Fractal Image Compression
An Introductory Overview

Dietmar Saupe, Raouf Hamzaoui, Hannes Hartenstein[*]

Abstract

Fractal image compression is a new technique for encoding images compactly. It builds on local self-similarities within images. Image blocks are seen as rescaled and intensity transformed approximate copies of blocks found elsewhere in the image. This yields a self-referential description of image data, which — when decoded — shows a typical fractal structure. This paper provides an elementary introduction to this compression technique. We have chosen the similarity to a particular variant of vector quantization as the most direct approach to fractal image compression. We discuss the hierarchical quadtree scheme and vital complexity reduction methods. Furthermore, we survey some of the advanced concepts such as fast decoding, hybrid methods, and adaptive partitionings. We conclude with a list of relevant WEB resources including complete public domain C implementations of the method and a comprehensive list of up-to-date references.

1 Introduction

About ten to fifteen years ago fractal techniques were introduced in computer graphics for modeling natural phenomena. One of these new ideas came from a mathematical theory called *iterated function systems* (IFS). This theory had previously been developed in 1981 by John Hutchinson, however, without any technical applications in mind. It was Michael Barnsley and his research group from the Georgia Institute of Technology who first saw and realized the potential of iterated function systems for modeling of, e.g., clouds, trees, and leaves. Although other modeling techniques in computer graphics such as procedural modeling and L-systems are dominating the IFS approach, one of the visions of Barnsley — namely that of encoding entire images using IFS — turned into one of the most innovative techniques in the image compression field at present. Back in 1987 Barnsley and Sloan speculated [BaSl87] about very high compression ratios and announced that it was possible to transmit

[*]Address: Institut für Informatik, Universität Freiburg, Am Flughafen 17, 79110 Freiburg, Germany. Email: saupe,hamzaoui,hartenst@informatik.uni-freiburg.de. This paper is a revised version of Chapter 2 from *Fractal Models for Image Synthesis, Encoding and Analysis,* D. Saupe, J. Hart (eds.), SIGGRAPH '96 Course Notes XX, New Orleans, Aug. 1996.

such compressed image files at video rates over normal telephone lines. However, at that time nobody seemed to know exactly how to faithfully reproduce images at reasonable compression rates with IFS. What was the problem? The fractals that one can easily generate with an iterated function system are all of a particular type. They are images which can be seen as collages of deformed and intensity transformed copies of themselves. Thus, in an IFS encoding of a picture of a face one should see tiny little distorted copies of the face everywhere. This seemed not only unnatural but also technically infeasible. Then, in 1989, Arnaud Jacquin, one of the graduate students of Barnsley, realized a first automatic fractal encoding system in his dissertation [Jacq89c], leaving behind the rigid thinking in terms of global IFS mappings. This broke the ice for a new direction of research in image coding.

1.1 The fractal goldrush

The basic new idea in Jacquin's approach was very simple. An image should not be thought of as a collage of copies of the entire image, but of copies of smaller parts of it. For example, a part of a cloud certainly does not look like an entire landscape with clouds, but it doesn't seem so unlikely to find another section of some cloud or some other structure in the image that looks like the given cloud section. Thus, the general approach is to first subdivide the image into a partition — fixed size square blocks in the simplest case — and then to find a matching image portion for each part. This setup has been known as a local or partitioned iterated function system (PIFS). The development of Jacquin was like that of an engine. Around the engine he built a first vehicle, a workable image compression implementation. However, how to design such a vehicle in an optimal way remained to be investigated. And there were lots of open questions: for example, how should the image be segmented, where should one search for matching image portions, how should the intensity transformation be designed, and — most annoyingly — the algorithm as proposed and as given later in the form of a C code in the book of Barnsley and Hurd [BaHu93] was creepingly slow. Thus, methods for acceleration were urgently needed. This set the stage for a crowd of researchers mostly from mathematics, electrical engineering and computer science. Since its launching in 1994, the IEEE ICIP (International Conference on Image Processing), worldwide most prominent scientific image processing convention, regularly features a section on fractal image coding. We have tried to keep track of the publications dealing directly with this subject, see the bibliography in this chapter for a listing and our ftp site (Section 5) for many of the PostScript files. Figure 1 shows graphically the growth of the field in terms of the total number of publications.

One of the good things of standards is that people can build further research and applications on them, thereby accelerating scientific progress. This is just what happened after Yuval Fisher made his well written C code for an adaptive quadtree based fractal encoder available on the world wide web with a thorough theoretical and practical documentation in his book [Fish94a]. Then, in the summer of 1995, Fisher organized a NATO Advanced Research Institute on fractal methods for analysis and encoding of images, held in Trondheim, Norway. This was the first conference

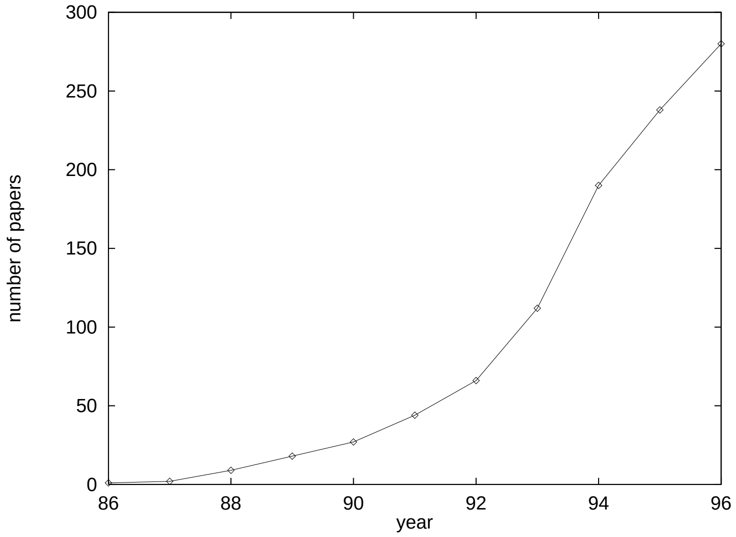

Figure 1: Total number of publications on fractal image compression.

devoted to the fractal approach. Another one followed, set up by the Georgia Institute of Technology and Iterated Systems, Inc., in Atlanta[1] in March 1996.

1.2 Four views of fractal image compression

As with any new methodology it is interesting to study interpretations from different perspectives. Several such views of fractal image compression have been considered:

1. **Iterated function systems (IFS).** Such systems are operators in metric spaces and were introduced in a mathematical paper by Hutchinson[2] in 1981, who showed that they have fractal subsets as attractors. This motivated Barnsley to search for an image compression system that models images as attractors of IFSs. Jacquin's solution of 1989 relies on a crucial modification of IFSs, namely that the mappings involved have domains that cover only part of the image. Thus, such IFSs were called *local* [BaHu93] or *partitioned* [Fish94a].

2. **Self vector quantization.** The basic fractal encoding is almost the same as a particular type of product code vector quantization (VQ), namely the so-called *mean-removed shape-gain vector quantization (MRSG-VQ)* [RaLe93]. In that approach an image block is approximated by the sum of a DC component and a scaled copy of an image block taken from the VQ codebook. Fractal encoding differs from MRSG-VQ because the codebook is not explicitly available at the decoder but rather given implicitly in a self-referential manner.

3. **Self-quantized wavelet subtrees.** Recently it has been noticed by Davis [Davi95a] and others that in some cases the fractal encoding is equal to a certain

[1] The First Annual Leadership Conference on Multimedia Imaging Technology and Applications.

[2] J. Hutchinson, *Fractals and Self-Similarity,* Indiana University Journal of Mathematics, vol. 30, pp. 713–740, 1981.

type of wavelet transform coding. The idea is to organize the (Haar) wavelet coefficients in a tree and to approximate subtrees by scaled copies of other subtrees closer to the root of the wavelet tree. See also Section 4.4.

4. **Convolution transform coding.** Also recently, it has been observed [Saup96b] that the operations carried out when searching a matching image region for a given one essentially are equivalent to a convolution operation. Only one of the convolution coefficients is selected for the fractal code. This establishes a close relation to common transform coding.

Each of these views of fractal encoding has led to a better understanding of the subject and inspired new research. For example, the similarities to VQ had already been studied by Jacquin [Jacq93] who, in fact, had imported useful classification methods, developed for VQ, to his fractal encoder. Moreover, the analogy to transform coding provides a new lossless technique for accelerating fractal encoding making use of the fast convolution transform, carried out in the frequency domain. The relationship to wavelets opens up interesting possibilities for hybrid codes which may hold the strongest prospects for the best rate-distortion curves available with fractal techniques.

In the following we will adopt the view point of self vector quantization rather than the traditional approach by iterated function systems. It is more straightforward since it is discrete by nature.

1.3 Vector quantization

Vector quantization (VQ) is a generalization of scalar quantization. In scalar quantization individual real or integer numbers are coded by an index listed in a fixed table of quantization values. For example, rounding to the nearest integer can be considered as a simple form of scalar quantization. In computer graphics quantization is associated mostly with undesirable artifacts, also known as aliasing. Geometric primitives such as lines and polygons need to be represented in terms of intensity values sampled on a regular discrete grid of pixels which necessarily leads to these artifacts. The art of vector quantization addresses the general problem of minimizing the errors associated with any quantization. Thus, the question is,

how to quantize if you must.

And clearly there are cases where quantization is a “must.” For example, consider displaying a true color image using graphics hardware supporting only a color lookup table of, say, 256 colors. Such a configuration is common in PCs and workstations as well. There are two problems to be distinguished here.

1. **The quantizer.** Given a table of 256 color vectors $C = \{y_1, \ldots, y_{256}\}, y_i \in [0,1]^3, i = 0, \ldots, 255$, called *codebook,* and a pixel color vector $x \in [0,1]^3$, find the index $i \in \{1, \ldots, 256\}$ such that the codebook vector y_i approximates the given color x best. In other words, define an optimal partitioning of the color

space $[0,1]^3$ into regions $R_1, \ldots, R_{256}$ so that the quantization is declared by the mapping $Q : [0,1]^3 \to C$, where $Q(x) = y_i$ if and only if $x \in R_i$.

2. **The codebook design.** Given an ensemble or a category of images, design an optimal size 256 codebook for color quantization. In other words, the color look-up table needs to be defined so that the quantization process described above yields the least color distortion for an image on average.

In order to solve these problems one needs a measure of how well a pixel color is approximated by an entry from the codebook. Such functions are called *distortion measures* in quantization theory. Most commonly, the squared Euclidean distance is used for this purpose,

$$d(x,y) = \sum_{k=1}^{n} (x^{(k)} - y^{(k)})^2$$

where n denotes the dimension of the quantizer (which is $n = 3$ for color quantization) and $x^{(k)}$ is the k-th component of the vector x. With this distortion measure the quantizer is a so-called *nearest-neighbor-quantizer,* because the codebook vector with minimal distortion for a given query vector x is the one that minimizes the Euclidean distance to x. The codebook design problem is very hard; only suboptimal solutions are obtainable in practice.

Optimal design of color look-up tables has been an issue in computer graphics research.[3] Interestingly, also fractal space-filling curves have been used in this context.[4]

To discuss the codebook design let us assume the more general case of quantizing n-dimensional data vectors $x_1, \ldots, x_M \in \mathbf{R}^n$ using a codebook $C = \{y_1, \ldots, y_N\}$ and the squared Euclidean distortion measure.[5] There are two optimality conditions, that need to be satisfied in an optimal quantizer.

1. **Nearest neighbor condition.** Given a codebook C, the optimal partition cells R_i satisfy

$$R_i \subset \{x \in \mathbf{R}^n \mid d(x, y_i) \le d(x, y_j) \text{ for all } j\}$$

Thus, the distortion for a given vector x is $d(x, Q(x)) = \min_{y_j \in C} d(x, y_j)$.

2. **Centroid condition.** Given a partition $R_1, \ldots, R_N$ the optimal codebook $C = \{y_1, \ldots, y_N\}$ consists of the centroids of the regions:

$$y_i = \text{cent}(R_i) := \frac{\sum_{j=1}^{M} \mathbf{1}_{R_i}(x_j) x_j}{\sum_{j=1}^{M} \mathbf{1}_{R_i}(x_j)}$$

---

[3] P. Heckbert, *Color image quantization for frame buffer display,* ACM Trans. Comput. Gr. 16,3 (1982) 297–307. S. J. Wan, S. K. M. Wong, P. Prusinkiewicz, *An algorithm for multidimensional data clustering,* ACM Trans. on Math. Software 14,2 (1988) 153–162.

[4] R. J. Stevens, A. F. Lehar, F. H. Preston, *Manipulation and presentation of multidimensional image data using the Peano scan,* IEEE Trans. on Pattern Analysis and Machine Intelligence PAMI-5,5 (1983) 520–526.

[5] The hasty reader may skip these details and go on to page 107.

Here $\mathbf{1}_R(x)$ denotes the indicator function, i.e., its value is 1 if $x \in R$ and 0 otherwise. In other words, the codebook vectors are the averaged vectors from the corresponding regions.

Given a training sequence of vectors and a codebook C_m an improved codebook C_{m+1} can be generated using the two optimality conditions. This is called a *generalized Lloyd iteration.*

1. **Step 1.** Given a codebook $C_m = \{y_1^m, \ldots, y_N^m\}$, partition the training set $x_1, \ldots, x_M \in \mathbf{R}^n$ into subsets R_i^m using the nearest neighbor condition, i.e.,

$$R_i^m = \{x_k \mid d(x_k, y_i^m) \leq d(x_k, y_j^m) \text{ for all } j\}$$

with a suitable tie-breaking rule.

2. **Step 2.** Using the centroid condition compute the centroids $\mathrm{cent}(R_i)$ and define the codebook $C_{m+1} = \{y_i^{m+1}\mathrm{cent}(R_i^m) \mid i = 1, \ldots, N\}$.

This procedure can be iterated. Starting out with an initial codebook C_0 with nearest-neighbor quantizer Q_0 the total distortion after the m-th such iteration is

$$D_m = \sum_{k=1}^{M} d(x_k, Q_m(x_k)) = \sum_{k=1}^{M} \min_{y_j \in C_m} d(x_k, y_j^m).$$

It follows from the optimality conditions that the sequence of total distortions $D_0, D_1, \ldots$ is decreasing. Since the distortions are bounded from below by 0 the sequence must converge to a limit. Moreover, since there are only finitely many different partitions of a finite set of training vectors it can be shown that the limit is achieved after a *finite* number of iterations. However, in practice a termination criterion

$$\frac{D_m - D_{m+1}}{D_m} \leq \epsilon$$

with a user specified tolerance ϵ is adopted. The mathematical theory for vector quantization and its many variants can be found, e.g., in the book of Gersho and Gray,[6] from which we have borrowed some of the notation as given in this section.

The method is straightforward to apply to grey scale images. Images are partitioned into blocks of fixed size, e.g., 4×4 pixels. These blocks are scanned row by row yielding vectors of dimension $n = 16$. Several images may be used to generate training vectors, an initial codebook is selected (there are several sophisticated algorithms for this), and generalized Lloyd iterations are performed until the convergence criterion is fulfilled. The resulting codebook can be used to encode a given image which normally is assumed to be different from the training images. Figure 2 shows as an example a small section of a reconstructed image along with the original.[7]

[6] A. Gersho, R. Gray, *Vector Quantization and Signal Compression,* Kluwer Academic Publishers, Boston, 1991.

[7] See http://isdl.ee.washington.edu/COMPRESSION/homepage.html for a C-code implementation of the full search vector quantization scheme described here. This package was also used to generate the codebooks and encodings.

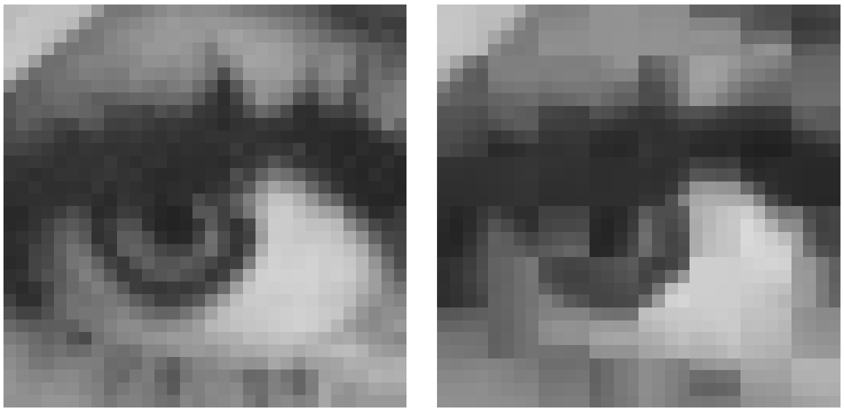

Figure 2: Standard vector quantization of a section of the Lenna test image (original on left) using a codebook of 512 blocks of size 4×4. The attainable PSNR with this approach is about 31.2 dB for the Lenna test image.

When evaluating the resulting VQ encoding of an image two closely related quantities need to be observed. The first one is the *compression ratio.* It is defined as the ratio of the file size of the original image representation to that of the encoded version of it. For the example in the figure the compression ratio is easily calculated. The original file size is 8×512^2 bits, since it is an image of 512 by 512 pixels, each one carrying an 8 bit intensity value. The compressed version requires $9 \times (512/4)^2$ bits, since 9 bits suffice to store the index from the set $\{1, \ldots, 512\}$ and there are $(512/4)^2$ image blocks to be encoded. Thus, the compression ratio is about 14.2. The other quantity measures quality. It is an open problem defining the visual quality of an image approximation in a mathematically expressible way. Thus, most authors use the simple root-mean-square (rms) error or *peak-to-peak signal-to-noise ratio* (PSNR). For 8-bit gray scale images the PSNR is defined as

$$PSNR = 10 \log_{10} \frac{255^2}{\text{ms-error}} = 10 \log_{10} \frac{255^2}{\frac{1}{\#\text{ pixels}} \sum_{i,j} (\hat{p}_{i,j} - p_{i,j})^2}$$

where $p_{i,j}$ and $\hat{p}_{i,j}$ denote the pixel intensities in the original and in the approximation respectively. The PSNR expresses the ratio of the maximal signal power to that of the error, also called *quantization noise.* It is measured in units of decibel (1 dB = one tenth of a logarithmic unit).

In a variable rate encoder different compression ratios can be realized which lead to encodings of varying quality. Thus, in order to compare different encoders or different parameter settings in one encoder one needs to record several points given by (compression ratio, PSNR) in a graph for both methods. When connecting some of these points we get curves that are called *rate-distortion curves.* The higher the curve in the graph the better the encoder. Sometimes rate-distortion curves specify the *bitrate* in place of the compression ratio. The bitrate simply is the file size in bits divided by the number of pixels.

1.4 Mean-removed shape-gain vector quantization

The standard VQ approach may produce the best possible rate-distortion curves, however, this can be achieved only with larger block sizes. But very large codebooks are impractical for two reasons. Firstly, the storage requirements for the codebook vectors at encoder as well as at the decoder are a hindrance. Secondly, the codebook design algorithm breaks down because of the huge time-complexity involved. For example, at a fixed bitrate of 1 bit/pixel (i.e., at fixed compression ratio 8) the codebook size is 2^d, where d denotes the block size in pixels. Clearly, already for 8×8 blocks such large codebooks are much beyond the capabilities of computers today.

For this reason there exist many variations of VQ in which codebooks with certain structures are used which makes them computable but suboptimal, i.e., this reduces the performance of the approach in terms of quality. One of the methods used is called *product code vector quantization*, and a particular variant of it is considered here, namely *mean-removed shape-gain VQ* (MRSG-VQ). As the name suggests, a vector $R \in \mathbf{R}^n$ to be encoded is written as

$$R = s \cdot D + o \cdot \mathbf{1}$$

where $\mathbf{1} = (1, \ldots, 1)^T \in \mathbf{R}^n$ and s, o are scalars. $D = (d_1, \ldots, d_n)^T$ is a zero-mean and unit-variance shape-vector, i.e.,

$$\sum_{i=1}^{n} d_i = 0, \qquad \sum_{i=1}^{n} d_i^2 = 1$$

With two scalar codebooks for s and o and a vector codebook of shape vectors the quantized form of the input vector R is

$$R \approx s_{ind_s(R)} D_{ind_D(R)} + o_{ind_o(R)} \mathbf{1}$$

where $ind_s(R)$, $ind_D(R)$, and $ind_o(R)$ are appropriate indices generated by the quantizer. Roughly, the scheme separately encodes the mean, the standard deviation, and the shape of a given vector. In effect, by considering all three codebooks simultaneously, a very large joint codebook is obtained. For example, if the codebook sizes for s, o, and D are 32, 128, and 4096 respectively, we get a total of 2^{24} vectors that can be represented exactly.

