nearest neighbors of the feature vector $\phi(S^{(3)} \cdots S^{(n)} \tau_{R^n} R^n)$. If condition (7) is not satisfied, then, letting $D(R^n)$ denote the best domain block among the domain blocks whose feature vectors are in the q selected clusters, if

$$\frac{1}{2^n} d(R^n, s(R^n), o(R^n), D(R^n), \tau_{R^n}^{-1} \tau_{D(R^n)})^{1/2} > t \tag{8}$$

then block R^n is not accepted, and it is partitioned into four smaller square blocks. In the other case, and if

$$d(R^n, \hat{s}(R^n), \hat{o}(R^n), \mathbf{m}_c^n, \tau_{R^n}^{-1})^{1/2} \leq (1+\epsilon) d(R^n, s(R^n), o(R^n), D(R^n), \tau_{R^n}^{-1} \tau_{D(R^n)})^{1/2},$$

then block R^n is encoded with cluster center $\mathbf{m}_c^n$, and otherwise, it is encoded with the domain block $D(R^n)$. When the lowest size is reached, the encoding proceeds as in the previous section.

Figure 3 shows the rate-distortion performance of our distance-based fractal scheme and our hybrid scheme for a three-level quadtree partition ($n_{max} = 4$). The fractal coder used a single set of 256 fixed centers of dimension 16. The hybrid scheme used three sets of 256 fixed centers. For each range size, the domain pool spacing was equal to eight pixels. Thus, we had 4096 disjoint domains of size 8×8, 3969 overlapping domains of size 16×16, and 3721 overlapping domains

of size 32×32. For both schemes we used Option 2 and a four-cluster search. At each level, we quantized the scaling factors and the offsets uniformly with five bits and seven bits, respectively. The parameters α and ϵ were set to 3 and 0.15, respectively. We varied the compression ratio by letting the rms threshold t take values between 22 and 1. Figure 4 shows the time as a function of compression ratio for the same series of tests. Figure 5 compares the perceptual quality of the decoded images at the same compression ratio.

IV. Optimal hybrid coding

The top-down quadtree partition technique of Fisher does not generate optimal quadtree encodings, that is, encodings that minimize the total collage error

$$\sum_{i=1}^{n_R} d(R_i, s(R_i), o(R_i), D(R_i), \tau(R_i))$$

over all quadtree encodings with the same or lower bit rate. Optimal quadtree encodings can be determined in reasonable time with the algorithm of Sullivan and Baker [44], which was introduced in a general image partition context for any type of quantizers that associate a rate and a distortion to the leaf nodes of a tree structure. To the best of our knowledge, the only application of this algorithm to fractal image compression is due to Lu [30]. In this section, we use the algorithm of Sullivan and Baker to find optimal quadtree encodings for the hybrid scheme.

Let us denote by d the total collage error and by r the total number of bits for a quadtree encoding in which the minimum range size is $2^2 \times 2^2$ and the maximum range size is $2^{n_{max}} \times 2^{n_{max}}$. Our objective is to find a quadtree encoding that minimizes d subject to a constraint on r. In the formulation of [8], this is a *cell problem* for the m_R square regions of maximum size $2^{n_{max}} \times 2^{n_{max}}$. Thus, optimal solutions can be found by solving independently for each $i \in \{1, \ldots, m_R\}$ the unconstrained problem

$$\min d_i + \lambda r_i$$

where $\lambda \geq 0$ is a Lagrange multiplier and d_i and r_i denote the total collage error and the total number of bits corresponding to a quadtree structure in a region of maximum size.

Let $n \in \{2, \ldots, n_{max}\}$ and let R^n be a $2^n \times 2^n$ block in a region of maximum size. We denote the number of bits and the error for representing R^n as a leaf node by $r^o(R^n)$ and $d^o(R^n)$, respectively. They correspond to the encoding of R^n in one of three ways: (1) with a domain

block (fractal coding); (2) with a cluster center (VQ); or (3) with only an offset. The optimal encoding as a leaf node is found by minimizing over all indices $k, m, j, p, k', m', j', p', m''$ the objective function

$$d(R^n) + \lambda r(R^n) \tag{9}$$

where

$$d(R^n) = \begin{cases} d(R^n, s_k, o_m, D_j, \tau_p) & \text{for fractal coding} \\ d(R^n, s_{k'}, o_{m'}, \mathbf{m}_{j'}, \tau_{p'}) & \text{for VQ} \\ \|R^n - o_{m''}\mathbf{1}\|_2^2 & \text{otherwise,} \end{cases}$$

and

$$r(R^n) = \begin{cases} \lceil \log_2 n_s \rceil + \lceil \log_2 n_o \rceil + \lceil \log_2 n_D \rceil + 3 + 2 & \text{for fractal coding} \\ \lceil \log_2 n_s \rceil + \lceil \log_2 n_o \rceil + \lceil \log_2 n_m \rceil + 3 + 1 & \text{for VQ} \\ \lceil \log_2 n_o \rceil + 2 & \text{otherwise.} \end{cases}$$

Here we use uniform quantization and fixed-length codes for the scaling factors and the offsets. The encoding method is specified by the following binary codes: 1 (VQ), 00 (fractal), and 01 (offset).

The algorithm of Sullivan and Baker works in a bottom-up fashion, merging subtrees that fulfill the following criterion. For $n > 2$ let the four subblocks of R^n be denoted by $R^{n,i}$, $i = 1, \ldots, 4$. Suppose one knows optimal quadtree structures for these four subblocks. Let the total number of bits and the total error of the subtrees corresponding to these optimal quadtree structures be denoted by $r^*(R^{n,i})$ and $d^*(R^{n,i})$, then the four subtrees are combined into a leaf when [44]

$$\Delta d \leq \lambda \Delta r \tag{10}$$

where

$$\Delta d = d^o(R^n) - \sum_{i=1}^{4} d^*(R^{n,i})$$

and

$$\Delta r = \sum_{i=1}^{4} r^*(R^{n,i}) - r^o(R^n).$$

The number of bits needed to represent the subtree for block R^n is

$$r^*(R^n) = \begin{cases} 1 + r^o(R^n) & \text{if } \Delta d \leq \lambda \Delta r \\ 1 + \sum_{i=1}^{4} r^*(R^{n,i}) & \text{otherwise,} \end{cases}$$

and the resulting error is

$$d^*(R^n) = \begin{cases} d^o(R^n) & \text{if } \Delta d \leq \lambda \Delta r \\ \sum_{i=1}^{4} d^*(R^{n,i}) & \text{otherwise.} \end{cases}$$

The algorithm starts at the lowest level of the quadtree ($n = 2$), finds the optimal one-level subtrees (which are leaves) according to (9), and merges sets of four subtrees into a single leaf when condition (10) is fulfilled. This gives optimal two-level subtrees. One proceeds similarly for the higher levels until one obtains the optimal $(n_{max} - 1)$-level subtree, which gives the optimal quadtree encoding for the region of maximum size.

In our implementation, we used variable-length codewords for the scaling factors of the VQ encoded range blocks. This is advantageous because the distribution of these scaling factors is nonuniform. For simplicity, we used uniform quantization with n_s levels followed by Huffmann coding. The Huffmann code was designed from a set of training images and made available at the decoder. In this way, if we denote the code length of the scaling factor index k' by $l(k')$, we have for VQ

$$r(R^n) = l(k') + \lceil \log_2 n_o \rceil + \lceil \log_2 n_m \rceil + 3 + 1.$$

No such attempt was made with the scaling factors of the fractal encoded range blocks since their distribution is almost uniform.

To accelerate the encoding, we used the techniques of Section II. For example, when minimizing $d(R^n) + \lambda r(R^n)$ in a one-cluster search with Option 1, we considered the domains corresponding to only one cluster. Also, for a cluster center or a domain block whose feature vector is in the selected cluster, we considered only one isometry and only the scaling factor-offset pair yielded by least-squares minimization.

Figure 6 illustrates, for the luminance (Y) 512×512 Lenna image, the rate-distortion (the distortion is given as before for the *decoded* images) performance of the following schemes:

1. Fisher's quadtree scheme ("Fractal quadtree") with a three-level quadtree and a domain pool spacing of eight pixels at each level. We searched 24 classes from a total of 72. Searching 24 classes is almost equal to full search [10]. It indicates the upper bound of the performance that could be achieved by conventional top-down quadtree fractal image compression.
2. The hybrid coder of Section III-B ("Hybrid") with the same domain pools as in "Fractal quadtree", 256 cluster centers per quadtree level, a four-cluster search with Option 2, $\alpha = 3$, and $\epsilon = 0.15$.

3. The hybrid coder based on Lagrange optimization ("Optimal hybrid"). We used the same domain pools and cluster centers as above, a four-cluster search with Option 2, and variable length codewords for the scaling factors of the VQ encoded blocks. We varied the compression ratio by letting the Lagrange multiplier λ take several values between 0 and 500.