We do not give details for the codebook design in this case.[8] Instead we present in Figure 3 the result of a particular design of 64 blocks of size 4×4 pixels. Using a given shape block from the codebook different blocks can be generated using different gains s and means o. Figure 4 shows these blocks for one example shape block from Figure 3. Using this approach blocks from an image can be approximated by an encoder. The decoder having access to the codebook and receiving the code consisting of the indices for the scalar gains and means and the indices for the shape vectors reassembles the approximation as shown in Figure 5. Finally, Figure 6 shows the performance that can be attained by this approach when using different sizes of the shape codebook.

[8] See, e.g., the book of Gersho and Gray or the article M. J. Sabin, R. M. Gray, *Product code vector quantizers for waveform and voice coding,* IEEE Trans. Acoust. Speech Signal Process. 32 (1984) 474–488.

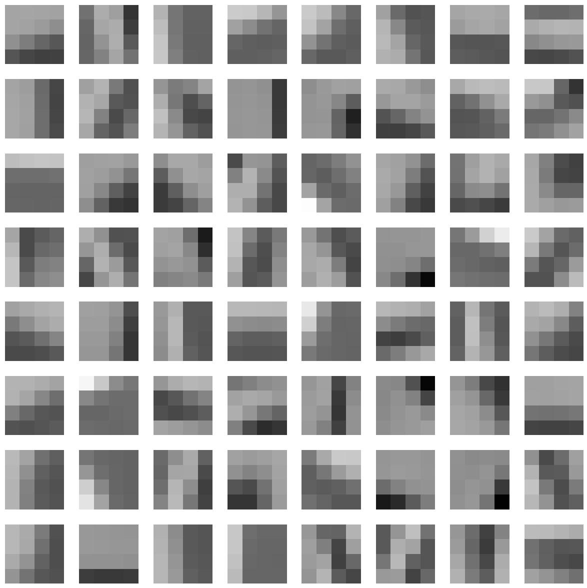

Figure 3: Visualization of a shape codebook in mean-removed shape-gain vector quantization. It consists of 64 blocks of size 4×4 pixels with zero mean and unit variance. For the display the vector components have been multiplied with a gain of 180 and are added to the mean of 127.

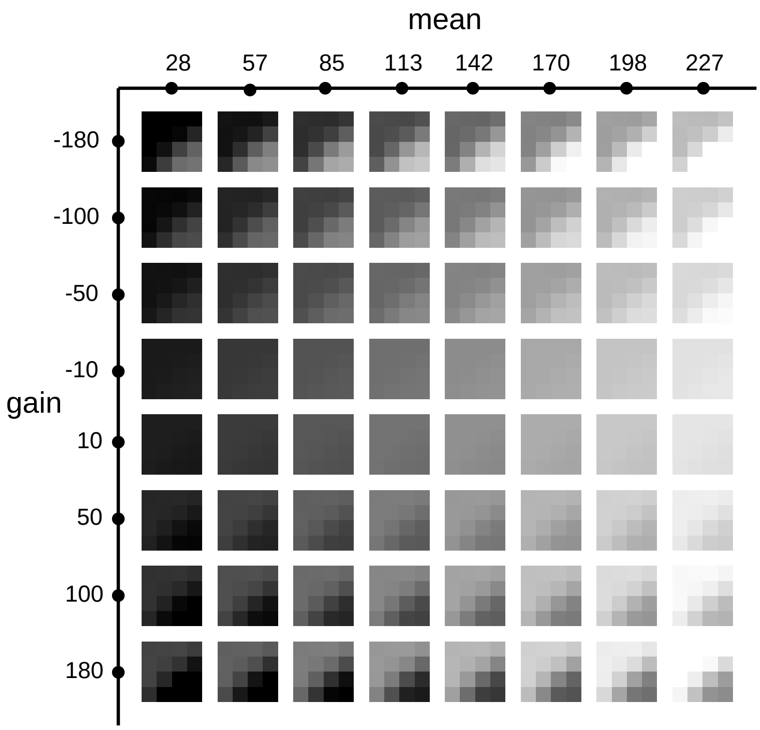

Figure 4: Visualization of the MRSG-VQ product code blocks for the second shape block in the third row of Figure 3. The scalar codebooks for the gain and mean contain only eight values as shown in the graph.

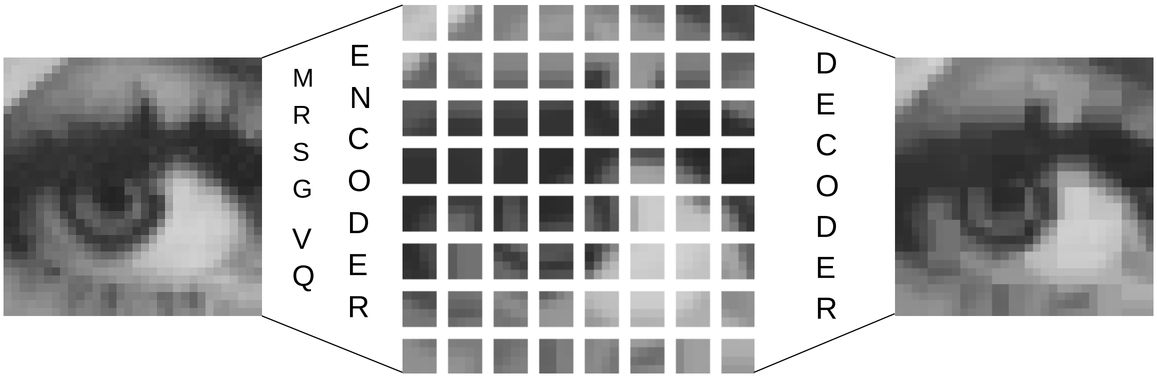

Figure 5: Visualization of the MRSG-VQ codec. The test image Lenna is encoded using the shape codebook of size 64 shown in Figure 3. The uniform scalar quantizers for the gain and the offset use 32 and 128 levels respectively. The picture shows only an enlarged section (32 by 32 pixels) of the entire image in order to better see the pixels and the image blocks. The PSNR for the entire image approximation is 34.6 dB

1.5 MRSG self-VQ and the fractal baseline encoder

The basic form of fractal image compression is very similar to mean-removed shape-gain *VQ*. The difference between the two is that in *VQ* a fixed, trained codebook is used, while in fractal image encoding an image adaptive codebook is used, which consists of blocks taken from the original image. This may seem like a contradiction since it is just the job of the decoder to recover the original and, thus, the decoder cannot have access to the codebook. So, if the image is encoded blockwise as scaled copies of other image blocks plus constant gray blocks, then how can the decoder reconstruct the original?

Let us give an example where for simplicity we encode just a single real number instead of an image, say $\pi = 3.1415\ldots$ We assume that the codebooks for the scale and offset are

$$s \in \{0, 0.25, 0.5, 0.75\}, \qquad o \in \{0.0, 0.4, 0.8, 1.2, 1.6, 2.0\}.$$

The "shape codebook" consists of just one number, namely π itself. Table 1 lists all the possible numbers $s \cdot \pi + o$ where s and o are from the given codebooks.

scale	offset o					
s	0.00	0.40	0.80	1.20	1.60	2.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.25	0.79	1.19	1.59	1.99	2.39	2.79
0.50	1.57	1.97	2.37	2.77	3.17	3.57
0.75	2.36	2.76	3.16	3.56	3.96	4.36

Table 1: This table lists all numbers rounded to two decimals that can be represented by $s\pi + o$ when using the scalar codebooks for s and o as shown.

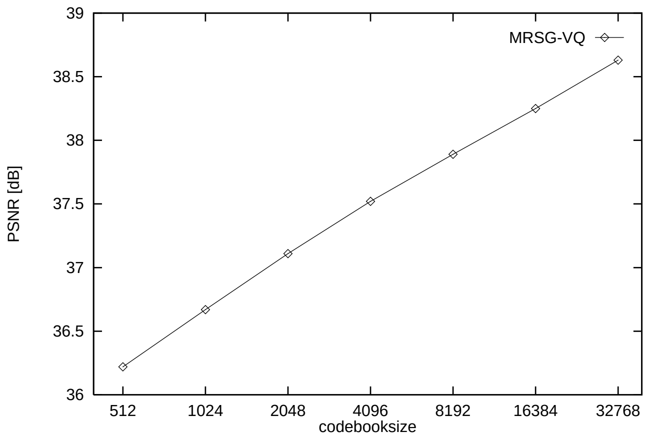

Figure 6: Codebook size vs. distortion curve for *MRSG-VQ* using blocks of size 4×4 and shape codebooks of various sizes. The uniform scalar quantizers for the gain and the mean use 32 and 128 levels respectively. The test image is 512×512 Lenna.

We see that $s = 0.75$ and $o = 0.8$ gives the best approximation of π, namely

$$s \cdot \pi + o = 0.75 \cdot \pi + 0.8 = 3.1561\ldots$$

Thus, the encoder may pass the following information to the decoder:

The original number is about 0.75 *times itself plus* 0.8.

The error in this approximation is not specified, and, of course, there are many numbers that satisfy this description. Lacking any other information, the decoder could determine anyone of them. However, one of them is a unique, special number x, namely the one that is *exactly* 0.75 times itself plus 0.8, i.e.,

$$x = 0.75 \cdot x + 0.8.$$

Solving this equation we obtain $x = 3.2$, which should be taken as the decoded number. Thus, the encoder approximates the input number using the codebooks for s and o and the original number, while the decoder cannot recover this approximation, but instead produces the unique number, which is characterized by the property, that the encoder could approximate it with no error when using the given coefficients.

The resulting equation $x = 0.75x + 0.8$ is easy to solve. But when we deal with images containing thousands of numbers (pixel intensities) the corresponding system of equations that arises in fractal image compression is so large, that it cannot be solved directly but only by iteration. This can also be demonstrated with our little toy example. If we define an operator $T : \mathbf{R} \to \mathbf{R}$ by $Tx = 0.75x + 0.8$, then the encoder statement is simply $\pi \approx T\pi$, and we have to solve the fixed point equation $x = Tx$. Given an arbitrary initial guess x_0 we iteratively apply T, which yields

$$x_1 = Tx_0, \quad x_2 = Tx_1, \quad x_3 = Tx_2, \ldots$$

Here we get with $x_0 = 0$, e.g.,

$$\begin{aligned} x_1 &= 0.8 \\ x_2 &= 0.75 \cdot 0.8 + 0.8 = 1.4 \\ x_3 &= 0.75 \cdot 1.4 + 0.8 = 1.85 \\ x_4 &= 2.1875 \end{aligned}$$

and then $x_{10} = 3.06\ldots, x_{20} = 3.192\ldots$, and $x_{30} = 3.1995\ldots$ This sequence of iterates converges to the fixed point 3.2, which is also called the *attractor* for the operator T. This is not a coincidence. Whenever the scaling factor is less than 1 in absolute value, $|s| < 1$, convergence to the fixed point ensues. The analogous property holds for the case of images considered next.

The encoder proceeds in a similar fashion as in MRSG-VQ. Here, the shape codebook is not given a priori as the result of some training and design process. Instead the shape codebook consists of image blocks extracted from the original image that has to be encoded. This implies that these blocks are not normalized to zero mean and unit variance. This "fractal" codebook is highly adaptive. Each image has its own codebook. Here is an example.

Example codebook. Suppose that the image is segmented into blocks of size 4×4 pixels, called *ranges.* Each range block R must be approximated as $R \approx sD + o\mathbf{1}$, where D is a 4×4 block from the shape codebook. Consider any *domain* block of size 8×8 in the image. Then shrink the block by pixel averaging to the desired size of 4×4 pixels. All such blocks are added to the shape codebook. For an image of size 512×512 this process yields a huge codebook with $(512 - 7)^2 = 255025$ blocks. In order to reduce the number of blocks to a more manageable size one may consider only those domain blocks that have their upper left corner pixel on a regular square grid with a spacing $l > 1$. For example, with $l = 8$ we would obtain a set of 4096 adjacent domain blocks, which is often used in practice.

The encoder has to solve the following problem. For each range block the best approximation $R \approx sD + o\mathbf{1}$ needs to be found. In fractal encoding the coefficients s and o are called *scaling* and *offset.* To obtain optimal s, o, and D, a scan of all codebook blocks D should be performed. For each codebook block D the best coefficients s and o need to be determined. In the above one dimensional example for π we computed a table of all possibilities and chose the best one. In principle this can also be done for vectors or image blocks as required here. However, for all but the smallest scalar codebooks for s and o this is computationally infeasible. It takes too long. Fortunately, there exists a shortcut. If we work with the Euclidean norm when making the selection of the best coefficients, i.e., when minimizing

$$E(D, R) = \min_{s,o} ||R - (sD + o\mathbf{1})||$$

we can use the well known method of least squares to find the optimal coefficients directly as follows.

Given the two blocks R and D with n pixel intensities, $r_1, \ldots, r_n$ and $d_1, \ldots, d_n$

we have to minimize the quantity

$$\sum_{i=1}^{n}(s \cdot d_i + o - r_i)^2.$$

The best coefficients s and o are

$$s = \frac{n\left(\sum_{i=1}^{n} d_i r_i\right) - \left(\sum_{i=1}^{n} d_i\right)\left(\sum_{i=1}^{n} r_i\right)}{n\sum_{i=1}^{n} d_i^2 - \left(\sum_{i=1}^{n} d_i\right)^2} \tag{1}$$

and

$$o = \frac{1}{n}\left(\sum_{i=1}^{n} r_i - s\sum_{i=1}^{n} d_i\right). \tag{2}$$

With s and o given the square error is

$$E(D,R)^2 = \frac{1}{n}\left[\sum_{i=1}^{n} r_i^2 + s\left(s\sum_{i=1}^{n} d_i^2 - 2\sum_{i=1}^{n} d_i r_i + 2o\sum_{i=1}^{n} d_i\right) + o\left(on - 2\sum_{i=1}^{n} r_i\right)\right].$$

If the denominator in equation (1) is zero, then $s = 0$ and $o = \sum_{i=1}^{n} r_i/n$.

This procedure yields two real numbers s and o. For the encoding we can only use the quantized values from the scalar codebooks. Usually, one employs uniform scalar quantization amounting to a rounding operation.

In summary the baseline fractal encoder with fixed block size operates in the following steps.

1. *Image segmentation.* Segment the given image using a fixed block size, e.g., 4×4. The resulting blocks are called ranges R_i.

2. *Domain pool and shape codebook.* By stepping through the image with a step size of l pixels horizontally and vertically create a list of domain blocks from the image, which are twice the range size. By averaging four pixels each shrink the domain blocks to match the size of the ranges. This produces the codebook of blocks D_i.

3. *The search.* For each range block R an optimal approximation $R \approx sD + o\mathbf{1}$ is computed in the following steps:

   (a) For each codebook block D_i compute an optimal approximation $R \approx sD_i + o\mathbf{1}$ in three steps:

      i. Perform the least squares optimization using formulas (1) and (2), yielding a real coefficient s and an offset o.

      ii. Quantize the coefficients using, e.g., a uniform quantizer.

      iii. Using the quantized coefficients s and o compute the error $E(R, D_i)$.

   (b) Among all codebook blocks D_i find the block D_k with minimal error $E(R, D_k) = \min_i E(R, D_i)$.

(c) Output the code for the current range block consisting of indices for the quantized coefficients s and o and the index k identifying the optimal codebook block D_k.

As already mentioned the output code of this baseline fractal encoder is not a code with which a decoder can directly recover an approximation of the original. Instead we have a description of an operator. Similar to the code for π we now have the following result of the encoder:

> *Given the original image along with its partitioning in square ranges replace each range R by the corresponding block $sD + o\mathbf{1}$ as specified by the code. The resulting image, called* collage*, is an approximation of the original.*

Thus, the code is nothing but the prescription of an image operator T. Given *any* image g_0 one can carry out the operations given in the code, arriving at another image, Tg_0. When applying T to the original image f, we obtain Tf, the collage, and the encoder result can be stated as $f \approx Tf$. The error of this approximation is called the *collage error*. It is defined as the sum of the square errors $E(D,R)^2$ taken over all ranges R of the image partition. From this sum the corresponding root-mean-square (rms) error or peak-to peak signal-to-noise ratio (PSNR) can be calculated.

Just as in the case of π and lacking any other information, the best job that the decoder can do is to compute the fixed point $g = Tg$. This is the image which, encoded by T gives a perfect encoding, i.e., one for which the collage error vanishes. In practice the decoder computes the fixed point by iteration of T. Thus, starting with an arbitrary initial image g_0, we get

$$g_1 = Tg_0, \quad g_2 = Tg_1, \quad g_3 = Tg_2, \ldots$$

and this sequence of images should converge to an *attractor*, which is the desired fixed point $g = Tg$. A sufficient condition for this to happen is the *contractivity* of the image operator T in the coefficients are less than 1 in absolute value. Then the contraction mapping principle guarantees the convergence. Moreover, a corollary of this principle, which has been called the *collage theorem* in context with fractal encodings states that the overall error, i.e., the error in the attractor relative to the original is bounded by $1/(1-s)$ times the collage error, where s denotes the contractivity of T which is less than 1 in absolute value.[9] This is the motivation for the encoder to minimize the collage error under the constraint that the scaling coefficient be sufficiently small. The mathematics behind these principles have been discussed many times and we do not reproduce this material here.[10]

[9] This condition is sufficient but not necessary. Even the contractivity condition can be weakened to so-called eventual contractivity. In some special cases (domain blocks are unions of ranges, see [Oien93]) no restrictions on the scaling coefficients need to be imposed at all.

[10] In connection with iterated function systems and fractal image encoding we refer the interested reader to the books [Barn88b, BaHu93, Fish94a] and also to *Chaos and Fractals,* H.-O. Peitgen, H. Jürgens, D. Saupe, Springer-Verlag, New York, 1992.

However, we make a remark regarding the size of the domain blocks. Usually they are chosen to be twice as large as the corresponding range blocks. The contractivity condition for the image operator does *not* require a geometric contraction of domain blocks. Therefore, it is possible to use domains that are of any size, for example, they could be of the same size as the ranges (see, e.g., [BDBKS94]). It seems, however, that the error propagation at the decoder is generally worse when the geometric scaling factor is too small. Therefore, shrinking the domains to half their size is practical from a computational point of view and seems to produce the best looking results. It would be interesting to study this issue in detail.

It is common practice to enlarge the domain pool by including blocks obtained by rotating by multiples of 90 degrees and by reflection. This makes the codebook eight times as large. Larger codebooks generally improve rate-distortion curves. However, our recent systematic study [Saup96c] shows that the same or even better quality of the encodings can be achieved by enlarging the domain pool by just reducing the step size with which the image is scanned by domains. Therefore, the extra complexity that isometries introduce to the algorithm cannot be justified.

The compression ratio can be computed from

$$\text{compression ratio} = \frac{8 \times (\text{block size in pixels})}{\#\text{bits for s} + \#\text{bits for o} + \lceil \log_2(8 \times \text{codebook size}) \rceil}$$

Here the nominator is the number of bits contained in a range block of the 8-bit grey scale image. In the denominator the codebook size is multiplied by 8 in order to account for the isometries (rotations and reflections). Table 2 summarizes the performances of this simple method and compares them for different range block sizes.

range	PSNR (dB)		encoding	compression
block size	collage	attractor	time (sec)	ratio
4 × 4	36.96	36.66	147.48	4.4
8 × 8	31.15	31.27	69.93	17.7
16 × 16	27.02	26.89	59.61	70.5
32 × 32	23.55	23.32	54.76	281.0

Table 2: Example performances of the fractal baseline encoder for the Lenna test image. The collage error in the second column is typically larger than that for the attractor recovered at the decoder, i.e., the PSNR is smaller. The encoding times were measured on an SGI Indy workstation running an R4600SC 133 MHz processor with the public domain quadtree encoder of Yuval Fisher. The domain pool consists of partially overlapping domains.