4. A quadtree-based OMSGVQ scheme ("OMSGVQ") derived from the optimized hybrid scheme by eliminating the fractal coding alternative. In this way, the quadtree partition was obtained from the algorithm of Sullivan and Baker, and the scaling factors had variable length codewords. We considered the 20 nearest cluster centers ($q = 20$). Also, as for "Hybrid" and unlike "Optimal hybrid", we included the value 0 as a quantization level for the scaling factors. Therefore, no bits were needed to specify the encoding method (cluster center or only offset).

5. Fisher's HV coder with Optimization level 0 [11]. This is one of the best pure fractal coders in the spatial domain.

We improved further the hybrid scheme by exploiting the redundancy of the offset symbols. This was done by writing them as a list of eight bit symbols (although only seven bits were used for the quantization), which was entropy encoded with gzip. It provided significant savings although in view of (4) the offsets of the fractal encoded range blocks are not equal to the means of the range blocks as in MSGVQ encoding. Consistency was achieved by using Øien's orthogonalization [34] in the fractal coding part. In this case, (1) was replaced by

$$W_i(R_i) = s(R_i)OS^{(n+1)}\tau(R_i)D(R_i) + o(R_i)\mathbf{1}. \tag{11}$$

where O is the orthogonal projection on $\mathcal{C}^{\perp}$. With this setting, the optimal least-squares offset is the mean of the range block. We point out that decoding with orthogonalization is guaranteed to converge for arbitrary domain pools if the scaling factors are bounded by one in absolute value. Figure 7 compares reconstructions of the luminance (Y) 512×512 Lenna image at 0.25 bpp. The schemes used are the optimized hybrid coder with orthogonalization and entropy coding of the offsets and Fisher's quadtree coder. The encoding parameters are as for "Optimal hybrid" and "Fractal quadtree". At this bit rate, only about 16 % of the blocks in the quadtree partition were encoded with domain blocks (Figure 8).

V. Summary and Discussion

We have presented a coding scheme that efficiently combines fractal image compression and oriented mean shape-gain vector quantization. The perceptual quality of the decoded images

was satisfactory for low and medium compression ratios (up to 40:1). At high compression ratios, blocking artifacts were visible. Blocking artifacts, which are typical for encoding schemes based on disjoint blocks, can be reduced by postprocessing techniques (see [9], p. 59 for a simple technique).

Compared to conventional top-down quadtree-based fractal image compression [10], our hybrid scheme had faster encoding, faster decoding, and a better rate-distortion performance. When the quadtree partition and the encoding method for a block (fractal or OMSGVQ) were determined with Lagrange optimization, the PSNR was further improved by about 1 dB for a wide range of compression ratios. At medium to high compression ratios, the optimized hybrid scheme had a better rate-distortion performance than state-of-the-art pure fractal image compression schemes [11], [37]. But these schemes, which are based on more adaptive image partitions, produce a better visual quality at high compression ratios. However, they both need significantly more time for the encoding.

The rate-distortion improvement on OMSGVQ was important at low compression ratios, which indicates that the domain pool is competitive for small block sizes. On the other hand, both the encoding and the decoding of OMSGVQ are faster. Moreover, it is possible to improve the rate-distortion performance of OMSGVQ by using vector codebooks of larger size. However, this comes at the expense of a severe increase in both time and space complexity not only in the encoding phase but more dramatically in the training.

In fact, Lightstone *et al.* [27], [28] showed the potential of MSGVQ. For the 512×512 Lenna image, the rate-distortion results of their quadtree scheme [28] are superior to those reported here. However, their scheme requires much more complexity in codebook design and encoding time. In particular, the leaf codebooks corresponding to the various block sizes have to be designed with entropy constrained VQ ([12] p. 653) for each target rate. But our philosophy is not to oppose fractal image compression to vector quantization; our work showed that it is beneficial to combine the methods.

Even though it significantly improves fractal coding, our scheme is still not competitive with the state-of-the-art wavelet coders. For example, the codec of Said and Pearlman [38] is computationally simpler and provides better rate-distortion results (see Table 5). Note, however, that the PSNR is not an absolute measure of image quality, and that the artifacts produced by our scheme differ from those produced by wavelet coders. Also, unlike this codec, our scheme does

not generate an embedded code. This is a disadvantage because embedded codes allow scalability and progressive transmission.

Currently, we are studying further techniques to enhance the rate-distortion performance of our scheme. These include a better design of the cluster centers, a better entropy coding, and an iterative refinement of the image code [16].

Acknowledgments

We thank Bertram Ganz and Martin Müller for their outstanding programming work.

References

[1] Barnsley, M., *Fractals Everywhere,* Academic Press, San Diego, 1988.