One particular feature of fractal image compression is the fact that images are described only implicitly as fixed points of an image operator and with no reference to any particular image scale or size in terms of pixels. Thus, the fractal code can be decoded at *any* resolution yielding details at all scales. This justifies calling the method a *fractal* one. Of course, it is clear that the detail generated from decodings at

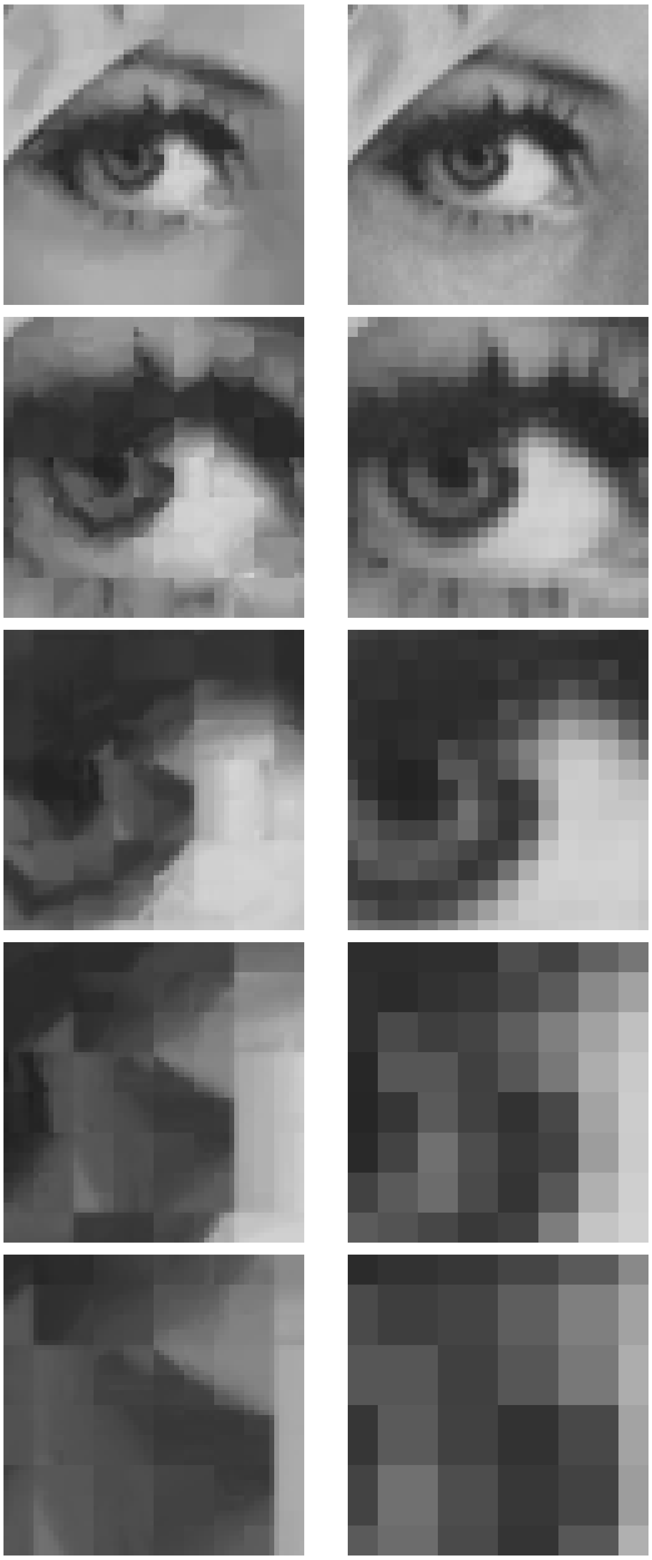

Figure 7: Fractal versus traditional zoom. From an encoding of the Lenna image (PSNR = 34.3 dB, compression ratio = 14.16) we decode the image with enlargement factors of 1, 2, 4, 8, and 12. The left column shows a section of the results. The right column presents the same zoom sequence applied to the original image.

much larger scale shown, e.g., in Figure 7, are only artificial. They do not truthfully represent any detail of the original. Yet this feature of resolution independence is useful in two regards. Firstly, the artificial details in the image are, due to the self-referential character of the code, somewhat consistent with the global appearance of the objects pictured. They look more "natural" than images obtained by mere pixel replication or interpolation. Secondly, this feature can be used as an image enhancement tool. A poor low-resolution image can be fractally encoded and then decoded at a larger resolution resulting in an enhanced version.

What we have described in this section — the baseline fractal encoder using fixed block sizes — is the most rudimentary version of fractal image compression and only meant to illustrate the essentials of it. There are many issues that are necessary to deal with in more detail when it comes to an efficient encoder capable of producing quality encodings: the partitioning, the choice of transformations, the domain pool selection, the encoder and decoder complexity, entropy coding of the fractal code, and so on. In the following sections we address some of these issues.

2 The adaptive quadtree encoder

An adaptive partitioning of an image may hold strong advantages over encoding range blocks of fixed size. There may be homogeneous image regions in which a sufficient collage can be attained using large blocks, while in high contrast regions smaller block sizes may be required to arrive at the desired quality. The first approach (already taken by Jacquin) was to consider square blocks of varying sizes, e.g., being 4, 8, and 16 pixels wide. This idea leads to the general concept of using a quadtree partition, first explored in the context of fractal coding in [JaFiBo92, BeDeKe92]. In contrast to fixed block size encodings the output file must also contain the specification of the quadtree underlying the encoding.

The use of variable partitionings makes it possible to design a variable rate encoder. The user may specify goals for either the image quality or the compression ratio. The encoder can recursively break up the image into suitable portions until either criterion is reached. In more detail the algorithm targeting fidelity might proceed as follows.

1. Define a tolerance for the root-mean-square error $E(R, D)/\sqrt{\#\text{pixels in } R}$ of the collage, a minimal and a maximal range size. Partition the image into ranges of maximal size.

2. Initialize a stack of ranges by pushing the maximal size ranges onto it.

3. While the stack is nonempty carry out the following steps:

    (a) Pop a range block R from the stack and search the corresponding codebook yielding an optimal approximation $R \approx sD + o\mathbf{1}$ and a least error $E(D, R)$.

    (b) If the root-mean-square error is less than the tolerance or if the range size is equal to the minimum range size, then save the code for the range, i.e., s, o, isometry, and address of D. If $s = 0$, do not store the rest.

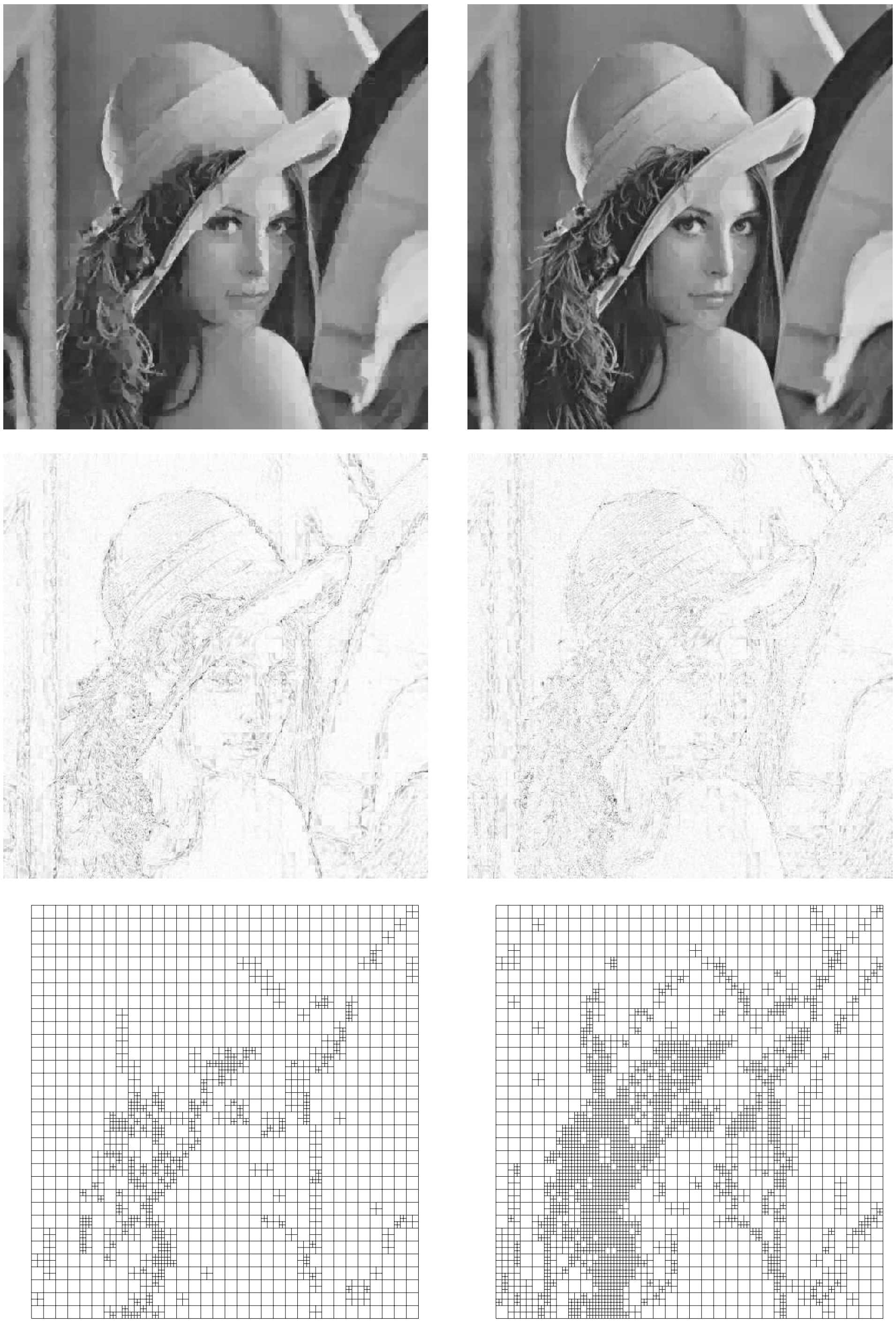

Figure 8: Results of two encodings using quadtree partitionings. Shown are a low and a medium quality encoding with error images (large errors scaled to black) and corresponding quadtrees. The PSNR values are 28.3 dB (left) and 32.0 dB (right). The compression ratios are 37.5 and 17.8.

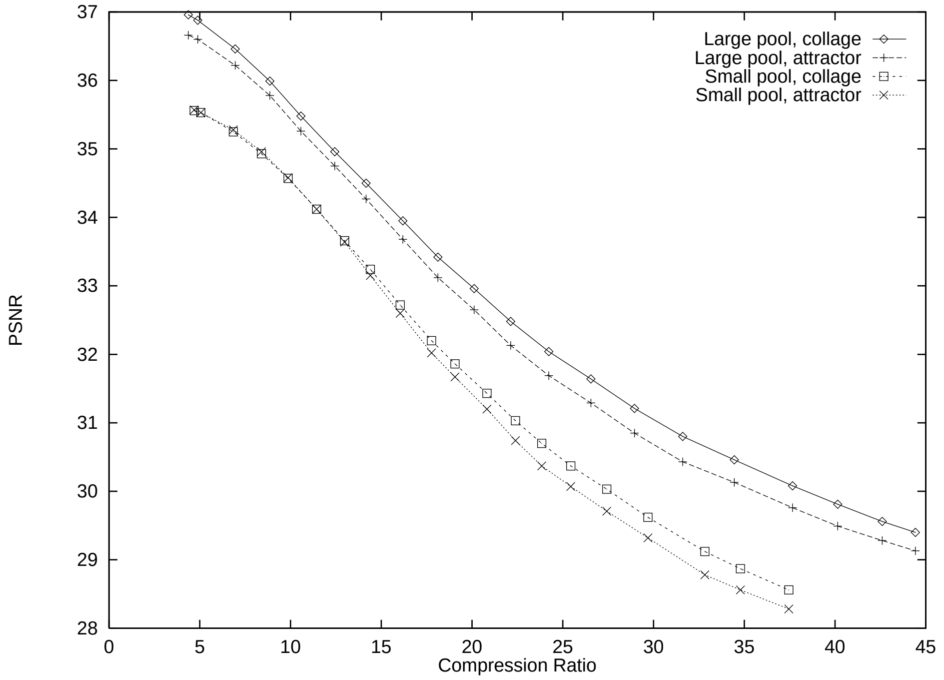

Figure 9: Rate-distortion curves for the fractal quadtree encoder and the test image Lenna 512×512. Shown are the results for two runs with differing domain pool sizes. The small domain pool contains only non-overlapping domains, while the large pool is generated using a fixed step size of 4 pixels horizontally and vertically. In each case we also show the achieved PSNR quality of the collage. The difference between the collage and attractor curves stems from the error propagation at the decoder. The corresponding cpu run times are given in Figure 10.

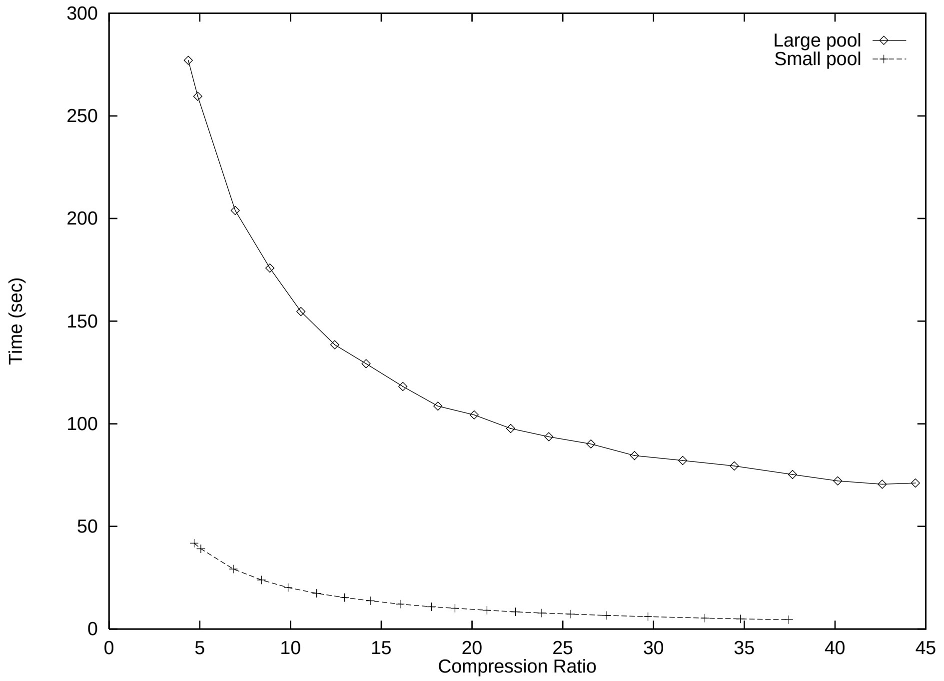

Figure 10: Run times for the fractal quadtree encodings of Figure 9. Measurements were taken on an SGI Indy workstation running an R4600SC 133 MHz processor. For speed the classification of domains and ranges was employed.

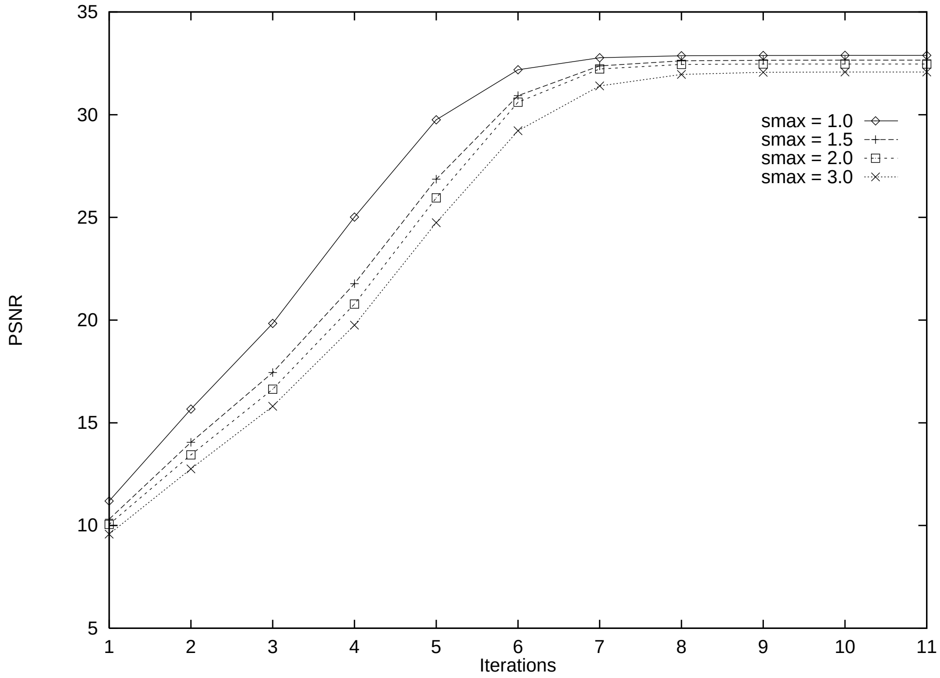

Figure 11: Decoder speed for varying maximal scaling factors s_{max}. The compression ratios for these encodings are about the same, between 18 and 19.

(c) Otherwise partition R into four quadrants and push them onto the stack.

By using different fidelity tolerances for the collage one obtains a series of encodings of varying compression ratios and fidelities. Two such encodings along with their error images and quadtrees are shown in Figure 8. We computed several more quadtree encodings, the results of which are summarized in Figures 9 and 10 showing rate-distortion curves and PSNR versus encoding time. All experiments were done using the quadtree encoder of Fisher [Fish94a] (see also the section on WEB resources below).

The decoder for the quadtree codes proceeds in the same way as for the case of fixed block size encodings, i.e., by iteration of the collage image operator. Only 7 or 8 iterations are required to get sufficiently close to the attractor. In Figure 11 we study the effect of changing the maximal allowable scaling factor s_{max}. Allowing scaling factors larger than 1 may destroy the contractivity property of the image operator, but this does not necessarily harm the convergence at the decoder. The reason for this is that the operator may be eventually contractive, i.e., only an iterate of T is a contraction even though T itself is not. Larger maximal scaling factors s_{max} may be tempting as one may get better collages. However, enlarging the allowed range of s without also increasing the bit allocation for the storage of s may actually worsen the result as shown in the figure. The reason is that the quantization of the scaling factors becomes less accurate.

We do not further elaborate on the various aspects of quadtree encodings. They are very well covered and documented in the book [Fish94a].

3 Complexity reduction techniques

Fractal image compression allows fast decoding but suffers from long encoding times. The time consuming part of the encoding step is the search for an appropriate domain for each range. The number of possible domains that theoretically may serve as candidates is prohibitively large. For example, the number of arbitrarily sized square subregions in an image of size n by n pixels is of order $O(n^3)$. Thus, one must impose certain restrictions in the specification of the allowable domains. In a simple implementation one might consider as domains, e.g., only sub-squares of a limited number of sizes and positions. This defines the so-called *domain pool.* Now for each range in the partition of the original image all elements of the domain pool are inspected. If the number of domains in the pool is N_D, then the time spent for each search is *linear* in N_D, $O(N_D)$. Several methods have been devised to reduce the time complexity of the encoding. In this section we review these methods. At the start we have to set up some notation.

3.1 A formula for the least squares error based on projections

For the discussion in this section let us assume that an image is partitioned into non-overlapping square range blocks of size $N \times N$. This is not a restriction since it will be clear how the principles described carry over to more general partitions.

We consider each range block as a vector R in the linear vector space $\mathbf{R}^n$ where $n = N \times N$. The conversion from a square subimage of side length N to a vector of length $n = N^2$ can be accomplished, e.g., by scanning the block line by line. Working with vectors in place of 2D-arrays simplifies the notation considerably without losing generality.

The *domain pool* is a collection of square blocks which are typically larger than the ranges and taken also from the image, called *domain blocks.* The domain pool is enlarged by including blocks obtained after applying the eight isometrical operators to the domain blocks (i.e., rotations and reflections). Finally, by pixel averaging, the size of these blocks is reduced to the size of a range block. The resulting blocks are called *codebook blocks.*

In the encoding process for a range block a search through the codebook blocks is required. A vector representing a codebook block will be denoted by D. A small set of $p < n$ blocks independent from the image is also considered. We represent them by the vectors $B_1, B_2, \ldots, B_p \in \mathbf{R}^n$, which are chosen so as to form an orthonormal basis of a p-dimensional subspace of $\mathbf{R}^n$. They are known as *the fixed basis blocks*[11]. The encoding problem can then be stated as the least squares problem

$$E(D, R) = \min_{a, b_1, \ldots, b_p \in \mathbf{R}} ||R - (aD + \sum_{k=1}^{p} b_k B_k)|| = \min_{x \in \mathbf{R}^{p+1}} ||R - Ax||, \tag{3}$$

where A is an $n \times (p+1)$ matrix whose columns are $D, B_1, B_2, \ldots, B_p$ and $x =$

[11] Here we generalize the discussion from the previous section, where only one fixed basis block has been considered, namely $\mathbf{1}$. The scaling and offset coefficients s and o are now called a and b_1.

$(a, b_1, \ldots, b_p) \in \mathbf{R}^{p+1}$ is a vector of coefficients.[12] This problem should be solved for all codebook blocks D and the one which gives the smallest error $||R - (aD + \sum_1^p b_k B_k)||$ is selected on condition that the value of the scaling factor a for the codebook block D ensures the convergence of the decoding process (e.g., by requiring $|a| < 1$). This condition on a can be removed when one uses the orthogonalized representation of Øien [Oien93]. A basic result of linear algebra states that if the codebook block D is not in the linear span of the fixed basis blocks $B_1, \ldots, B_p$, then the minimization problem (3) has the unique solution

$$\bar{x} = (A^T A)^{-1} A^T R$$

where the matrix $A^+ = (A^T A)^{-1} A^T$ is also known as the pseudo-inverse of A. Thus, the range block R is approximated by the *collage* block AA^+R where AA^+ is the orthogonal projection matrix onto range(A). Now let P be the orthogonal projection operator which projects $\mathbf{R}^n$ onto the subspace $\mathcal{B}$ spanned by only the fixed basis blocks $B_1, B_2, \ldots, B_p$. Thus, by orthogonality of the fixed basis blocks we have for $R \in \mathbf{R}^n$

$$PR = \sum_{k=1}^{p} b_k B_k = \sum_{k=1}^{p} \langle R, B_k \rangle B_k.$$

Then the range block R has a unique orthogonal decomposition $R = OR + PR$ where the operator $O = I - P$ projects onto the orthogonal complement $\mathcal{B}^\perp$. For $Z = (z_1, \ldots, z_n) \in \mathbf{R}^n \backslash \mathcal{B}$, we define the operator

$$\phi(Z) = \frac{OZ}{||OZ||}. \tag{4}$$

Now for a given domain block $D \notin \mathcal{B}$ the collage block AA^+R can be given explicitly as

$$AA^+R = \langle R, \phi(D) \rangle \phi(D) + \sum_{k=1}^{p} \langle R, B_k \rangle B_k. \tag{5}$$

To get the least squares error we use the orthogonality of $\phi(R), B_1, \ldots, B_p$ to express the range block R as

$$R = \langle R, \phi(R) \rangle \, \phi(R) + \sum_{k=1}^{p} \langle R, B_k \rangle B_k. \tag{6}$$

We insert the result for R in the first part of the collage block AA^+R in (5) and after three lines of computations find that

$$\langle R, \phi(D) \rangle \phi(D) = \langle R, \phi(R) \rangle \, \langle \phi(D), \phi(R) \rangle \, \phi(D).$$

[12] Note that we use the norm of the error in place of the squared norm. This simplifies the notation for the following sections. Moreover, in practice usually the root mean square error (rms) is used equivalently in place of $E(D, R)$. This is just $E(D, R)/\sqrt{n}$. We use the notation $\langle \cdot, \cdot \rangle$ for the common inner product in $\mathbf{R}^n$, thus, $||x|| = \sqrt{\langle x, x \rangle}$.

Thus, the collage block can be rewritten as

$$AA^+R = \langle R, \phi(R)\rangle \langle \phi(D), \phi(R)\rangle \phi(D) + \sum_{k=1}^{p} \langle R, B_k\rangle B_k. \tag{7}$$

Using (6) and (7) we can now compute the error

$$E(D,R) = ||R - AA^+R|| = \sqrt{\langle R - AA^+R, R - AA^+R\rangle}.$$

The result follows after a few lines of calculations, namely

$$E(D,R) = \langle R, \phi(R)\rangle \sqrt{1 - \langle \phi(D), \phi(R)\rangle^2}. \tag{8}$$

Thus, the minimization of the error $E(D,R)$ among domain codebook blocks D can be achieved using an *angle criterion:* The minimum of $E(D,R)$ occurs when the squared inner product $\langle \phi(D), \phi(R)\rangle^2$ is maximal. Since

$$\langle \phi(D), \phi(R)\rangle^2 = \cos^2 \angle(\phi(D), \phi(R))$$

this means minimizing the angle $\angle(\phi(D), \phi(R))$, or, equivalently $\angle(OD, OR)$.

3.2 Feature vectors

In the feature vector approach introduced by Saupe in [Saup94a, Saup94b, Saup95a] a small set of d real-valued keys is devised for each domain which make up a d-dimensional feature vector. These keys are carefully constructed such that searching in the domain pool can be restricted to the *nearest neighbors* of a query point, i.e., the feature vector of the current range. Thus, the sequential search in the domain pool is substituted by multi-dimensional nearest neighbor searching which can be run in logarithmic time.

We consider a set of N_D codebook blocks $D_1, \ldots, D_{N_D} \in \mathbf{R}^n$ and a range block $R \in \mathbf{R}^n$. We let $E(D_i, R)$ denote the smallest possible error of an approximation of the range data R by an affine transformation of the codebook block D_i. In terms of a formula, this is

$$E(D_i, R) = \min_{a, b_1, \ldots, b_p \in \mathbf{R}} ||R - (aD_i + \sum_{k=1}^{p} b_k B_k)||.$$

The following theorem provides the mathematical foundation for our feature vector approach.

Theorem 1 [Saup94a, Saup94b].
Let $n \geq 2$ and $X = \mathbf{R}^n \backslash \mathcal{B}$. Define the function $\Delta : X \times X \to [0, \sqrt{2}]$ by

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

For $D_i, R \in X$ the error $E(D_i, R)$ is given by

$$E(D_i, R) = \langle R, \phi(R)\rangle\, g(\Delta(D_i, R))$$

where

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

Proof. The least squares approximation of a range block R was given by equation (5) in Section 3.1:

$$\langle R, \phi(D)\rangle\, \phi(D) + \sum_{k=1}^{p} \langle R, B_k\rangle B_k. \tag{9}$$

By orthogonality we can express the range block as

$$R = \langle R, \phi(R)\rangle\, \phi(R) + \sum_{k=1}^{p} \langle R, B_k\rangle B_k. \tag{10}$$

Using this in the first expression (9) we obtain

$$\langle R, \phi(R)\rangle\, \langle \phi(D), \phi(R)\rangle\, \phi(D) + \sum_{k=1}^{p} \langle R, B_k\rangle B_k \tag{11}$$

for the least squares approximation of R. The square of the difference of (6) and (7) gives us the least squares error and is calculated as

$$E(D, R) = \langle R, \phi(R)\rangle \sqrt{1 - \langle \phi(D), \phi(R)\rangle^2}.$$

Since

$$||\phi(R) \pm \phi(D)|| = \sqrt{2(1 \pm \langle \phi(D), \phi(R)\rangle)}$$

we have

$$\Delta(D, R) = \sqrt{2(1 - |\langle \phi(D), \phi(R)\rangle|)}.$$

Solving for $|\langle \phi(D), \phi(R)\rangle|$ and inserting the square of the result in the formula for $E(D, R)$ completes the proof.

The theorem states that the least squares error $E(D_i, R)$ is proportional to the simple function g of the Euclidean distance Δ between the projections $\phi(D_i)$ and $\phi(R)$ (or $-\phi(D_i)$ and $\phi(R)$). Since $g(\Delta)$ is a monotonically increasing function for $0 \leq \Delta \leq \sqrt{2}$ we conclude that *the minimization of the errors* $E(D_i, R)$ *for* $i = 1, \ldots, N_D$ *is equivalent to the minimization of the distance expressions* $\Delta(D_i, R)$. Thus, we may replace the computation and minimization of N_D least squares errors $E(D_i, R)$ by the search for the nearest neighbor of $\phi(R) \in \mathbf{R}^n$ in the set of $2N_D$ vectors $\pm\phi(D_i) \in \mathbf{R}^n$. The problem of finding closest neighbors in Euclidean spaces has been thoroughly studied in computer science. For example, a method using kd-trees that runs in expected logarithmic time is presented by Friedman, Bentley, and Finkel[13] together with pseudo code. After a preprocessing step to set up the required kd-tree, which takes $O(N \log N)$ steps, the search for the nearest neighbor of a query point can be completed in expected logarithmic time, $O(\log N)$. However, as the

[13]Friedman, J. H., Bentley, J. L., Finkel, R. A., *An algorithm for finding best matches in logarithmic expected time,* ACM Trans. Math. Software 3,3 (1977) 209–226.

dimension d increases, the performance may suffer. A method that is more efficient in that respect, presented by Arya et al[14], produces a so-called approximate nearest neighbor. For domain pools that are not large other methods, that are not based on space-partitioning trees, may perform better. For example, the modified equal average nearest neighbor search (ENNS)[15] seems to be one of the best. Before we turn to practical issues, we remark, that we can use the result of the Theorem 1 in order to identify all codebook blocks D_i that satisfy a given tolerance criterion $E(R, D_i) \leq \delta$. In other words, solving the equality for Δ in the expression for the error $E(D, R)$ in the theorem yields a necessary and sufficient condition for a codebook block D to fulfill the tolerance criterion.

Corollary 2 (A necessary and sufficient condition)
Let $\delta > 0$ and $n \geq 2$. Let R and D be in $\mathbf{R}^n \backslash \mathcal{B}$ with $\langle R, \phi(R)\rangle \geq \delta$. Then $E(D, R) = \min_{a,b_1,\dots,b_p \in \mathbf{R}} \|R - (aD + \sum_{k=1}^{p} b_k B_k)\| \leq \delta$ if and only if

$$\Delta(D, R) \leq \sqrt{2 - 2\sqrt{1 - \frac{\delta^2}{\langle R, \phi(R)\rangle^2}}},$$

where $\Delta(D, R)$ is defined as in Theorem 1.

Proof. From Theorem 1, $E(D, R) = \langle R, \phi(R)\rangle\, g(\Delta(D, R))$ with $g(\Delta) = \Delta\sqrt{1 - \frac{\Delta^2}{4}}$. Thus, for $0 \leq \Delta \leq \sqrt{2}$ we have $E(D, R) \leq \delta$ if and only if $\Delta^4 - 4\Delta^2 + 4\delta^2/\langle R, \phi(R)\rangle^2 \geq 0$. From this the assertion easily follows.

The condition $\langle R, \phi(R)\rangle \geq \delta$ does not impose any restrictions. To see this, observe that in the case of $\langle R, \phi(R)\rangle < \delta$ we already have

$$E(D, R) = \langle R, \phi(R)\rangle \sqrt{1 - \langle \phi(D), \phi(R)\rangle^2} \leq \langle R, \phi(R)\rangle < \delta.$$

for *any* codebook block D. Thus, it suffices to encode R only using the fixed basis blocks, i.e., by $\sum_{k=1}^{p} b_k B_k$.

We continue with some remarks on generalizations and implications of the theory presented above. In practice, there is a limit in terms of storage for the feature vectors of domains and ranges. For example, the keys for ranges of size of 8 by 8 pixels require 64 floating point numbers each. Thus, 32K domains from a domain pool would already fill 8 MB of memory on a typical workstation, while we would like to work with pools of a hundred thousand and more domains. To cope with this difficulty, we settle for a compromise and proceed as follows. We *down-filter* all ranges and domains to some prescribed dimension of moderate size, e.g., $d = 4 \times 4 = 16$. Moreover, each of the d components of a feature vector is *quantized* (8 bits/component suffice). This allows the processing of an increased number of domains and ranges,

[14] Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., Wu, A., *An optimal algorithm for approximate nearest neighbor searching,* Proc. 5th Annual ACM-SIAM Symposium on Discrete Algorithms (1994) 573–582.

[15] Lee, C.-H., Chen, L. H, *Fast closest codeword search algorithm for vector quantization,* IEE Proc.-Vis. Image Signal Process. 141, 3 (1994) 143–148.

however, with the implication that the formula of the theorem is no longer exact but only approximate. This, however, is not a severe disadvantage as pointed out in the following remark and as demonstrated by many experiments [Saup95b].

The approach of pixel averaging in order to reduce the dimensionality of the domains and ranges (64 and higher is typical) to a more feasible number (here $d = 16$) may be improved by better concentrating relevant subimage information in the d components. Based on our report [Saup94a] Barthel et al [BSVN94] have suggested and implemented an alternative reduction of dimension. They have used a two-dimensional discrete cosine transformation (DCT) of the projected codebook blocks $\pm\phi(D_i)$. The distance preserving property of the unitary transform carries over the result of Theorem 1 to the frequency domain and nearest neighbors of DCT coefficient vectors will yield the smallest least squares errors. In practice one computes the DCT for all domains and ranges. Then, from the resulting coefficients, the DC component is ignored and the next d coefficients are normalized and make up the feature vector.

Because of the downfiltering and the quantization of both the feature vectors and the coefficients $a, b_1, \ldots, b_p$, it can happen that the nearest neighbor in feature vector space is not the codebook block with the minimum least squares error using quantized coefficients. Moreover, it could yield a scaling factor a being too large to be allowed. To take that into consideration, we search the codebook not only for the nearest neighbor of the given query point but also for, say, the next 5 or 10 nearest neighbors (this can still be accomplished in logarithmic time using a priority queue). From this set of neighbors the non-admissible domains are discarded and the remaining domains are compared using the ordinary least squares approach. This also takes care of the problem from the previous remark, namely that the estimate by the theorem is only approximate. While the domain corresponding to the closest point found may not be the optimal one, there are usually near-optimum alternatives among the candidates.

We make two technical remarks concerning *memory requirements* for the kd-tree. Firstly, it is not necessary to create the tree for the full set of $2N_D$ keys in the domain pool. We need to keep only one multi-dimensional key per domain, e.g., by keeping only the key which has a non-negative first component (multiply key by -1 if necessary). In this set-up a kd-tree of all $2N_D$ vectors has two symmetric main branches (separated by a coordinate hyperplane), thus, it suffices to store only one of them. Secondly, there is some freedom in the choice of the *geometric transformation* that maps a domain onto a range coming from the 8 possible rotations and reflections of a square subimage. This will create a total of 8 entries per domain in the kd-tree, enlarging the size of the tree. However, we can get away without this tree expansion. To see this, just note that we may instead consider the 8 transformations of the *range* and search the original tree for nearest neighbors of each one of them.

The *preprocessing time* to create the data structure for the multi-dimensional search is not a limitation of the method as demonstrated by our experiments.

A forerunner of feature vectors as described above has been presented by Hürtgen and Stiller [HuSt93]. As in the classification of Fisher, Jacobs, and Boss an image block is partitioned into its four quadrants and their mean intensities are computed. Then a vector consisting of four bits is constructed as follows: the i-th bit is 1 if the mean of the i-th quadrant is above the overall mean, and 0 otherwise. Thus, in the terminology of his paper, this is our feature vector after downsampling to size $d = 2 \times 2$ and quantizing to 1 bit per component. Due to these strict limitations a nearest neighbor search is not practical, rather, these vectors serve as a means for classification into 16 classes. Then a range is compared only with codebook blocks from the same class.

3.3 Classification schemes

The classification as described below has been explained only for the case $p = 1$, where just one fixed basis block of constant intensity $B = 1/\sqrt{n}(1, \ldots, 1)$ is used. However, at this point we already notice that the method extends to the general case allowing $p > 1$, provided that certain modifications are made. Essentially, this amounts to considering the transformed domains $\phi(D_i)$ in place of the original domains.

3.3.1 Jacquin's approach

In his original work [Jacq89b, Jacq92] Jacquin used a classification scheme coming from a study of Ramamurthi and Gersho[16]. The domain blocks are classified according to their perceptual geometric features. Only three major types of blocks are differentiated: shade blocks, edge blocks, and midrange blocks. In shade blocks the image intensity varies only very little, while in edge blocks a strong change of intensity occurs, e.g., along a boundary of an object displayed in the image. The class of edge blocks is further subdivided into two subclasses: simple and mixed edge blocks. Midrange blocks have larger intensity variations than shade blocks, but there is no pronounced gradient as in an edge block. Thus, these blocks typically are blocks containing texture. Since ranges that would be classified as shade blocks can be approximated well by the constant fixed block B, scaled by an appropriate factor b, it is not necessary to search for a corresponding domain for them (in effect setting the coefficient $a = 0$). Thus, in this scheme there are really only two (major) classes, one of which must be searched for each non-shade block range.

3.3.2 Classification by intensity and variance

A more elaborate classification technique was proposed by Boss, Fisher and Jacobs [JaFiBo92, Fish94b]. It works as follows. A square range or domain is subdivided into its four quadrants (upper left, upper right, lower left, and lower right). In the quadrants the average pixel intensities A_i and the corresponding variances V_i are computed ($i = 1, \ldots, 4$). It is easy to see that one can always orient (rotate and flip) the range or domain such that the average intensities are ordered in one of the three ways:

$$\text{major class 1: } A_1 \geq A_2 \geq A_3 \geq A_4,$$

[16] Ramamurthi, B., Gersho, A., *Classified vector quantization of images,* IEEE Trans. Commun., COM-34, 1986.

$$\text{major class 2: } A_1 \geq A_2 \geq A_4 \geq A_3,$$
$$\text{major class 3: } A_1 \geq A_4 \geq A_2 \geq A_3.$$

Once the orientation of the range or domain has been fixed accordingly, there are 24 different possible orderings of the variances which define 24 subclasses for each major class. If the scale factor a in the approximation $aD + bB$ of the range block R is negative then the orderings in the classes must be modified accordingly. Thus, for a given range two subclasses out of 72 need to be searched in order to accommodate positive and negative scale factors.

Although successful this approach is not satisfying in the sense that a notion of neighboring classes is not available. So if the search in one class does not yield a sufficiently strong match for a domain, one cannot easily extend the search to any neighboring classes. A solution for this problem has been given by Caso, Obrador and Kuo in [CaObKu95], where the unflexible ordering of variances of an image block has been replaced by a vector of variances. These variance vectors are strongly quantized leading to a collection of classes where each class has a neighborhood of classes which can be searched. Another solution is offered by clustering methods discussed below.

3.3.3 Archetype classification

A method that defines the classes a priori by some empirical studies carried out on a collection of training images is the archetype classification presented by Boss and Jacobs in [BoJa94]. An archetype for a set of codebook blocks is given by that particular codebook block that can best cover all others in the usual least squares sense. For a set of blocks D_i this is the block D_k,

$$D_k = \arg\min_{D_k} \sum_{i \neq k} \min_{a,b} ||D_i - (aD_k + bB)||.$$

Starting out from an arbitrary classification (e.g., the one given by Fisher et al above) of subimage blocks taken from a set of training images one can compute the archetype for each class. Then the blocks are reclassified according to the archetype by which they can be covered best. This yields a new classification, and the process of archetype computation and reclassification is repeated until self-consistency, i.e., until no further change occurs in an iteration. The final set of archetypes becomes a part of the encoder. Given an image to be compressed, the encoder defines the domain pool and classifies all codebook blocks, i.e., for each codebook block the archetype is found that can best cover the block under consideration. In this way it can be expected that a given range can be covered very well by a block in the corresponding class. In fact, this is verified in the experiments reported in the paper. Thus, in order to arrive at a certain image fidelity, one needs to search fewer classes, which saves some computing time. On the other hand, the classification process is more elaborate. As a result, a conventional classification scheme is overall faster for low quality image encoding, while the best image fidelity can be attained much faster using the archetype classification.

3.3.4 Clustering methods

In clustering methods domains and ranges are grouped around cluster centers which are computed either adaptively from the test image or from a set of training images. The classes will depend on the clustering algorithm chosen, and on the criterion function used to describe the quality of the clustering. The first attempt to adaptive clustering with Kohonen's Self-Organizing Map (SOM) for fractal image compression was presented in [BoMe92]. However, the reported results were not satisfying. An implementation employing frequency sensitive competitive learning is reported in [WaKi93]. An efficient clustering method based on the LBG algorithm was proposed in [OiLeRa92, Leps93, Oien93, LeOi94]. These important works introduced also a block decimation technique to perform the clustering and the searching at a low dimensional space. In [Hamz95], the SOM approach has been successfully combined with the block intensity classification of Fisher et al. [FiJaBo92, Fish94a], and the nearest neighbor approach of Saupe [Saup94b] to yield a distance based classification scheme. In the following we explain our clustering approach. Let $\{\pm\phi(D_1), \ldots, \pm\phi(D_{N_D})\}$ be the set of projected codebook blocks. We want to partition this set into a finite number of disjoint subsets (clusters) defined by representatives (cluster centers) such that vectors in the same cluster are closer to each other than vectors in different clusters. The quality of the clustering can be measured by a criterion function that one tries to optimize. For example, one can choose to construct the cluster centers such that the sum of squared Euclidean distances $J = \sum_{i=1}^{N_D} ||\phi(D_i) - m(\phi(D_i))||^2 + || - \phi(D_i) - m(-\phi(D_i))||^2$ is minimized. Here $m(\pm\phi(D_i))$ denotes the cluster center closest to the projected codebook block $\pm\phi(D_i)$. A cluster of center m is formed by grouping around m all projected codebook blocks having m as their nearest neighbor. After the cluster centers have been designed, the set of projected codebook blocks $\{\pm\phi(D_1), \ldots, \pm\phi(D_{N_D})\}$ is clustered by mapping each vector $\pm\phi(D_i)$ to its nearest cluster center. A range block R is encoded in two steps. First, we map its feature vector $\phi(R)$ to its closest cluster center $m(\phi(R))$. Then the range block R is compared only to the codebook blocks whose feature vectors are in the cluster of center $m(\phi(R))$. This corresponds to a 1-class search. We can evidently search in more classes by considering the next nearest cluster centers of $\phi(R)$. This will yield more accurate encodings at the expense of increased time. The reason why the method works is obvious. Suppose that both $\phi(D_i)$ (or $-\phi(D_i)$) and $\phi(R)$ are close enough to cluster center m. Then the triangular inequality ensures that $\Delta(D_i, R)$ is small enough. Thus, by Theorem 1, codebook block D_i will provide a good match for range block R. To avoid the heavy computations involved when the blocks have a high dimension, the clustering is performed at a low dimension. However, contrary to Øien's approach, we orthonormalize the blocks *after* decimation.