[2] Barthel, K. U., Schüttemeyer, J., Voyé, T., Noll, P., *A new image coding technique unifying fractal and transform coding,* in: *Proc. ICIP-94 IEEE Int. Conf. Image Processing,* Vol. III, pp. 112–116, Austin, Texas, Nov. 1994.

[3] Bedford, T., Dekking, F. M., Keane, M. S., *Fractal image coding techniques and contraction operators,* Nieuw Arch. Wisk. (4) 10,3 (1992) 185–218.

[4] Bogdan, A., Meadows, H., E., *Kohonen neural network for image coding based on iteration transformation theory,* in: *Proc. SPIE Neural and Stochastic Methods in Image and Signal Processing,* Vol. 1766, pp. 425–436, 1992.

[5] Boss, R. D., Jacobs, E. W., *Archetype classification in an iterated transformation image compression algorithm,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), pp. 79–90, Springer-Verlag, New York, 1994.

[6] Davis, G. M., *A wavelet-based analysis of fractal image compression,* IEEE Trans. Image Processing 7,2 (1998) 141–154.

[7] Dekking, F. M., *Fractal image coding: some mathematical remarks on its limits and its prospects,* in: *Proc. NATO ASI Fractal Image Encoding and Analysis,* Trondheim, July 1995, Y. Fisher (ed.), pp. 117–131, Springer-Verlag, Berlin Heidelberg, 1998.

[8] Everett, H., *Generalized Lagrange multiplier method for solving problems of optimum allocation of resources,* Operations Research 11 (1963) 399–417.

[9] Fisher, Y. (ed.), *Fractal Image Compression — Theory and Application,* Springer-Verlag, New York, 1994.

[10] Fisher, Y., *Fractal image compression with quadtrees,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), pp. 55–77, Springer-Verlag, New York, 1994.

[11] Fisher, Y., Menlove, S., *Fractal encoding with HV partitions,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), pp. 119–136, Springer-Verlag, New York, 1994.

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

[13] Hamzaoui, R., *Codebook clustering by self-organizing maps for fractal image compression,* in: *NATO ASI*

Fractal Image Encoding and Analysis, Trondheim, July 1995, Fractals, Vol. 5, Supplementary Issue, pp. 27–38, April 1997.

[14] Hamzaoui, R., *Encoding and Decoding Complexity Reduction and VQ Aspects of Fractal Image Compression,* Doctoral Dissertation, University of Freiburg, 1997.

[15] Hamzaoui, R., Ganz, B., Saupe, D., *Quadtree based variable rate oriented mean shape-gain vector quantization,* in: *Proc. DCC'97 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Comp. Soc. Press, pp. 327–336, March 1997.

[16] Hamzaoui, R., Hartenstein, H., Saupe, D., *Local iterative improvement of fractal image codes,* accepted for publication in Image and Vision Computing.

[17] Hamzaoui, R., Müller, M., Saupe, D., *VQ-enhanced fractal image compression,* in: *Proc. ICIP-96 IEEE Int. Conf. Image Processing,* Volume 1, pp. 153–156, Lausanne, Sept. 1996.

[18] Hürtgen, B., Stiller, C., *Fast hierarchical codebook search for fractal coding of still images,* in: *EOS/SPIE Visual Communications and PACS for Medical Applications'93,* Volume 1977, pp. 397–408, Berlin, April 1993.

[19] Jacobs, E. W., Fisher, Y., Boss, R. D., *Image compression: A study of the iterated transform method,* Signal Processing 29 (1992) 251–263.

[20] Jacquin, A. E., *Image coding based on a fractal theory of iterated contractive image transformations,* IEEE Trans. Image Processing 1 (1992) 18–30.

[21] Kohonen, T., *Self-Organizing Maps,* Springer-Verlag, 1995.

[22] Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J., *SOM-PAK, The self-organizing map program package, Version 3.1,* Laboratory of Computer and Information Science, Espoo, April 1995.

[23] Lepsøy, S., *Attractor Image Compression: Fast Algorithms and Comparisons to Related Techniques,* PhD Thesis, The Norwegian Institute of Technology, Trondheim, Norway, June 1993.

[24] Lepsøy, S., Carlini, P., Øien, G. E., *On fractal image compression and vector quantization,* in: *Proc. NATO ASI Fractal Image Encoding and Analysis,* Trondheim, July 1995, Y. Fisher (ed.), pp. 21–47, Springer-Verlag, Berlin Heidelberg, 1998.

[25] Lepsøy, S., Øien, G. E., *Fast attractor image encoding by adaptive codebook clustering,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), pp. 177–197, Springer-Verlag, New York, 1994.