3.3.5 Invariant moments

In [Nova93a], Novak assigns a 4-dimensional feature vector to each block. The components of the feature vector are certain moment invariants defined from the grey level distribution within the block. A useful property of these moment invariants is that they are invariant with respect to the geometric transformation, i.e., one feature vector suffices for each domain. The isometric versions of that domain block then have the same moment vector. However, the moments are not invariant w.r.t. the affine transformation regarding the luminance. To cope with this problem Novak proposed a normalization procedure. There are three problems with this approach: The negative intensity blocks are omitted from consideration causing the loss of some of the possible fidelity. The values of the invariant moments range over several orders of magnitude, thus, a logarithmic rescaling becomes necessary before nearest neighbor search becomes feasible. And then, most importantly, the method is intuitive in the sense that no supporting theory is given to the goal that closeness in the feature space ensures good approximations in the least squares sense. The fact is, that such a theory cannot exist. Novak worked with triangular partitioning, and Frigaard continued the work in [Frig95] using a quadtree partitioning. However, Frigaard does not normalize feature vectors with respect to mean and variance in order to make the moments invariant relative to the affine luminance transformation. He reports that normalizing would in fact degrade the overall quality of an encoding of an image, which apparently documents the weakness of the method.

Götting, Ibenthal, and Grigat [GoIbGr95] and Popescu and Yan [PoYa93] also pursue complexity reduction using invariant moments of different types.

3.4 Functional methods

In [BeDeKe92], Bedford, Dekking and Keane proposed a criterion which tells when a codebook block cannot provide a good approximation to a range block. The idea is to compare not ranges agains domains, but rather to compare ranges and domains independently against a certain unit vector. Only when these comparisons come out about the same can a range be covered by a given domain. One can thus reduce the encoding time by eliminating a large number of codebook blocks. We do not give their original result but generalize to arbitrary unit vectors and also to the case of fractal image compression with several fixed basis blocks.

Theorem 3 *[SaHa94a]* **(A necessary condition)**
Let $\delta > 0$, and let U be a unit vector in $\mathbf{R}^n$. Let R and D be in $\mathbf{R}^n$ with $\langle R, \phi(R)\rangle \geq \delta$. If $E = \min_{a,b_1,\ldots,b_p \in \mathbf{R}} ||R - (aD + \sum_1^p b_k B_k)|| \leq \delta$ then

$$\Big| |\langle \phi(R), U\rangle| - |\langle \phi(D), U\rangle| \Big| \leq \sqrt{2 - 2\sqrt{1 - \frac{\delta^2}{\langle R, \phi(R)\rangle^2}}}. \tag{12}$$

Proof. We compute using the Cauchy-Schwarz inequality

$$\begin{aligned}(|\langle \phi(R), U\rangle| - |\langle \phi(D), U\rangle|)^2 &\leq (\langle \phi(R), U\rangle - \langle \phi(D), U\rangle)^2 \\ &= |\langle \phi(R) - \phi(D), U\rangle|^2 \leq ||\phi(R) - \phi(D)||^2 \cdot ||U||^2 \\ &= ||\phi(R) - \phi(D)||^2 = 2 - 2\langle \phi(R), \phi(D)\rangle. \end{aligned} \tag{13}$$

If ρ denotes $\langle \phi(R), \phi(D) \rangle$, then the square of the error is $E^2 = \langle R, \phi(R) \rangle^2 (1 - \rho^2)$ (see the proof of Theorem 1). Thus, it follows from the assumption $E \leq \delta$ that

$$\langle R, \phi(R) \rangle^2 (1 - \rho^2) \leq \delta^2.$$

We may assume $\rho \geq 0$ (otherwise replace D by $-D$) and obtain

$$\rho \geq \sqrt{1 - \frac{\delta^2}{\langle R, \phi(R) \rangle^2}}.$$

Inserting this result in the inequality (13) completes the proof. It is easy to check that in the case $p = 1$ where $\mathcal{B}$ is spanned by the fixed block of constant intensity, $B = (1, 1, \ldots, 1)/\sqrt{n}$, we have for any block $Z \in \mathbf{R}^n$

$$\phi(Z) = \frac{1}{\sqrt{V(Z)}} (z_1 - \overline{z}, \ldots, z_n - \overline{z}),$$

where $\overline{z} = (z_1 + \cdots + z_n)/n$ is the average intensity and $V(Z) = \sum_{k=1}^{n} (z_k - \overline{z})^2$ the n-fold variance. This is the special case given in [BeDeKe92] where the condition $\langle R, \phi(R) \rangle \geq \delta$ was stated as $V(R) \geq \delta^2$.

The algorithm to encode a range block R can be described as follows:

Algorithm 4 (A functional algorithm)

1. *Choose a tolerance δ and a unit vector U.*
2. *(Preprocessing) For every codebook block D compute $|\langle \phi(D), U \rangle|$.*

For each range R do:

3. *Compute $\langle R, \phi(R) \rangle$ and the upper bound in (12).*
4. *If $\langle R, \phi(R) \rangle < \delta$, then no search is needed since $a = 0$ gives the least squares error $E = \langle R, \phi(R) \rangle \leq \delta$ for any codebook block D.*
5. *If $\langle R, \phi(R) \rangle \geq \delta$ then compute $|\langle \phi(R), U \rangle|$ and reject all the codebook blocks D for which the inequality (12) of Theorem 3 is not fulfilled.*

It is possible to enhance this functional method by considering several unit vectors U for the criterion in the theorem. In this way one can expect to discard a larger set of domain blocks for a given range block.

In an efficient implementation of the functional method above one would not scan the entire domain pool to extract those domains that pass the test of the theorem. Instead, with any functional method it is better to proceed along the following algorithm:

Algorithm 5 (General functional method)
Assume that a function $F : \mathbf{R}^n \to \mathbf{R}$ is given such that $|F(R) - F(D)| \leq \epsilon_R$ implies that the range R can be covered well by the domain D. Let the domain pool be denoted by $\{D_1, \ldots, D_{N_D}\}$. In a preprocessing step do:

1. *For every domain* $D \in \{D_1, \ldots, D_{N_D}\}$ *compute* $F(D)$.

2. *Sort all domains according to the functional value in a linear array and relabel domains such that* $F(D_1) \leq F(D_2) \leq \cdots \leq F(D_{N_D})$.

For each range R *do:*

3. *Compute* $F(R)$ *and the upper bound* ϵ_R *(e.g., the right term in (12) of Theorem 3).*

4. *Using, e.g., the bisection method find the indices* k_0, k_1 *such that* $|F(R) - F(D_k)| \leq \epsilon_R$ *if and only if* $k_0 \leq k \leq k_1$.

5. *Check all domains* D_k *with* $k_0 \leq k \leq k_1$.

In this procedure a list of candidate domains D_k is produced in $O(\log N_D)$ time while the full scan rejecting the domains that do not pass the test takes $O(N_D)$ time.

3.5 Tree structured methods

Besides the dimensional reduction and the variance based classification mentioned above Caso, Obrador and Kuo propose a tree structured search in [CaObKu95]. The pool of codebook blocks is recursively organized in a binary tree. Initially two (parent) blocks are chosen randomly from the pool. Then all blocks are sorted into one of two bins depending on by which of the two parent blocks the given block can be covered best in the least squares sense. This results in a partitioning of the entire pool into two subsets. The procedure is recursively repeated for each one of them until a prescribed bucket size is reached. Given a range one can then compare this block with the blocks at the nodes of the binary tree until a bucket is encountered at which point all of the codebook blocks in it are checked. This does not necessarily yield the globally best match. However, the best one (or a good approximate solution) can be obtained by extending the search to some nearby buckets. A numerical test based on the angle criterion is given for that purpose. The procedure is related to the nearest neighbor approach since the least squares criterion (minimize $E(D, R)$) is equivalent to the distance criterion (minimize $\Delta(\phi(D), \phi(R))$. Thus, the underlying binary tree can be considered to be randomized version of the kd-tree structure we have used here.

Van der Walle [Wall95] worked on a wavelet representation of fractal image compression, where similarly to ordinary fractal image compression, range vectors (corresponding to subtrees of the tree of wavelet coefficients) have to be matched with domain vectors (also corresponding to nodes of the wavelet tree), which may be scaled by an arbitrary scaling factor. For each node a feature vector is generated based on angles between the coefficient vectors and axes in the wavelet coefficient space. These vectors are then sorted into a multi-dimensional space-partitioning data structure within which the fast search is organized. In terms of distances of feature vectors $\pm\phi(D)$ the interpretation is as follows: We define a small set of anchor points in feature space (e.g., at the positions of the main principal components of the set of all feature vectors). For each (projected and normalized) codebook block as well as for each range block we compute the distances Δ to the anchor points. Then a point in feature space that is close to a given range feature vector must necessarily have distances to the anchor points that are near those of the range. To facilitate the search for such codebook blocks, the blocks can be organized in a tree structure.

3.6 Multiresolution approaches

Two multi-resolution approaches for encoder complexity reduction are presented by Dekking in [Dekk95a, Dekk95b], and by Lin and Venetsanopoulos in [LiVe95a]. The idea is to use the grey value pyramid associated with an image to reduce the cost of the search. The search is first performed at a low resolution of the image. If no matches can be found at this resolution, then no matches can be found at a finer resolution. The computational savings are due to the fact that less computations of the least squares optimization are needed at a coarser resolution. For a more precise description let us introduce some notations. A grey value pyramid of an image f seen as a 2-D array is defined as the sequence of images $f^{(0)}, \ldots, f^{(r)}$, where $f^{(r)} = f$ and

$$f^{(k)}(i,j) = \frac{1}{4} \sum_{m,l=0}^{1} f^{(k+1)}(2i+m, 2j+l)$$

for $k = 0, \ldots, r-1$ and $0 \leq i,j < 2^k$. Similarly, one can obtain range blocks and domain blocks at resolution k from those at resolution $k+1$. The basic result in [Dekk95a] can be stated as follows.

Theorem 6 *Let $R^{(k)}$ and $D^{(k)}$ be respectively a range block and a codebook block at resolution k. Then $E(D^{(k+1)}, R^{(k+1)}) \geq E(D^{(k)}, R^{(k)})$.*

However, since not all domains at resolution $k+1$ have corresponding domains at resolution k, applying the theorem as stated above will take into consideration only domains of resolution $k+1$ who have their corners at positions $(2i, 2j)$. To circumvent this problem, one may consider a pyramid tree, where every resolution $k+1$ domain at (i,j) has four resolution k domain children at $(2i,2j), (2i+1,2j), (2i,2j+1), (2i+1,2j+1)$. It is also remarked that one cannot discard a $k+1$ resolution domain simply because its k resolution children has a scaling factor s_k such that $|s_k| > 1$. Actually one may find cases where $|s_k| > 1$ but $s_{k+1} = 0$.

Another method, related to the multiresolution approach, is the dimension reduction presented by Caso, Obrador and Kuo in [CaObKu95]. An incremental procedure at the pixel level has been given by Bani-Eqbal in [Bani94].

3.7 Fast search via fast convolution

Most of the techniques discussed above are *lossy* in the sense that they trade in a speedup for some loss in image fidelity. In contrast, with a *lossless* method the codebook block with the minimal (collage) error is obtained rather than an acceptable but suboptimal one. The method presented in this section is the first one that takes advantage of the fact that the codebook blocks, taken from the image, are usually overlapping. The fast convolution — based on the convolution theorem and carried out in the frequency domain — is ideally suited to exploit this sort of codebook coherence. The essential part of the basic computation in fractal image compression is a certain convolution [Saup96b, SaHar96a]. To see that denote by $\langle \cdot, \cdot \rangle$ the inner product in a Euclidean space of dimension n (= number of pixels in a range block). For a range block R and codebook block D the optimal coefficients are

$$s = \frac{n\langle D, R\rangle - \langle D, \mathbf{1}\rangle \langle R, \mathbf{1}\rangle}{n\langle D, D\rangle - \langle D, \mathbf{1}\rangle^2}, \quad o = \frac{1}{n}\left(\langle R, \mathbf{1}\rangle - s\langle D, \mathbf{1}\rangle\right).$$

For any (s, o) the error $E(D, R)$ can be regarded as a function of $\langle D, R\rangle, \langle D, D\rangle$, $\langle D, \mathbf{1}\rangle$, $\langle R, R\rangle$, and $\langle R, \mathbf{1}\rangle$. Its evaluation requires 23 floating point operations. Typically, the computations are organized in two nested loops:

- Global preprocessing: compute $\langle D, D\rangle$, $\langle D, \mathbf{1}\rangle$ for all codebook blocks D.
- For each range R do:
  - Local preprocessing: compute $\langle R, R\rangle, \langle R, \mathbf{1}\rangle$.
  - For all codebook blocks D do:
    - Compute $\langle D, R\rangle$ and $E(D, R)$.

The calculation of the inner products $\langle D, R\rangle$ dominates the computational cost in the encoding. The codebook blocks D are typically defined by downfiltering the image to half its resolution. Any subblock in the downfiltered image, that has the same shape as the range, can be considered a codebook block for that range. In this setting the inner products $\langle D, R\rangle$ are nothing but the *finite impulse response* (FIR) of the downfiltered image with respect to the range. In other words, the convolution (or, more precisely, the cross-correlation) of the range R with the downfiltered image is required. This discrete two-dimensional convolution can be carried out more efficiently in the frequency domain when the range block is not too small (convolution theorem). This procedure takes the inner product calculation out of the inner loop and places it into the local preprocessing where the inner products $\langle D, R\rangle$ for *all* codebook blocks D are obtained in one batch by means of fast Fourier transform convolution. Clearly, the method is lossless.

Moreover, the global preprocessing requires a substantial amount of time, but can be accelerated by the same convolution technique. The products $\langle D, \mathbf{1}\rangle$ are obtained by convolution of the downfiltered image with a range block where all intensities are set equal (called *range shape matrix*). The sum of the squares is computed in the same way where all intensities in the downfiltered image are squared before the convolution.

3.8 Fractal image compression without searching

Complexity reduction methods that are somewhat different in character are based on reducing the domain pool rigorously to a small subset of all possible domains.

For example, in the work that followed Monro and Dudbridge [MoDu92a] for each range the codebook block to be used to cover the range is uniquely predetermined to be a specific block that contains the range block [WoMo95]. A similar idea has been pursued by Hürtgen and Stiller [HuSt93] where the search area for a domain is restricted to a neighborhood of the current range. Additionally, a few sparsely spaced domains far from the range are taken into account as an option. Iterated Systems, Inc., seems also to prefer a local searching [GeLu96].

In [Saup96a] we considered a parametrized and non-adaptive version of domain pool reduction by allowing an adjustable number of domains to be excluded (ranging from 0% to almost 100%) and investigated the effects on computation time, image fidelity and compression ratio. We showed that there is no need for keeping domains with low intensity variance in the pool. Eliminating a fraction $1-\alpha$, $\alpha \in (0,1]$, of the domain pool consisting of the domains with least variance yields a lean and more productive domain pool. Using the adaptive quadtree method of Fisher [Fish94a, Appendix A] we showed the following:

1. The computation time scales linearly with α.
2. Even for low values of α, e.g., $\alpha = 0.15$, there is no degradation in image quality. On the contrary, the fidelity improves slightly.

Signes [Sign95] and Kominek [Komi95b] pursue similar ideas for domain pool reduction. An adaptive version of spatial search based on optimizing the rate-distortion performance is presented in [BSVN94].

4 More advanced issues

4.1 The partitioning

The partitionings mentioned so far are the fixed block size approach and the quadtree scheme. Of course, there are many other methods for partitioning the image support. What characterizes a good partition for fractal image compression? It should divide the image in regions that show similarity to other areas of the image. The fractal code consists of the partition information and of the transform coefficient information. We will only accept the higher coding costs of irregular partitions if those partitions lead to a better quality in terms of rate distortion curves. While using fixed block size image tilings, e.g. with squares, triangles or rectangles, our partition costs are zero and all information is in the transform part. The weakness of those partitions is their non-adaptivity to the image content. The opposite approach would be represented by contour coding: the lion's share of information is given by the partition code. As you may guess, the optimal approach will lie somewhere in the middle of those extremes. The methods used can be classified as hierarchical partitionings and split-and-merge methods.

Hierarchical partitionings:

- The **quadtree scheme** [BeDeKe92, JaFiBo92] can be considered as a first step towards adaptivity. If for a given square range there is no domain which fits well, the range is divided into four equally sized subsquares. This is done recursively with some given bounds for the minimum and maximum range size. Note what has changed: the partitioning is depending on the search result and there are different range sizes. During the procedure some ranges will be rejected and subdivided, thus, rendering the corresponding search void. Therefore, the maximal range size must not be chosen too large in order to avoid a large number of useless searches. The cost for storing the partition information is small. It amounts to a quadtree describing the splitting structure.

- In **HV (horizontal-vertical) partitioning** [FiMe94] the image is segmented into rectangles (see Figure 13). If for a given rectangular range block no acceptable domain match is found, the block is split into two rectangles either by a horizontal or a vertical cut. The splitting is based on block uniformity and also incorporates a rectangle degeneration prevention mechanism. For the range $R = (r_{ij})_{0 \le i < N, 0 \le j < M}$, the biased differences of vertical and horizontal pixel intensity sums, respectively, are computed:

$$h_j = \frac{\min(j, M-j-1)}{M-1} \left(\sum_i r_{i,j} - \sum_i r_{i,j+1} \right),$$

$$v_i = \frac{\min(i, N-i-1)}{N-1} \left(\sum_j r_{i,j} - \sum_j r_{i+1,j} \right).$$

  The maximal value of these differences determines splitting direction and position. A decision tree containing this information has to be stored.

  The resulting number of different range shapes leads to a higher time complexity. In spite of the higher cost for storing the partition information, the simulation results show a considerable rate-distortion improvement over the quadtree scheme.

- A further step in adaptivity is **polygonal partitioning** [Reus94b]. Based on the work of Wu and Yao[17] this is actually similar to HV partitioning including in addition 45^o and 135^o cutting directions.

Split and merge:

- Davoine et al. [DaBeCh93, DaCh94] advocate the use of **Delaunay triangulations** as partitioning method. The advantage of triangulations is the unconstrained orientation of edges. The Delaunay triangulation is the triangulation

[17] X. Wu and C. Yao, *Image coding by adaptive tree-structured segmentation,* in: *Proceedings DCC'91 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Comp. Soc. Press, 1991.

Figure 12: A partition with 1000 ranges gained via evolutionary coding.

Figure 13: An HV partition with 1000 range blocks.

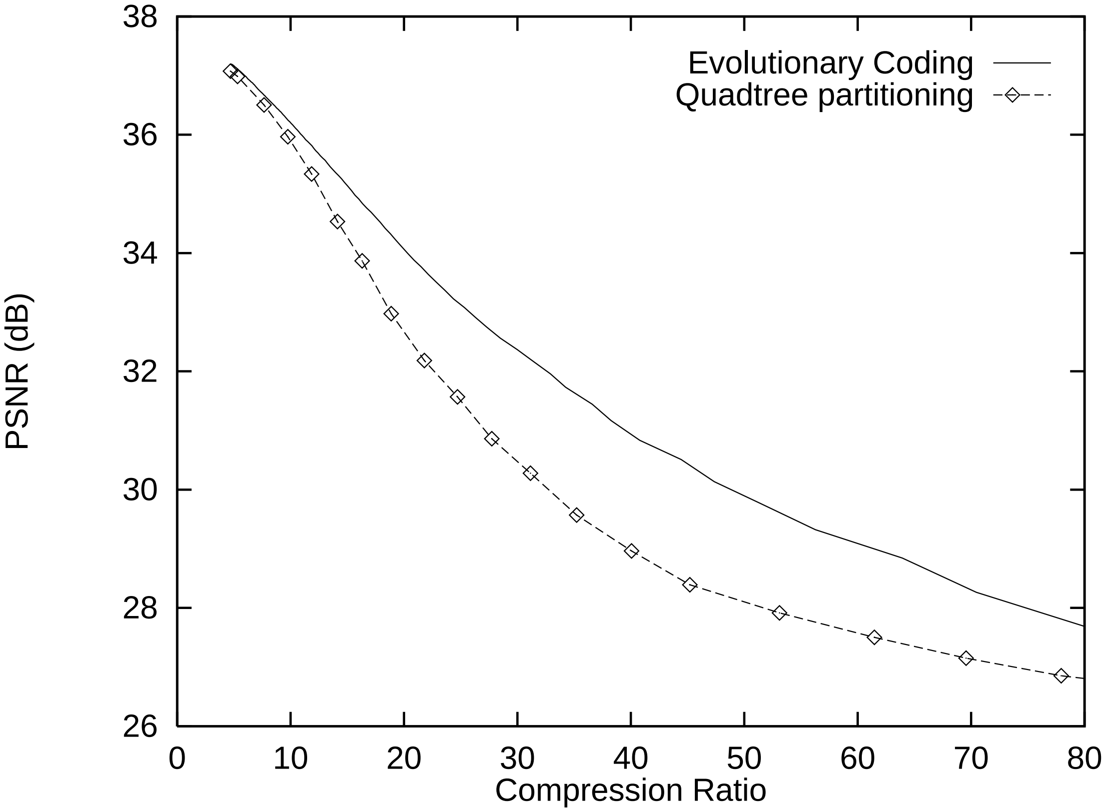

Figure 14: Rate distortion performance of the evolutionary and the quadtree methods using Lenna.

which maximizes the minimal interior angle. This imposes some regularity.