[26] Li, J., Kuo, C.-C. J., *Hybrid fractal-wavelet image compression based on a rate-distortion criterion,* in: *Proc. SPIE Visual Communications and Image Processing '97,* Vol. 3024, pp. 1014– 1025, 1997.

[27] Lightstone, M., Mitra, S. K., *Entropy-constrained, mean-gain-shape vector quantization for image compression,* in: *Proc. SPIE Visual Communication and Image Processing '94,* Vol. 2308, pp. 389–400, 1994.

[28] Lightstone, M., Rose, K., Mitra, S., *Locally optimal codebook design for quadtree-based vector quantization,* in: *Proc. ICASSP-1995 IEEE Int. Conf. Acoustics, Speech and Signal Proc.* pp. 2479–2482, 1995.

[29] Lin, H., Venetsanopoulos, A. N., *A pyramid algorithm for fast fractal image compression,* in: *Proc. ICIP-95 IEEE Int. Conf. Image Processing,* Volume III, pp. 596–599, Washington, D.C., Oct. 1995.

[30] Lu, N., *Fractal Imaging,* Academic Press, 1997.

[31] Lundheim, L. M., *Fractal Signal Modellings for Source Coding,* PhD Thesis, The Norwegian Institute of Technology, Trondheim, Sept. 1992.

[32] Monro, D. M., Dudbridge, F., *Fractal approximation of image blocks,* in: *Proc. ICASSP-1992 IEEE Int. Conf. Acoustics, Speech and Signal Processing,* Vol. 3, pp. 485–488, San Francisco, California, March 1992.

[33] Murakami, T., Asai, K., Yamazaki, E., *Vector quantizer of video signals,* Electronics Letters 7 (1982) 1005–1006.

[34] Øien, G. E., Lepsøy, S., *A class of fractal image coders with fast decoder convergence,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), pp. 153–175, Springer-Verlag, New York, 1994.

[35] Øien, G. E., Lepsøy, S., Ramstad, T., *Reducing the complexity of a fractal-based image coder,* in: *Proc. Vth European Signal Processing Conf. EUSIPCO'92,* 1992.

[36] Ramstad, T. A., Lepsøy, S., *Block-based attractor coding: Potential and comparison to vector quantization,* in: *Conf. Record 27th Asilomar Conference on Signals, Systems and Computers,* pp. 1504–1508, 1993.

[37] Ruhl, M., Hartenstein, H., Saupe, D., *Adaptive partitionings for fractal image compression,* in: *Proc. ICIP-97 IEEE Int. Conf. Image Processing,* Volume II, pp. 310–313, Santa Barbara, California, Oct. 1997.

[38] Said, A., Pearlman, W. A., *A new fast and efficient image codec based on set partitioning in hierarchical trees,* IEEE Trans. Circuits and Systems for Video Technology 6,3 (1996) 243–250.

[39] Saupe, D., *From classification to multi-dimensional keys,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), pp. 302–305, Springer-Verlag, New York, 1994.

[40] Saupe, D., *Fractal image compression via nearest neighbor search,* in: *Proc. NATO ASI Fractal Image Encoding and Analysis,* Trondheim, July 1995, Y. Fisher (ed.), pp. 95–115, Springer-Verlag, Berlin Heidelberg, 1998.

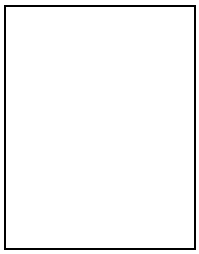
[41] Saupe, D., *Lean domain pools for fractal image compression,* SPIE Journal of Electronic Imaging 8,1 (1999) 98–103.

[42] Saupe, D., Hamzaoui, R., Hartenstein, H., *Fractal image compression– An introductory overview,* in: *Fractal Models for Image Synthesis, Compression and Analysis,* D. Saupe, J. Hart (eds.), ACM SIGGRAPH'96 Course Notes 27 , New Orleans, Louisiana, Aug. 1996.

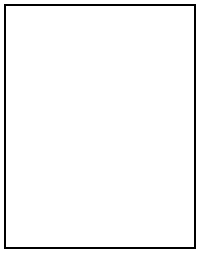
[43] Saupe, D., Hartenstein, H., *Lossless acceleration of fractal image compression by fast convolution,* in: *Proc. ICIP-96 IEEE Int. Conf. Image Processing,* Volume I, pp. 185–188, Lausanne, Sept. 1996.

[44] Sullivan, G. J., Baker, R. L., *Efficient quadtree coding of images and video,* IEEE Trans. Image Processing 3,3 (1994) 327–331.