The method works as follows: starting with a Delaunay triangulation of a set of regular distributed points, the partition is refined by splitting non-uniform triangles (as measured by standard deviation). This splitting step is performed by adding an additional point at the barycenter and recomputing the Delaunay triangulation for this new set of points. The splitting is stopped via a uniformity criterion. In the merging pass, a vertex p is removed if all triangles with vertex p have approximately the same mean value, and again the Delaunay triangulation of this new set of points is computed. In another paper [DaSvCh95] the authors also allow the merging of two triangles when the resulting quadrilateral is convex and both triangles have more or less the same grey value distribution.

- The **region-based fractal coder using heuristic search** of Thomas and Deravi [ThDe95] is another split and merge approach. First, the image is split in atomic square blocks, e.g., of size 4×4 or 8×8. Then neighboring blocks are merged successively to build larger ranges of irregular shapes. Since one ends up with only a few large ranges there are only a few transformations to store. But the large sizes and the irregular shapes of the ranges prohibit the conventional best domain search, therefore a heuristic strategy has to be employed. Thomas and Deravi give three methods differing in the level of sophistication. With the simple algorithm, for a seed atomic block an optimal domain match is searched. Then it is checked whether this transformation can be extended to a block neighboring the seed range. This extension step is stopped by a distortion criterion. Another seed is selected and the procedure goes on until the whole image is coded. This algorithm is then improved by some updating procedures and built-in competition between ranges.

- Another adaptive partitioning method using **evolutionary computation** is presented in [SaRu96a]. Here, for a fixed size square block partition a fractal code is sought as in standard fractal coding, but for each range the best d codebook entries are kept in a list together with the optimal scaling and offset parameters. We take N times this configuration as the starting population for the evolution. The offspring are built by randomly merging two neighboring blocks; the fractal code is modified by only considering the transformations kept in the lists of those two blocks. A selection is performed by only keeping the fittest configurations in terms of collage error. A comparison of this method and the quadtree scheme is given in Figure 14. Figure 12 shows a partition in which the image support is split into 1000 ranges.

4.2 The block transformation

In fractal image compression, the coding of an image f consists of finding a contractive mapping T[18] whose fixed point $g = Tg$ is the best possible approximation of f. The collage theorem [Barn88b] states that by minimizing the distance between f and Tf,

[18] The contractivity of T is only a sufficient condition. A more general condition is the eventual contractivity of T.

it is expected to minimize the distance between the fixed point g and the image f. When choosing the mapping T one should keep in mind the following constraints [Oien94]

- *T should not be linear, otherwise its fixed point is the zero image.*
- *T should be computationally and structurally simple, in order to provide simple collage optimization, fast decoding, and simple analysis.*
- *The fixed point of T should be robust with respect to the quantization of its parameters.*

In his original approach, Jacquin coded each range block by a linear combination of one codebook block and one block of fixed intensity. It can be easily shown that in this case the mapping T is affine, i.e, $Tf = Af + b$, where A is an $N \times N$ matrix and $b \in \mathbf{R}^N$. Here N is the total number of pixels in the image. In this section we will describe more general mappings proposed in the literature. For the sake of clarity, we will take the following approach. Since an image is equal to the union of the ranges, a mapping T will be defined implicitly by specifying its action on each single range.

Let $R = (r_1, \ldots, r_n)^T$ be a range block. Let $D_i = (d_1^i, \ldots, d_n^i)^T$, $i = 1, \ldots, N_D$ be a codebook block. A more general formulation for the least squares problem (3) is

$$\min_{(x_1,\ldots,x_m)^T \in \mathbf{R}^m} \sum_{k=1}^{n} \{r_k - t_k(x_1, \ldots, x_m)\}^2 \tag{14}$$

where $t_k(x_1, \ldots, x_m) = t(d_k^1, \ldots, d_k^{N_D}; x_1, \ldots, x_m)$. Thus in our formulation, the collage of the range block R is the block

$$\begin{pmatrix} t_1(x_1, \ldots, x_m) \\ \vdots \\ t_n(x_1, \ldots, x_m) \end{pmatrix}$$

If all functions t_k have continuous partial derivatives with respect to all x_i, then a necessary condition for $x = (x_1, \ldots, x_m)^T$ to solve (14) is

$$\frac{\partial}{\partial x_j} \sum_{k=1}^{n} \{r_k - t_k(x_1, \ldots, x_m)\}^2 = 0,\ j = 1, \ldots, m \tag{15}$$

The mapping T has an affine form if there exists an $n \times m$ matrix M such that

$$\begin{pmatrix} t_1(x_1, \ldots, x_m) \\ \vdots \\ t_n(x_1, \ldots, x_m) \end{pmatrix} = Mx \tag{16}$$

For example, the fixed size baseline encoder of Section 2.1.5 is the simple case

$$M = \begin{pmatrix} d_1 & 1 \\ \vdots & \vdots \\ d_n & 1 \end{pmatrix}$$

A modified version is the encoder introduced by Øien [Oien93, OiLe94a]. By subtracting the mean $\frac{d_1+\ldots+d_n}{n}$ from each coefficient d_k in the above matrix, it is shown that for a special choice of the domain pool, one can obtain a fast decoder (see the section on fast decoding).

Utilizing several fixed blocks $B_1, \ldots, B_p$ has been suggested by many researchers [OiLeRa91, Monr93a, Monr93b]. It corresponds to the matrix

$$M = \begin{pmatrix} d_1 & b_1^1 & \ldots & b_p^1 \\ \vdots & \vdots & \vdots & \vdots \\ d_n & b_1^n & \ldots & b_p^n \end{pmatrix}.$$

Other attempts consisted of using several codebook blocks $D_{i_1}, \ldots, D_{i_l}$ [GhHu93a, GhHu94a, GhHu94b, Vine94]. In this case the matrix M has the structure

$$M = \begin{pmatrix} d_1^{i_1} & \ldots & d_1^{i_l} & b_1^1 & \ldots & b_p^1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ d_n^{i_1} & \ldots & d_n^{i_l} & b_1^n & \ldots & b_p^n \end{pmatrix}.$$

One may as well use the square of the codebook block coefficients. With this setting one gets

$$M = \begin{pmatrix} d_1^2 & d_1 & 1 \\ \vdots & \vdots & \vdots \\ d_n^2 & d_n & 1 \end{pmatrix}.$$

All these variants aim at providing a tighter collage for the given range block. Unfortunately, they suffer from longer encoding times. Furthermore, the code for a range block is clearly more expensive than in the baseline encoder. For the case of several codebook blocks, there is an additional complication in ensuring the contractivity of the mapping T [GhHu94b]. Nevertheless, as long as the linearity condition (16) holds, we have the following result.

Theorem 7 *The minimization problem (14) has at least one solution x_0. Moreover, if the columns of the matrix M are independent, then x_0 is unique and it is given by*

$$x_0 = (M^T M)^{-1} M^T R.$$

The case where the linearity condition is not assumed has not yet been sufficiently explored. Lin and Venetsanopoulos [LiVe94a, LiVe94b] used a scheme where $m = 4$ and

$$t_k(x_1, \ldots, x_4) = \pm \frac{1}{1 + e^{x_1 k_1 + x_2 k_2 + x_3}} + x_4.$$

Here (k_1, k_2) is the 2-D representation of k (remember that we converted the square block into a vector). At a fixed bit rate, the authors report a visually better decoded image and a faster decoding. However, they concede difficulties in the solving of the least squares problem (14).

4.3 Color and video

Very little work has been published on color fractal image compression. This may be due to the fact that encoding color images can be considered as a straightforward extension of the encoding of monochrome images. For example, Fisher [Fish94a] recommends not to encode the RGB components individually. It is advised to determine the YIQ values first. Then, each YIQ channel can be encoded separately, the I- and Q-channels being encoded at a lower bit rate than the Y-channel. However, in [Bogd95a], the green (G) component is encoded individually and it is then used to predict the other components.

For fractal compression of image sequences there are two main approaches. The simplest one is to separately encode each 2-D frame or only a section of it by a fast fractal coder [MoNi95]. A variant is to take profit of domain blocks from previous frames [FiRoSh94]. The second technique is to consider time as a third variable and to apply fractal coding to the 3-D range and domain blocks [LiNoFo93, BaVo95]. Of course hybrid methods are also possible. An overview on fractal video coding can be found in [GhHu96b].

4.4 Wavelets and fractal image compression

In fractal coding usually a square block of size $2^r \times 2^r$ is approximated by another image block of size $2^{r+1} \times 2^{r+1}$ under an affine mapping. Thus, one tries to find similar structures at two different scales (this can be expressed as a two-scale difference equation as pointed out, e.g., in [Bogd94b]). Since fractals have the property of self-similarity at different scales, it is natural to use multiresolution methods for an analysis of fractal coding. The first approach in this direction was done by Baharav et al. in [BaMaKa93, BaMaKa94]. Let us explain their ideas briefly by an example. For a given 512×512 grey scale image, partitioned into non-overlapping 16×16 blocks, a fractal code C is determined in the standard way, considering the domain pool of nonoverlapping 32×32 blocks. C contains the information of $32 \cdot 32$ transformations. In the decoding, C is used to compute the attractor A_1 of size 512×512. But C can also be iteratively applied to an arbitrary 256×256 image, partitioned into 8×8 blocks, gaining an attractor A_2, or to an 128×128 image, partitioned into 4×4 blocks, giving an attractor A_3, and so on. Thus, one ends up with a pyramid $A_1, A_2, ..., A_5$, describing different resolutions of the attractor A_1. The relationship between those layers can be easily understood, e.g., using the Haar discrete wavelet transform.

Such explicit formulations of fractal coding by means of wavelet analysis are given in the papers of Davis [Davi95a], van de Walle [Wall95], Krupnik et al. [KrMaKa95] and Simon [Simo95a, Simo95b]. The earlier published paper [RiCa94] of Rinaldo and Calvagno also contains the main ideas used for combining wavelets and fractal coding.

After applying the Haar transform, a range or a domain is given by the block mean and a *wavelet subtree*, as depicted in Figure 15. Averaging and subsampling of the domain block essentially translate into truncating the domain wavelet subtree by cutting off the leaves. For symmetrical or antisymmetrical wavelets, isometry operations are easily incorporated. Thus, in fractal coding a scaled version of the

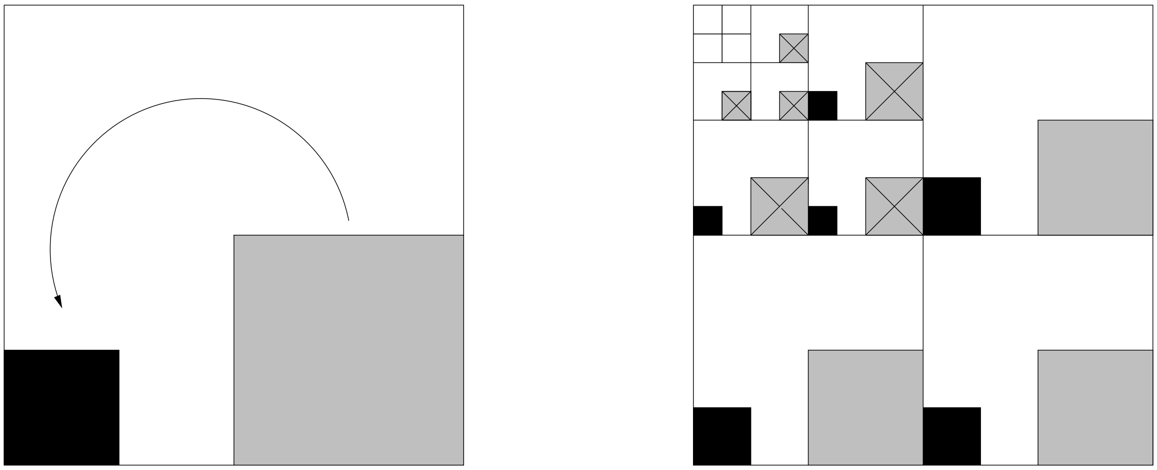

Figure 15: Left: 4×4 range (black), 8×8 domain (shaded). Right: the corresponding wavelet subtrees; the crossed shaded parts of the domain wavelet subtree can be used for a prediction of the range wavelet subtree.

truncated domain tree is used as a prediction for a range wavelet subtree. Note that the subtree gives the dynamic part of a block; the mean value has to be stored separately (see Øien's orthogonalization method). The orthogonality of the wavelet transform allows the computation of the scaling parameters in the wavelet domain (when using the l^2 norm).

With the knowledge of the range block means and the tree transformation parameters, decoding is performed by predicting higher frequency coefficients by lower frequency coefficients. The number of octaves decoded in this manner determines the resulting attractor resolution.

The Haar wavelet is used to demonstrate the mechanisms of fractal coding in time-frequency space. The use of higher order wavelets gives visually much better results, since there is no strict blocking of the ranges. In other words, with higher order wavelets one works with overlapped partitions. The elimination of the tiling effects is one of the main features of the combined fractal wavelet approach.

Another main advantage can be seen in Davis' *self-quantization of subtree* (SQS) scheme. Here, fractal methods in the wavelet domain are combined with zerotree coding, scalar coding and a clever way of using the various schemes optimally. Comparing SQS to Shapiro's embedded zerotree wavelet coder[19], it is no surprise that the SQS scheme achieves competitive compression ratio results.

4.5 Hybrid methods: entropy constrained frequency-domain encoding

In their papers [BaVo94, BSVN94, Bart95], Barthel et al. have introduced several improvements for fractal coding that led to impressive results (for the 512×512

[19] J. Shapiro, *Embedded image coding using zerotrees of wavelet coefficients,* IEEE Trans. on Signal Processing 41,12 (1993) 3445–3462.

Lenna image a PSNR of 30 dB at a compression ratio of 80:1 is reported). Here we will restrict our attention to the modified value (luminance) transformations. Let $\hat{D}$ be the 2-dimensional Fourier transform of an image block $D \in \mathbf{R}^n$; $\hat{d}_i$ should refer to the i-th coefficient in the zig-zag scanned transform block $\hat{D}$. The range-domain correspondence $R = sD + o\mathbf{1}$, $s, o \in \mathbf{R}$, when represented in frequency domain translates into

$$\begin{bmatrix} \hat{r}_1 \\ \hat{r}_2 \\ \vdots \\ \hat{r}_n \end{bmatrix} = \begin{bmatrix} s \cdot \hat{d}_1 \\ s \cdot \hat{d}_2 \\ \vdots \\ s \cdot \hat{d}_n \end{bmatrix} + \begin{bmatrix} o \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

To decorrelate the s and o parameters, this is changed to

$$\begin{bmatrix} \hat{r}_1 \\ \hat{r}_2 \\ \vdots \\ \hat{r}_n \end{bmatrix} = \begin{bmatrix} s_1 \cdot \hat{d}_1 \\ s \cdot \hat{d}_2 \\ \vdots \\ s \cdot \hat{d}_n \end{bmatrix} + \begin{bmatrix} o \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

where s_1 is a fixed parameter in $[0, 1)$. This is called a modified 1st order (luminance) transform. In the case of $s_1 = 0$ it leads to Øien's orthogonalization method. Barthel recommends the use of $s_1 = 0.5$.

The optimal coefficients in the least square sense are given by

$$o_{opt} = \hat{r}_1 - s_1 \hat{d}_1$$

$$s_{opt} = \frac{\sum_{i=2}^{n} \hat{d}_i \hat{r}_i}{\sum_{i=2}^{n} \hat{d}_i^2}$$

We get an even better match between a range and a domain by using the following higher order transforms. If a spectral coefficient $\hat{r}_i$ is not well approximated by $s_{opt} \cdot \hat{d}_i$, the value $\hat{r}_i$ is coded separately:

$$\begin{bmatrix} \hat{r}_1 \\ \hat{r}_2 \\ \vdots \\ \hat{r}_i \\ \vdots \\ \hat{r}_n \end{bmatrix} = \begin{bmatrix} s_1 \cdot \hat{d}_1 \\ s \cdot \hat{d}_2 \\ \vdots \\ 0 \\ \vdots \\ s \cdot \hat{d}_n \end{bmatrix} + \begin{bmatrix} o \\ 0 \\ \vdots \\ t \\ \vdots \\ 0 \end{bmatrix}$$

This is called a 2nd order transform; correspondingly, by changing m spectral coefficients one gets an $m + 1$ order transform.

Thus, by allowing this wider class of transforms, we are able to get better approximations. Since there are many more choices, the optimization becomes difficult. The proposed optimization strategy is based on entropy constrained code refinement:

1. **Initialization:** find a fractal code by only considering the class of modified first order transforms.

2. **Refinement:** change that block code which gives the biggest decrease in distortion (MSE) at lowest (bitrate) cost by increasing the order of the transform by one.

3. Stop at a given MSE or rate level

This is the global strategy. But how do we find, e.g., the best 2nd order transform for a given block? Since the optimal solution to this problem is too expensive to compute, the recommendation is to use the following greedy strategy: change the spectral coefficient which is responsible for the highest error component in the collage error.

4.6 Hybrid methods: VQ-enhanced fractal image compression

Though evoked by some authors, combining fractal coding with vector quantization (VQ) has not been deeply investigated. In [Jacq93] it is only suggested that fractal coding should be employed for sharp-edge blocks, whereas vector quantization is more advantageous for other blocks. Gharavi-Alkhansari and Huang [GhHu94b] claim that vector quantization can be seen as a special case of their generalized transform. An interesting study was presented in [Leps93, RaLe93] where the performance of a fractal image coder and a product code vector quantizer have been compared.

In [HaMuSa96a, HaMuSa96b] we investigate how to take advantage of a vector quantization codebook in order to enhance the performance of a fractal image coder.

First, a set of fixed cluster centers is designed as explained in [Hamz95] (see Section 2.3.3.4). Then these cluster centers are normalized. The new cluster centers can be considered as an integral part not only of the encoder but also of the decoder. The hybrid scheme works as follows. If the least squares approximation of a range block by an affine transformation of its nearest cluster center m_c is "good enough", then the cluster center will serve as a VQ codebook block. Otherwise, the range block will be encoded by a domain block. The requirement "good enough" can be for example the fulfilment of one of the two conditions:

$$\frac{1}{\sqrt{n}} E(R, m_c) \leq \delta \tag{17}$$

or

$$E(R, m_c) \leq (1+\epsilon) E(R, D) \tag{18}$$

for all codebook blocks D in the cluster with center m_c. Here ϵ and δ are parameters of our method. In this way, the bit rates can be improved by a clever choice of the ratio of the number of cluster centers to the number of domain blocks used in the fractal code. For example, if we denote by N_R the number of range blocks and by N_1 the number of range blocks VQ encoded, then our hybrid scheme will improve the rate of the fractal coder if $N_1 > \frac{N_R}{p-k}$, where 2^p is the number of domain blocks and 2^k is the number of cluster centers. In the above computation, one bit per range has been included to specify the way a range block has been encoded. Furthermore, the new scheme reduces the complexity of the already fast algorithm described in [Hamz95] since the search for a matching codebook block is only started if the cluster center was not able to provide an acceptable approximation. As discussed in [Hamz95], the search for a matching codebook block can be extended to neighboring clusters. Note that for VQ-encoded range blocks no contractivity condition on the scaling factor is required. Moreover, the offset of a VQ-encoded range block reduces to its DC value. The decoding proceeds as with a conventional fractal decoder, i.e, through iterations from any initial image with the advantage, however, that the reconstruction of the VQ-encoded range regions is already obtained after the first iteration. Thus, in addition to a less complex decoder, we expect to obtain a faster convergence.

Our experimental results showed that the hybrid scheme was able to improve the performance of the conventional fractal coder in all its aspects. The rate-distortion curve was ameliorated, and both the encoding and the decoding were faster.

4.7 Fast decoding

One of the most remarkable features of fractal image compression is the simplicity of the decoder. The reconstruction of the image is obtained by iterating the mapping T on any initial image f_0. Since the mapping T is contractive, the contraction mapping principle ensures the convergence of the sequence of iterates $\{T^k(f_0)\}$ to the fixed point g. Typically, the baseline decoder needs less than 10 iterations to converge. However, for applications where the speed of the decoding is vital (e.g., in real-time video), one may wish to find faster methods.