[45] Woolley, S. J., Monro, D. M., *Optimum parameters for hybrid fractal image coding,* in: *Proc. ICASSP-1995 IEEE Int. Conf. Acoustics, Speech and Signal Proc.* Volume IV, pp. 2571–2574, Detroit, 1995.



Raouf Hamzaoui is a researcher at the University of Leipzig, Germany. He received the Maîtrise de mathématiques from the Faculty of Sciences of Tunis, Tunisia, in 1986, the M.Sc. degree in mathematics from the University of Montreal, Canada, in 1993, and the Dr. Rer. nat. degree from the Faculty of Applied Sciences of the University of Freiburg, Germany, in 1997. His research interests include digital image processing and numerical methods.



Dietmar Saupe received his Dr. rer. nat. and Habilitation degrees, both from the University of Bremen, in 1982 and 1993, respectively. He has served as visiting assistant professor of mathematics at the University of California at Santa Cruz (1985–1987), assistant professor at the University of Bremen (1987–1993), professor of computer science at the Albert-Ludwigs-University of Freiburg (1993–1998), and at the University of Leipzig (since 1998). His research has focused on image processing and coding, computer graphics, visualization, and dynamical systems. He is the co-author and editor of several books on fractals, e.g., Chaos and Fractals (Springer-Verlag, New York, 1992). He is a member of the IEEE Signal Processing Society, ACM SIGGRAPH, Eurographics, and others.

VI. List of Figures

Fig 1: Illustration of canonical classes. In the columns labeled B, blocks are shown with their quadrant means. There are $4! = 24$ possible rankings of these quadrant means. In each case, the next column labeled τ_B lists the isometry that specifies the canonical class of B, which is given in the last column. There are eight isometry operators, which are called

τ_1 = identity transformation,

τ_2 = counterclockwise rotation by 90 degrees,

τ_3 = counterclockwise rotation by 180 degrees,

τ_4 = counterclockwise rotation by 270 degrees,

τ_5 = reflection at the horizontal axis,

τ_6 = reflection at the vertical axis,

τ_7 = reflection at the diagonal,

τ_8 = reflection at the anti-diagonal

Fig 2: PSNR versus encoding time for the 512×512 luminance Lenna image. Dots are obtained by varying the number of classes (1,3,24,72) or clusters (1,2,3,4,8,24,72) searched.

Fig 3: PSNR versus compression ratio for the 512×512 Boat image.

Fig 4: Time versus compression ratio for the 512×512 Boat image.

Fig 5: (a) Hybrid scheme. Compression ratio = 32.06:1, PSNR = 29.55 dB. (b) Pure fractal scheme. Compression ratio = 31.96:1, PSNR = 28.48 dB.

Fig 6: Rate-distortion performance of the various schemes for the 512×512 Lenna image.

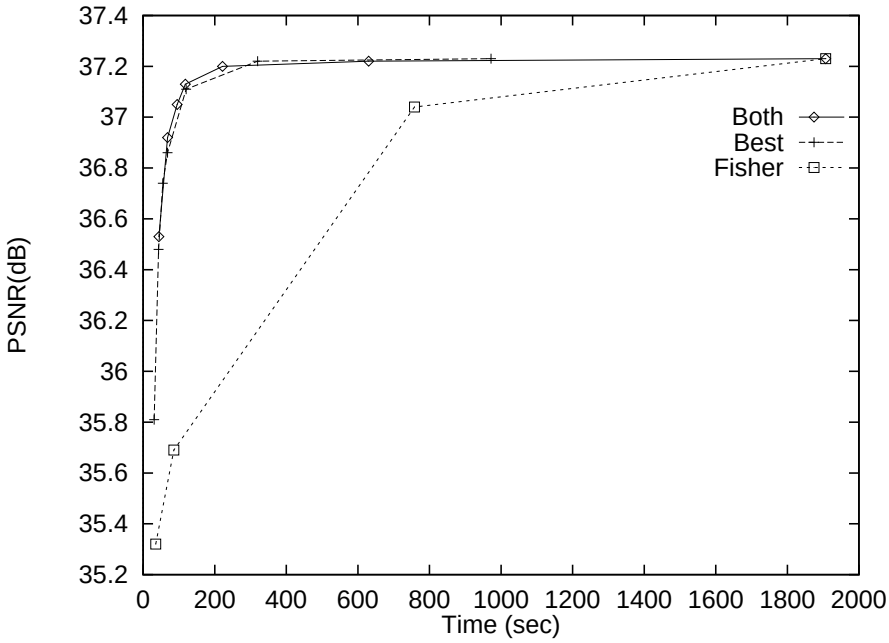
Fig 7: (a) Optimized hybrid scheme using orthogonalization and entropy coding of the offsets. Compression ratio: 32:1. PSNR = 32.50 dB. (b) Fisher's quadtree scheme for the same domain pools. Compression ratio: 32:1. PSNR = 30.46 dB.

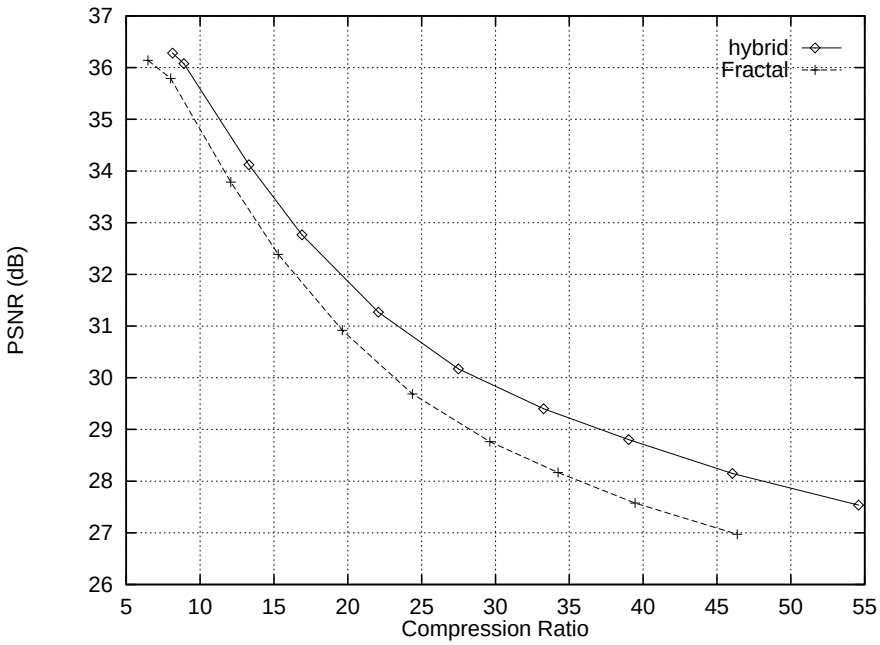
Fig 8: 512×512 Lenna image encoded with the optimized hybrid scheme at 0.25 bpp. There are 484 range blocks (shown in black) encoded with domain blocks, 1884 blocks encoded with cluster centers, and 654 blocks encoded with their mean value.

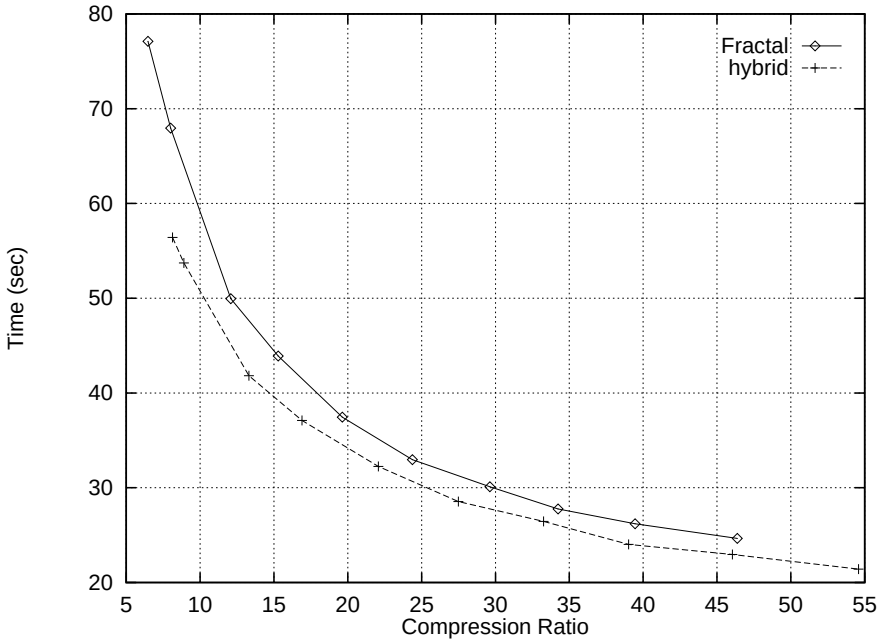
VII. List of Tables

B	τ_B	$\tau_B B$	class
	τ_3		1
	τ_5		2
	τ_5		3
	τ_7		1
	τ_2		2
	τ_2		3
	τ_3		2
	τ_5		1
	τ_4		3
	τ_7		2
	τ_2		1
	τ_6		3