4.7.1 Fast decoding with orthogonalization

Øien's encoding scheme [Oien93] requires a codebook where each domain block consists of a union of range blocks. However, its impressive aspects fully justify this restriction. Some of these are:

- a convergence of the decoding in a finite number of iterations without any constraints on the scaling factors; this number depends only on the domain and range sizes.
- a convergence at least as fast as in the conventional scheme.
- a pyramid-structured decoding algorithm with a low computational complexity.

4.7.2 Hierarchical decoding

In [BaMaKa93, BaMaKa94] Baharav et al. proposed a fast decoding algorithm based on a hierarchical interpretation of the PIFS-code. Essentially the method prescribes

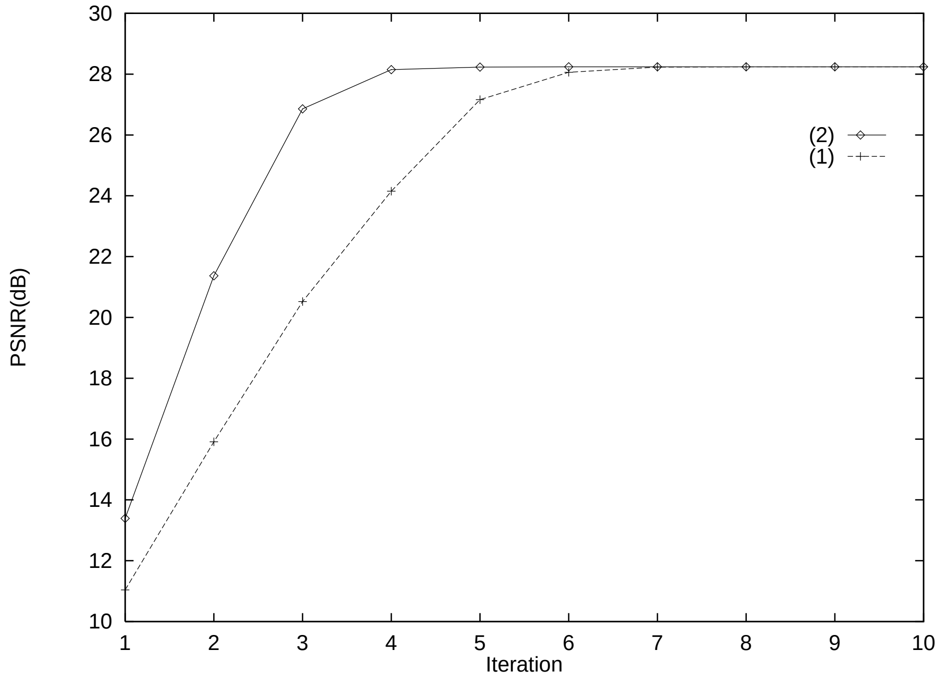

Figure 16: PSNR vs. iteration step for the 512 $\times$ 512 Lenna image. The image was encoded with Fisher's quadtree code with the default parameters (three-level quadtree, one-class search). Curve (1) corresponds to the conventional decoding, curve (2) to the method with codebook update.

the usual iteration for the decoding, however, with the modification, that the iterations are carried out at a coarse resolution of the image. Once the fixed point of the PIFS at the coarse resolution is reached, a deterministic algorithm is used to find the fixed point at any higher resolution (compare Section 4.4). The savings in computation are due to the fact that the iterations are applied to a vector of low dimension.

4.7.3 Codebook update

This method introduced in [Hamz96a] works in the spirit of the Gauss-Seidel method. Each time a new range block is computed, the domain blocks used in the decoding and covering these range blocks are updated. In [Hamz96b] we prove the convergence of the decoding if the scaling factors are less than one. Experimental results show that our method converges faster than the conventional procedure (see Figure 16).

4.7.4 Other methods

The simple scheme of Monro and Dudbridge [MoDu92b, MoDu95, Dudb94] has a fast noniterative decoding algorithm giving an exact reconstruction of the fixed point. In [DoVa95] the dependency between domain blocks and range blocks is analyzed. As a consequence it is shown that the decoding can be made faster by reconstructing in a noniterative way some of the range blocks.

5 WEB resources

The increasing interest in fractal image compression has led to the creation of many World Wide Web resources dedicated to this field. The following is a list of some of the most important ones.

- Yuval Fisher's site at http://inls.ucsd.edu/y/Fractals/ contains valuable information on bibliographies, books, conferences, announcements, internet resources, papers and software. A C quadtree code capable of encoding images in a few seconds, decoding at arbitrary resolution, and achieving high compression ratios is also available.
- Iterated Systems, Inc at http://www.iterated.com/ offers commercial software on video and still image compression.
- The University of Bath Image Processing Group at http://dmsun4.bath.ac.uk/ has a demonstration video decoder based on the Bath fractal transform.
- The Waterloo Montreal Verona fractal research initiative at http://links.uwaterloo.ca/ is designed to further the theoretical understanding of the mathematics of fractals and its application to signal processing. It contains repositories of fractal compression software and papers. Results of various compression schemes are compared against a 32 element suite.
- The Groupe Fractales site at http://www-syntim.inria.fr/fractales/ is mainly consecrated to fractal analysis.
- Brendt Wohlberg from the University of Cape Town has a BibTex format bibliography in http://dip1.ee.uct.ac.za/fractal.bib.html/.
- John Hart's home page at http://www.eecs.wsu.edu/∼hart has many interesting links to fractal compression stuff.

Our University of Freiburg ftp site at

ftp://ftp.Informatik.Uni-Freiburg.DE/documents/papers/fractal/

contains papers, software and a regularly updated bibliography.

The first international meeting dedicated to fractal image encoding and analysis was held in Trondheim in Norway in July 1995. A Web site of this meeting, a NATO ASI, is in http://inls.ucsd.edu/y/ASI/.

Fractal coding is also discussed in the newsgroups comp.compression, comp.compression.research and sci.fractals.

Acknowledgments. The authors thank Bertram Ganz, Luigi Grandi and Matthias Ruhl for producing most of the figures. Also we thank Yuval Fisher for making his C-code for the fractal quadtree encoding available on the internet. It surely served as a catalyst for our research and that of many others as well.

References

[NOTE-THIS] The papers marked with an asterisk are electronically available from our ftp-server at ftp://ftp.Informatik.Uni-Freiburg.DE/papers/fractal/.

[AlCl91] ∗Ali, M., Clarkson, T.G., *Fractal image compression,* in: *Proc. 1st Seminar on Information Technology and its Applications (ITA'91),* Leicester, Sept. 1991.

[AlCl93] Ali, M., Clarkson, T.G., *Using linear fractal interpolation functions to compress static and motion images,* in: *Proc. 3rd Conf. on Information Technology and its Applications (ITA'94),* Leicester, April 1994.

[AlCl94] ∗Ali, M., Clarkson, T., *Using linear fractal interpolation functions to compress video images,* Fractals 2,3 (1994) 417–421.

[AlGeCl93] ∗Ali, M., Gennert, M.A., Clarkson, T.G., *Analysis, generation and compression of pavement distress images using fractals,* in: *The Applications of Fractals and Chaos,* A.J. Crilly, R.A. Earnshaw and H. Jones (eds.), Springer-Verlag, Berlin, 1993.

[AlPaCl92] ∗Ali, M., Papadopoulos, C., Clarkson, T., *The use of fractal theory in a video compression system,* in: *Proceedings DCC'92 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Comp. Soc. Press, 1992.

[Anso93] Anson, L., *Fractal image compression,* BYTE Magazine, Oct. 1993.

[Apiki93] Apiki, S., *Compressing with fractals,* BYTE Magazine, Special Issue Spring 1993.

[ArBaMu94] Arneodo, A., Bacry, E., Muzy, J. F., *Solving the inverse fractal problem from wavelet analysis,* Europhysics Letters 25,7 (1994) 479–484.

[BaMaKa93] ∗Baharav, Z., Malah, D., Karnin, E., *Hierarchical interpretation of fractal image coding and its applications to fast decoding,* in: *Intl. Conf. on Digital Signal Processing,* Cyprus, July 1993.

[BaMaKa94] Baharav, Z., Malah, D., Karnin, E., *Hierarchical interpretation of fractal image coding and its applications,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[Bani94] ∗Bani-Eqbal, B., *Speeding up fractal image compression,* in: *Proceedings from IS&T/SPIE 1995 Symposium on Electronic Imaging: Science & Technology,* Vol. 2418: Still-Image Compression 1995.

[BaDu96] Barakat, M., Dugelay, J.-L., *Image sequence coding using 3-D I.F.S.,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[Barn88a] Barnsley, M., *Fractal modelling of real world images,* in: *The Science of Fractal Images,* H.-O. Peitgen and D. Saupe (eds.), Springer-Verlag, New York, 1988.

[Barn88b] Barnsley, M., *Fractals Everywhere,* Academic Press, San Diego, 1988.

[BaHu93] Barnsley, M., Hurd, L., *Fractal Image Compression,* AK Peters, Wellesley, 1993.

[BaHuGu92] Barnsley, M., Hurd, L., Gustavus, M., *Fractal video compression,* in: *Proceedings of the Thirty-Seventh IEEE Computer Society International Conference,* Vol. 37, pp. 41–42, 1992.

[BaJa88] Barnsley, M. F., Jacquin, A. E., *Applications of recurrent iterated function systems to images,* in: *Proceedings from SPIE Visual Communications and Image Processing,* Vol. 1001, pp. 122–131, 1988.

[BJMRS88] Barnsley, M. F., Jacquin, A. E., Malassenet, F., Reuter, L., Sloan, A., *Harnessing chaos for image synthesis,* Computer Graphics 22,4 (1988) 131–140.

[BaSl87] Barnsley, M. F., Sloan., A. D., *Chaotic compression,* Computer Graphics World, Nov. 1987.

[BaSl88] Barnsley, M. F., Sloan., A. D., *A better way to compress images,* BYTE Magazine, Jan. 1988.

[BaSl90] Barnsley, M. F., Sloan, A., *Methods and apparatus for image compression by iterated function system,* United States Patent # 4,941,193.

[BaSl91] Barnsley, M. F., Sloan, A., *Method and apparatus for processing digital data,* United States Patent # 5,065,447.

[Bart95] *Barthel, K. U., *Entropy constrained fractal image coding,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[BSVN94] *Barthel, K. U., Schüttemeyer, J., Voyé, T., Noll, P., *A new image coding technique unifying fractal and transform coding,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[BaVo94] *Barthel, K. U., Voyé, T., *Adaptive fractal image coding in the frequency domain,* in: *Proceedings of International Workshop on Image Processing: Theory, Methodology, Systems and Applications,* Budapest, June 1994.

[BaVo95] *Barthel, K. U., Voyé, T., *Three-dimensional fractal video coding,* in: *Proc. ICIP-95 IEEE International Conference on Image Processing,* Washington, D.C., 1995.

[BaVo96] Barthel, K. U., Voyé, T., *Combining wavelet and fractal coding for 3-D video coding,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[BaVoNo93] *Barthel, K. U., Voyé, T., Noll, P., *Improved fractal image coding,* in: *Proceedings of the International Picture Coding Symposium PCS'93,* Lausanne, March 1993.

[Beau90] Beaumont, J. M., *Advances in block based fractal coding of still pictures,* in: *Proceedings IEE Colloquium: The application of fractal techniques in image processing,* pp. 3/1-3/6, 1990.

[Beau91] Beaumont, J. M., *Image data compression using fractal techniques,* British Telecom Technol. Journal 9,4 (1991) 93–109.

[BDBKS94] Bedford, T., Dekking, F. M., Breewer, M., Keane, M. S., van Schooneveld, D., *Fractal coding of monochrome images,* Signal Processing: Image Communication 6 (1994) 405–419.

[BeDeKe92] Bedford, T., Dekking, F. M., Keane, M. S., *Fractal image coding techniques and contraction operators,* Nieuw Arch. Wisk. (4) 10,3 (1992) 185–218.

[BBBP96] Belloulata, K., Baskurt, A., Benoit-Cattin, H., Prost, R., *Fractal coding of subbands using an oriented partition,* in: *Proceedings of the VIIIth European Signal Processing Conference EUSIPCO'96,* Trieste, Sept. 1996.

[BeMo95] Bethel, D. M., Monro, D. M., *Polynomial image coding with vector quantised error compensation,* in: *Proceedings of ICASSP-1995 IEEE International Conference on Acoustics, Speech and Signal Processing,* Detroit, 1995.

[BiFi94] Bielefeld, B., Fisher, Y., *A convergence model,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[Bogd94a] ∗Bogdan, A., *Multiscale (inter/intra-frame) fractal video coding,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[Bogd94b] Bogdan, A., *Multiscale fractal image coding and the two-scale difference equation,* Tech. Report CU/CTR/TR 358-94-05, Columbia University, March 1994.

[Bogd95a] Bogdan, A., *The Fractal pyramid with application to image coding,* in: *Proceedings of ICASSP-1995 IEEE International Conference on Acoustics, Speech and Signal Processing,* Detroit, 1995.

[Bogd95b] Bogdan, A., *Image coding using iterative transformations with applications to image communication,* PhD thesis, Columbia University, 1995.

[BoMe92] ∗Bogdan, A., Meadows, H., E., *Kohonen neural network for image coding based on iteration transformation theory,* in: *Proceedings from SPIE Neural and Stochastic Methods in Image and Signal Processing,* Vol. 1766, pp. 425–436, 1992.

[Bone95] Bone, D. J., *Orthonormal fractal image encoding using overlapping blocks,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[BoJa89] Boss, R. D., Jacobs, E. W., *Fractal-based image compression,* Technical Report 1315, Naval Ocean Systems Center, San Diego, Sept. 1989.

[BoJa91] Boss, R. D., Jacobs, E. W., *Studies of iterated transform image compression and its applications to color and DTED,* Technical Report 1468, Naval Ocean Systems Center, San Diego, 1991.

[BoJa94] 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.), Springer-Verlag, New York, 1994.

[CFMV93] Cabrelli, C. A., Forte, B., Molter, U. M., Vrscay, E. R., *Iterated fuzzy set systems: a new approach to the inverse problem for fractal and other sets,* J. Math. Anal. and Appl. 17,4 (1993) 465–486.

[CAAII94] Cai, D., Arisawa, T., Asai, N., Ikebe, Y., Itoh, T., *Fractal image compression using locally refined partitions,* Fractals 2,3 (1994) 405–408.

[Carl95] Carlini, P., *Self-organizing maps, vector quantization, and fractal image coding,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[CaKu95] Caso, G., Kuo, C.-C. J., *Multiresolution analysis of fractal image compression,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[CaObKu95] Caso, G., Obrador, P., Kuo, C.-C. J., *Fast methods for fractal image encoding,* in: *Proceedings from IS&T/SPIE 1995 Symposium on Electronic Imaging: Science & Technology,* Vol. 2501, pp. 583–594, 1995.

[ChKu96] Chang, H. T., Kuo, C. J., *Finite-state fractal block coding of images,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[ChDaBe93] ∗Chassery, J.-M., Davoine, F., Bertin, E., *Compression fractale par partitionnement de Delaunay,* in: *Proc. of 14th Conference GRETSI,* Vol. 2, pp. 819–822, Juan-les-Pins, Sept. 1993.

[ChZh95a] Cheng, B., Zhu, X., *A multiresolution approximation theory of fractal transform,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[ChZh95b] ∗Cheng, B., Zhu, X., *Multiresolution approximation of fractal transform,* to appear in Signal Processing.

[ChSh91] Cheung, K.-M., Shahshahani, M., *A comparison of the fractal and JPEG algorithms,* TDA Progress Report 42,107 (1991) 21–26.

[CoHaFl95] Cochran, W. O., Hart, J. C., Flynn, P., J., *Principal component classification for fractal volume compression,* in: *Proceedings of Western Computer Graphics Symposium,* March 1995.

[Cohe96] Cohen, H. A., *Thumbnail-based image coding utilizing the fractal transform,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[CuDu91] Culik II, K., Dube, S., *Using fractal geometry for image compression,* in: *Proceedings DCC'91 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Computer Society Press, 1991.

[CuDuRa93] Culik II, K., Dube, S., Rajcani, P., *Efficient compression of wavelet coefficients for smooth and fractal-like data,* in: *Proceedings DCC'93 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Comp. Soc. Press, 1993.

[CuKa94] Culik II, K., Kari, J., *Inference algorithms for WFA and image compression,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[Curi94] Curinga, S., *A proposal about fractal coding,* in: *Proceedings of the International Picture Coding Symposium PCS'94,* Sacramento, California, Sept. 1994.

[Davi95a] Davis, G., *Self-quantized wavelet subtrees: A wavelet-based theory for fractal image compression,* in: *Proceedings DCC'95 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Computer Society Press, March 1995.

[Davi95b] Davis, G., *Adaptive self-quantization of wavelet subtrees: A wavelet-basedtheory for fractal image compression,* in: *SPIE Conference on Mathematical Imaging: Wavelet Applications in Signal and Image Processing,* San Diego, July 1995.

[DaAnChBa96] Davoine, F., Antonini, M., Chassery, J.-M., Barlaud, M., *Fractal image compression based on Delaunay triangulation and vector quantization,* IEEE Transactions on Image Processing 5,2 (1996) 338–346.

[DaBeCh93] ∗Davoine, F., Bertin, E., Chassery, J.-M., *From rigidity to adaptive tessellation for fractal image compression: Comparative studies,* in: *Proc. of IEEE 8th Workshop on Image and Multidimensional Signal Processing,* pp. 56–57, Cannes, Sept. 1993.

[DaBeCh95] Davoine, F., Bertin, E., Chassery, J.-M., *An adaptive partition for fractal image coding,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[DaCh94] *Davoine, F., Chassery, J.-M., *Adaptive Delaunay triangulation for attractor image coding,* in: *Proc. of 12th International Conference on Pattern Recognition (ICPR),* Jerusalem, Oct. 1994.

[DaSvCh95] *Davoine, F., Svensson, J., Chassery, J.-M., *A mixed triangular and quadrilateral partition for fractal image coding,*in: *Proc. ICIP-95 IEEE International Conference on Image Processing,* Washington, D.C., 1995.

[Dekk95a] Dekking, F. M., *An inequality for pairs of martingales and its applications to fractal image coding,* Technical Report 95-10, Faculty of Technical Mathematics and Informatics, Delft Univeristy of Technology, 1995.

[Dekk95b] Dekking, F. M., *Fractal image coding: some mathematical remarks on its limits and its prospects,* Technical Report 95-95, Faculty of Technical Mathematics and Informatics, Delft Univeristy of Technology, 1995.

[Dett92] Dettmer, R., *Form and functions — Fractal based image compression,* IEE Review 38,9 (1992) 323–327.

[DoKu96] Domaszewicz, J., Kuklinski, S., *Fractal coding versus classified transform coding, Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[DoVa93] Domaszewicz, J., Vaishampayan, V. A., *Structural limitations of self-affine and partially self-affine fractal compression,* in: *Proceedings from SPIE Visual Communications and Image Processing,* Vol. 2094, pp. 1498–1504, 1993.

[DoVa94] Domaszewicz, J., Vaishampayan, V. A., *Iterative collage coding for fractal compression,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[DoVa95] Domaszewicz, J., Vaishampayan, V. A., *Graph-theoretical analysis of the fractal transform,* in: *Proceedings of ICASSP-1995 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 4, Detroit, 1995.

[Dudb92] Dudbridge, F., *Image approximation by self affine fractals,* PhD Thesis, University of London, 1992.

[Dudb94] Dudbridge, F., *Least-squares block coding by fractal functions,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[DuMo94] Dudbridge, F., Monro, D. M., *Fractal approximation functions for image and signal coding,* in: *Mathematics in Signal Processing III,* J. G. Mc Whirter (ed.), Clarendon Press, Oxford, 1994.

[DuSa96] Dugelay, J.-L., Sadoul, J.-M., *Moving picture fractal coding using a mixed approach I.F.S. and motion,* in: *Proceedings of the VIIIth European Signal Processing Conference EUSIPCO'96,* Trieste, Sept. 1996.

[DuSaBa95] Dugelay, J.-L., Sadoul, J.-M., Barakat, M., *Moving picture fractal compression using I.F.S.-A review,* Internal Report RR-95-018-bis, Multimedia and Communication Department, Eurecom, France, 1995.

[EwWo96] Ewing, G. J., Woodruff, C. J., *Comparison of JPEG and fractal-based image compression on target acquisition by human observers,* Optical Engineering 35 (1996) 284–288.

[FaGh94] de Faria, S., Ghanbari, M., *Variable block size fractal video coding with spatial transform motion compensation,* in: *Proceedings of the International Picture Coding Symposium PCS'94,* Sacramento, California, Sept. 1994.