B	τ_B	τ_B	class
	τ_3		3
	τ_4		1
	τ_4		2
	τ_7		3
	τ_6		2
	τ_6		1
	τ_8		3
	τ_8		2
	τ_8		1
	τ_1		3
	τ_1		2
	τ_1		1

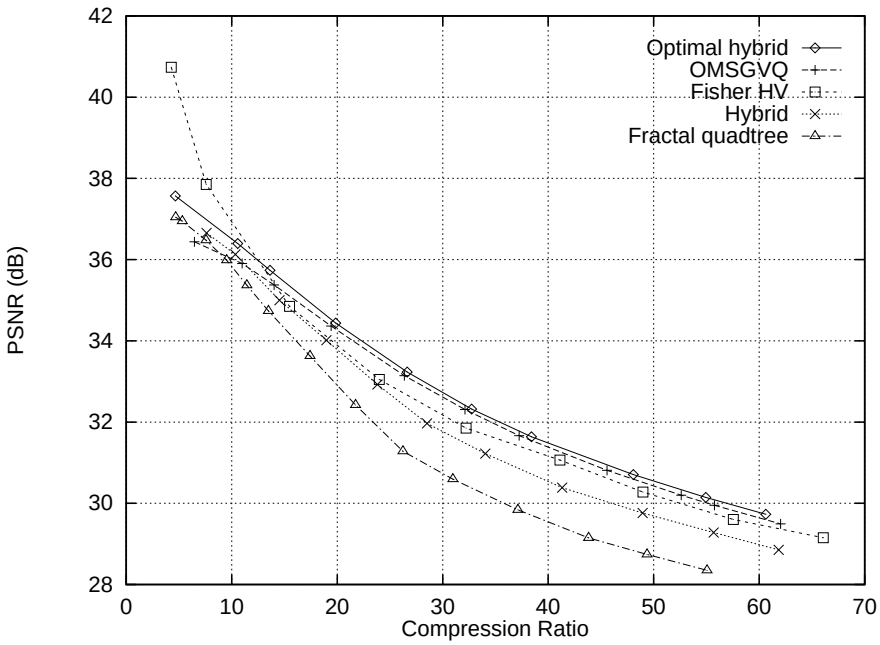

37.4
37.2
37
36.8
36.6
36.4
36.2
36
35.8
35.6
35.4
35.2
PSNR(dB)
0
200
400
600
800
1000
1200
1400
1600
1800
2000
Time (sec)
Both
Best
Fisher


PSNR (dB)
Compression Ratio
hybrid
Fractal
37
36
35
34
33
32
31
30
29
28
27
26
5
10
15
20
25
30
35
40
45
50
55


Time (sec)
Compression Ratio
Fractal
hybrid
80
70
60
50
40
30
20
5
10
15
20
25
30
35
40
45
50
55

(a)

(b)


PSNR (dB)
Compression Ratio
Optimal hybrid
OMSGVQ
Fisher HV
Hybrid
Fractal quadtree
42
40
38
36
34
32
30
28
0
10
20
30
40
50
60
70

(a)

(b)

Number of clusters	Block Size		
	8 × 8	16 × 16	32 × 32
1	31.04	26.46	22.31
2	31.31	26.61	22.48
3	31.38	26.73	22.54
4	31.42	26.77	22.57
8	31.46	26.82	22.62
72	31.48	26.83	22.61

Number of clusters searched	Time(sec)		PSNR(dB)	
	Adap	Fix	Adap	Fix
1	43	32	35.76	35.89
2	67	59	36.23	36.29
3	89	81	36.37	36.41
4	113	105	36.42	36.45

ϵ	α	n_1	Comp	PSNR	Time
0.1	2	9075	5.74	36.72	49
	3	12662	6.31	36.54	39
0.15	2	9947	5.80	36.64	49
	3	13313	6.37	36.47	39
0.2	2	10786	5.85	36.53	49
	3	13937	6.43	36.38	39
0.25	2	11523	5.91	36.44	49
	3	14441	6.49	36.29	39

Iteration k	Hybrid PSNR(dB)	Fractal PSNR(dB)
1	22.46	11.56
3	35.48	21.48
5	36.47	35.20
9	-	36.29

Image	PSNR at 0.25 bpp		PSNR at 0.5 bpp	
	Hybrid	SP	Hybrid	SP
512 × 512 Lenna	32.50	34.11	35.26	37.21
512 × 512 Goldhill	29.46	30.56	31.72	33.12