[Fish92a] Fisher, Y., *Fractal image compression,* in: *Chaos and Fractals: New Frontiers of Science,* H.-O. Peitgen, H. Jürgens and D. Saupe, Springer-Verlag, New York, 1992.

[Fish92b] Fisher, Y., *Fractal image compression,* in: *Fractals — From Folk Art to Hyperreality,* P. Prusinkiewicz (ed.), ACM SIGGRAPH'92 Course Notes 12, 1992.

[Fish94a] Fisher, Y., *Fractal Image Compression — Theory and Application,* Springer-Verlag, New York, 1994.

[Fish94b] Fisher, Y., *Fractal Image Compression,* Fractals 2,3 (1994) 325–334.

[FiJaBo91] Fisher, Y., Jacobs, E. W., Boss, R. D., *Fractal image compression using iterated transforms,* Technical Report 1408, Naval Ocean Systems Center, San Diego, 1991.

[FiJaBo92] Fisher, Y., Jacobs, E. W., Boss, R. D., *Fractal image compression using iterated transforms,* in: *Image and Text Compression,* J. A. Storer (ed.), Kluwer Academic Publishers, Boston, 1992.

[FiLa90] Fisher, Y., Lawrence, A., *Fractal image coding: S.B.I.R Phase 1, Final Report,* Technical Report, Netrologic Inc., The University of California, San Diego and The Naval Ocean Systems Center, 1990.

[FiLa92] Fisher, Y., Lawrence, A., *Fractal image compression for mass storage applications,* in: *Proceedings from SPIE Visual Communications and Image Processing,* Vol. 1662, pp. 244–255, 1992.

[FiMe94] Fisher, Y., Menlove, S., *Fractal encoding with HV partitions,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[FiRoSh94] Fisher, Y., Rogovin, D., Shen, T.-P., *Fractal (self-VQ) encoding of video sequences,* in: *Proceedings from SPIE Visual Communications and Image Processing,* Chicago, Sept. 1994.

[FiShRo94] Fisher, Y., Shen, T.-P., Rogovin, D., *A comparison of fractal methods with dct (jpeg) and wavelets (epic),* in: *Proceedings from SPIE Neural and Stochastic Methods in Image and Signal Processing,* Vol. 2304-16, 1994.

[FoVr94a] ∗Forte, B., Vrscay, E. R., *Solving the inverse problem for function/image approximations using iterated function systems, I. Theoretical basis,* Fractals 2,3 (1994) 325–334.

[FoVr94b] ∗Forte, B., Vrscay, E. R., *Solving the inverse problem for function/image approximation using iterated function systems, II. Algorithm and computations,* Fractals 2,3 (1994) 335–346.

[FoVr95a] ∗Forte, B., Vrscay, E. R., *Solving the inverse problem for function and image approximation using iterated function systems,* to appear in Dynamics of Continuous, Discrete and Impulsive Systems 1,2 (1995).

[FoVr95b] ∗Forte, B., Vrscay, E. R., *Theory of generalized fractal transforms,* in Conf. Proc. NATO ASI *Fractal Image Encoding and Analysis,* Trondheim, July 1995, Y. Fisher (ed.), to appear in Springer-Verlag, New York, 1995.

[FrLaBi94a] Franich, R. E. H., Lagendijk, R. L., Biemond, J., *Fractal picture sequence coding: Looking for the effective search,* in: *Proceedings of the International Picture Coding Symposium PCS'94,* Sacramento, California, Sept. 1994.

[FrLaBi94b] Franich, R. E. H., Lagendijk, R. L., Biemond, J., *Fractal coding in an object-based system,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[Frig95] Frigaard, C., *Fast fractal 2D/3D image compression,* Manuscript, Institute for Electronic Systems, Aalborg University, 1995.

[FGHS94] ∗Frigaard, C., Gade, J., Hemmingsen, T., Sand, T., *Image compression based on fractal theory,* Manuscript, Institute for Electronic Systems, Aalborg University, 1994.

[FuAuLi95] Fu, K. W., Au, O. C., Liou, M. L., *Very low bit rate fractal video coding by genetic algorithm,* in: *Proc. of 2nd IASTED/ISMM Int. Conf. on Distributed Multimedia Systems and Applications,* Stanford, California, Aug. 1995.

[FYKS96] Fuchigami, T., Yano, S., Komatsu, T., Saito, T., *Fractal block coding with cost-based image partitioning,* in: *Proceedings of the International Picture Coding Symposium PCS'96,*, Melbourne, March 1996.

[GeLu96] Geronimo, J., Lu, N., *Lectures on fractal techniques in image compression,* in: *Image Tech Conference,* Atlanta, Georgia, March 1996.

[GhHu93a] Gharavi-Alkhansari, M., Huang, T., *A fractal-based image block-coding algorithm,* in: *Proceedings of ICASSP-1993 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 5, pp. 345–348, 1993.

[GhHu93b] Gharavi-Alkhansari, M., Huang, T., *A fractal-based image block-coding algorithm,* in: *Proceedings of the International Picture Coding Symposium PCS'93,* Lausanne, 1993.

[GhHu94a] Gharavi-Alkhansari, M., Huang, T. S., *Generalized image coding using fractal-based methods,* in: *Proceedings of the International Picture Coding Symposium PCS'94,* Sacramento, California, Sept. 1994.

[GhHu94b] Gharavi-Alkhansari, M., Huang, T., *Fractal based techniques for a generalized image coding method,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[GhHu96a] Gharavi-Alkhansari, M., Huang, T., *Fractal video coding by matching pursuit,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[GhHu96b] Gharavi-Alkhansari, M., Huang, T., *Fractal-based image and video Coding,* in: *Video Coding: The Second Generation Approach,* L. Torres and M. Kunt (eds.), Kluwer Academic Publishers, Boston, MA, 1996.

[GoDoAl94] González, F. J., Docampo, D., Alba, J. L., *Reducing the complexity in a fractal-based image coder,* in: *Proc. of the VIth IEEE 1994 Digital Signal Processing Workshop,* Yosemite, California, Oct. 1994.

[GoIbGr95] ∗Götting, D., Ibenthal, A., Grigat, R.-R., *Fractal image coding and magnification using invariant features,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[GrSt96] Grammalidis, N, Strintzis, M. G., *Hierarchical fractal image coding using a genetic algorithm,* in: *Proceedings of the International Picture Coding Symposium PCS'96,* Melbourne, March 1996.

[GuPoMe95] Guggisberg, M., Pontiggia, I., Meyer, U., *Parallel fractal image compression using iterated function systems,* Technical Report CSCS-TR-95-07, Swiss Scientific Computing Center, May 1995.

[Hafn95] Hafner, U., *Asymmetric coding in (m)-WFA image compression,* Technical Report 132, Dept. of Computer Science, University of Würzburg, 1995.

[Hafn96] Hafner, U., *Refining image compression with weighted finite automata,* in: *Proceedings DCC'96 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Comp. Soc. Press, March 1996.

[Hamz95] ∗Hamzaoui, R., *Codebook clustering by self-organizing maps for fractal image compression,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[Hamz96a] ∗Hamzaoui, R., *Decoding algorithm for fractal image compression,* Electronics Letters 32,14 (1996) 1273–1274.

[Hamz96b] Hamzaoui, R., *Fast decoding algorithms for fractal image compression,* Technical Report, Institut für Informatik, Universität Freiburg, to appear 1996.

[HaMuSa96a] ∗Hamzaoui, R., Müller, M., Saupe, D., *VQ-enhanced fractal image compression,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[HaMuSa96b] ∗Hamzaoui, R., Müller, M., Saupe, D., *Enhancing fractal image compression with vector quantization,* in: *Proc. IEEE Digital Signal Processing Workshop,* Loen, Sept. 1996.

[HaJa94] Hannah, S. J., Jackson D. J., *A hybrid fractal-LZW encoding technique for lossless image compression,* in: *Proceedings of the Thirty-Second Annual ACM Southeast Conference,* pp. 233-240, March 17-18, 1994.

[Hart94] Hart, J., *Fractal image compression and the inverse problem of recurrent iterated function systems,* in: *New Directions for Fractal Modeling in Computer Graphics,* ACM SIGGRAPH'94 Course Notes 13, J. Hart (ed.), 1994.

[Hart96] Hart, J., *Fractal image compression and the inverse problem of recurrent iterated function systems,* IEEE Computer Graphics and Applications 16,4 (1996) 25–32.

[HaViMa91] Hayes, M. H., Vines, G., Mazel, D. S., *Using fractals to model one-dimensional signals,* in: *Proceedings of the Thirteenth GRETSI Symposium,* pp. 197–200, 1991.

[Holl91] Hollatz, S. A., *Digital image compression with two-dimensional affine fractal interpolation functions,* Technical Report 91-2, Department of Mathematics and Statistics, University of Minnesota-Duluth, 1991.

[Huer93] ∗Hürtgen, B., *Contractivity of fractal transforms for image coding,* Electronics Letters 29 (1993) 1749–1750.

[Huer95a] ∗Hürtgen, B., *Performance bounds for fractal coding,* in: *Proceedings of ICASSP-1995 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 4, Detroit, 1995.

[Huer95b] ∗Hürtgen, B., *Statistical evaluation of fractal coding schemes,* in: *Proc. ICIP-95 IEEE International Conference on Image Processing,* Washington, D.C., 1995.

[HuBu93] ∗Hürtgen, B., Büttgen, P., *Fractal approach to low-rate video coding,* in: *Proceedings from SPIE Visual Communications and Image Processing,* Vol. 2094, pp. 120–131, 1993.

[HuHa94] ∗Hürtgen, B., Hain, T., *On the convergence of fractal transforms,* in: *Proceedings of ICASSP-1994 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 5, pp. 561–564, Adelaide, 1994.

[HuMoSi94] ∗Hürtgen, B., Mols, P., Simon, S. F., *Fractal transform coding of color images,* in: *Proceedings from SPIE Visual Communications and Image Processing,* pp. 1683–1691, 1994.

[HuMu94] ∗Hürtgen, B., Müller, F., *Modelling of fractal coding schemes,* in: *Proceedings of the VIIth European Signal Processing Conference EUSIPCO'94,* pp. 600–603, Edinburgh, 1994.

[HuMuSt93] ∗Hürtgen, B., Müller, F., Stiller, C., *Adaptive fractal coding of still pictures,* in: *Proceedings of the International Picture Coding Symposium PCS'93,* Lausanne, March 1993.

[HuSi94] ∗Hürtgen, B., Simon, S. F., *On the problem of convergence in fractal coding schemes,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Vol. 3, pp. 103–106, Austin, Texas, Nov. 1994.

[HuSt93] ∗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,* Berlin, 1993.

[Ida94] Ida, T., *Improvement of fractal image coding using new code-decision-criterion,* in: *Proceedings of the International Picture Coding Symposium PCS'94,* Sacramento, California, Sept. 1994.

[JaBl95] Jackson, D. J., Blom, T., *A parallel fractal image compression algorithm for hypercube multiprocessors,* in: *Proceedings of the Twenty-Seventh Southeastern Symposium on System Theory,* pp. 274-278, March 12-14, 1995.

[JaMa96] Jackson, D. J., Mahmoud, W., *Parallel pipelined fractal image compression using quadtree recomposition,* The Computer Journal 39,1 (1996) 1–13.

[JaTi95] Jackson, D. J., Tinney, G. S., *Performance analysis of distributed implementations of a fractal image compression algorithm,* to appear in Concurrency: Practice and Experience.

[JaBoFi90] Jacobs, E. W., Boss, R. D., Fisher, Y., *Fractal-based image compression II,* Technical Report 1362, Naval Ocean Systems Center, San Diego, June 1990.

[JaFiBo92] Jacobs, E. W., Fisher, Y., Boss, R. D., *Image compression: A study of the iterated transform method,* Signal Processing 29 (1992) 251–263.

[Jacq89a] Jacquin, A. E., *Image coding based on a fractal theory of iterated contractive Markov operators, Part I: Theoretical Foundation,* Report Math. 91389-016, Georgia Institute of Technology, 1989.

[Jacq89b] Jacquin, A. E., *Image coding based on a fractal theory of iterated contractive Markov operators, Part II: Construction of fractal codes for digital images,* Report Math. 91389-017, Georgia Institute of Technology, 1989.

[Jacq89c] Jacquin, A. E., *A Fractal Theory of Iterated Markov Operators with Applications to Digital Image Coding,* PhD Thesis, Georgia Institute of Technology, August 1989.

[Jacq90a] Jacquin, A. E., *Fractal image coding based on a theory of iterated contractive image transformations,* in: *Proceedings from SPIE Visual Communications and Image Processing,* Vol. 1360, pp. 227–239, 1990.

[Jacq90b] Jacquin, A. E., *A novel fractal block-coding technique for digital images,* in: *Proceedings of ICASSP-1990 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 4, pp. 2225–2228, 1990.

[Jacq92] Jacquin, A. E., *Image coding based on a fractal theory of iterated contractive image transformations,* IEEE Trans. Image Processing 1 (1992) 18–30.

[Jacq93] Jacquin, A. E., *Fractal image coding: A review,* Proceedings of the IEEE 81,10 (1993) 1451–1465.

[Kaou91] Kaouri, H., *Fractal coding of still images,* in: *IEE 6th International Conference on Digital Processing of Signals in Communications,* pp. 235–239, 1991.

[KaNaHi94] Kawamata, M., Nagahisa, M., Higuchi, T., *Multi-resolution tree search for iterated transformation theory-based coding,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[KiKiLe95] ∗Kim, C.-S., Kim, R.-C., Lee, S.-U, *Novel fractal image compression method with non-iterative decoder,* in: *Proc. ICIP-95 IEEE International Conference on Image Processing,* Washington, D.C., 1995.

[KiLe95] ∗Kim, C.-S., Lee, S.-U, *Fractal coding of video sequence by circular prediction mapping,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[KiPa94] Kim, I. K., Park, R.-H., *Image coding based on fractal approximation and vector quantization,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[Kocs89a] Kocsis, S., *Digital compression and iterated function systems,* in: *Proceedings from SPIE Application of Digital Image Processing,* Vol. 1153, pp. 19–27, 1989.

[Kocs89b] Kocsis, S., *Fractal-based image compression,* in: *Conference Record of the Twenty-Third Asilomar Conference on Signals, Systems and Computers,* pp. 177–181, 1989.

[Kohn96] Köhn, C., *Bildanalyse und Bilddatenkompression: fraktale Verfahren und klassische Methoden im praktischen Einsatz,* Hanser, München, Wien, 1996.

[Komi94a] Kominek, J., *Understanding fractal image compression,* Technical Report, Department of Computer Science, University of Waterloo, 1994.

[Komi94b] Kominek, J., *Image compression: An issue of quality,* Technical Report, Department of Computer Science, University of Waterloo, 1994.

[Komi95a] ∗Kominek, J., *Algorithm for fast fractal image compression,* in: *Proceedings from IS&T/SPIE 1995 Symposium on Electronic Imaging: Science & Technology,* Vol. 2419 Digital Video Compression: Algorithms and Technologies 1995.

[Komi95b] ∗Kominek, J., *Convergence of fractal encoded images,* in: *Proceedings DCC'95 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Comp. Soc. Press, March 1995.

[Komi96] Kominek, J., *Codebook reduction in fractal image compression,* in: *Proceedings from IS&T/SPIE 1996 Symposium on Electronic Imaging: Science & Technology – Still Image Compression II,* Vol. 2669, to appear Jan. 1996.

[KrMaKa95] *Krupnik, H., Malah, D., Karnin, E., *Fractal representation of images via the discrete wavelet transform,* in: *IEEE 18th Conf. of Electrical Engineering,* Tel-Aviv, March 1995.

[LaBr93] Lazar, M. S., Bruton, L. T., *Efficient fractal block coding using multiresolution decomposition based search constraints,* Technical Report, University of Calgary, Dept. of Electrical and Computer Engineering, Oct. 1993.

[LaBr94] Lazar, M. S., Bruton, L. T., *Fractal block coding of digital video,* IEEE Trans. on Circuits and Systems for Video Technology 4,3 (1994) 297–308.

[Leps93] Lepsøy, S., *Attractor Image Compression: Fast Algorithms and Comparisons to Related Techniques,* PhD Thesis, The Norwegian Institute of Technology, Trondheim, Norway, June 1993.

[LeOi92] Lepsøy, S., Øien, G. E., *Attractor image compression featuring direct attractor optimization and non-iterative decoding,* in: *INDUMAT-92, Nordic Conference on Industrial Mathematics,* 1992.

[LeOi94] Lepsøy, S., Øien, G. E., *Fast attractor image encoding by adaptive codebook clustering,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[LeOiRa93] Lepsøy, S., Øien, G. E., Ramstad, T., *Attractor image compression with a fast non-iterative decoding algorithm,* in: *Proceedings of ICASSP-1993 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 5, pp. 337–340, 1993.

[LeWi96] Levy, I., Wilson, R., *Predictive wavelet transform coding: unifying fractal and transform coding,* in: *Proceedings of the International Picture Coding Symposium PCS'96,* Melbourne, March 1996.

[LiNoFo93] Li, H., Novak, M., Forchheimer, R., *Fractal-based image sequence compression scheme,* Optical Engineering 32,7 (1993) 1588–1595.

[Lin94] Lin, D. W., *Fractal image coding as generalized predictive coding,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[Lin96] Lin, H., *Fast pyramid search for perceptually lossless fractal image compression,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[LiVe94a] Lin, H., Venetsanopoulos, A. N., *Incorporating nonlinear contractive functions into fractal coding,* in: *Proceedings of International Workshop on Intelligent Signal Processing and Communication Systems,* Seoul, Oct. 1994.

[LiVe94b] Lin, H., Venetsanopoulos, A. N., *Fractal-based image coding by nonlinear contractive functions,* in: *Proceedings of the 17th Biennial Symposium on Communications,* Kingston, Ontario, pp. 295–298, May-June 1994.

[LiVe95a] Lin, H., Venetsanopoulos, A. N., *Fast fractal image coding using pyramids,* in : *Proceedings of 8th International Conference on Image Analysis and Processing,* San Remo, Sept. 1995.

[LiVe95b] Lin, H., Venetsanopoulos, A. N., *Parallel implementation of fractal image compression,* in: *Proc. of Canadian Conference on Electrical and Computer Engineering,* Montreal, Sept. 1995.

[LiVe95c] Lin, H., Venetsanopoulos, A. N., *Removing blocking effects of fractal compressed images using multichannel filtering,* in: *Proceedings of 2nd International Workshop on Image and Signal Processing: Theory, Method, Systems and Applications,* Budapest, Nov. 1995.

[LiVe95d] Lin, H., Venetsanopoulos, A. N., *A pyramid algorithm for fast fractal image compression, Proc. ICIP-95 IEEE International Conference on Image Processing,* Washington, D.C., 1995.

[LiVe96] Lin, H., Venetsanopoulos, A. N., *Incorporating human visual system (HVS) models into the fractal image compression,* in: *Proceedings of ICASSP-1996 IEEE International Conference on Acoustics, Speech and Signal Processing,* 1996.

[LiMaMu93] Liu, J., Marlow, S., Murphy, N., *Multi-level fractal block coding in video compression,* in: *Proceedings of Amsterdam Conference "1993 DSP- The Enabling Technology for Communication",* 1993.

[Liu95] Liu, Y., *Solution of fractal equation,* in: *Proceedings from IS&T/SPIE 1995 Symposium on Electronic Imaging: Science & Technology,* Vol. 2418: Still-Image Compression 1995.

[Lu93] Lu, G., *Fractal image compression,* Signal Processing: Image Communication 5 (1993) 327–343.

[LuYe94a] Lu, G., Yew, T.-L., *Image compression using partitioned iterated function systems,* in: *Proceedings from SPIE Image and Video Compression,* Vol. 2186, 1994.

[LuYe94b] Lu, G., Yew, T.,-L., *Image compression using quadtree partitioned iterated function systems,* Electronics Letters 30,1 (1994) 23–24.

[Lund90] Lundheim, L. M., *Filters with fractal root signals suitable for signal modelling and coding,* in: *Proc. of the IEEE 1990 Digital Signal Processing Workshop,* pp. 1.3.1-1.3.2, New Palz, 1990.

[Lund92] Lundheim, L. M., *Fractal Signal Modellings for Source Coding*, PhD Thesis, The Norwegian Institute of Technology, Trondheim, Sept. 1992.

[Lund94] Lundheim, L. M., *A discrete framework for fractal signal modelling,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[MaAuLi96] Ma, L. K., Au, O. C., Liou, M. L., *A novel method in reducing the complexity of fractal encoding,* in: *Proceedings of the VIIIth European Signal Processing Conference EUSIPCO'96,* Trieste, Sept. 1996.

[MaSl89] Mantica, G., Sloan, A., *Chaotic optimization and the construction of fractals: Solution of an inverse problem,* Compl. Syst. 3 (1989) 37–62.

[Maze91] Mazel, D. S., *Fractal Modelling of Time-Series Data,* PhD Thesis, Georgia Institute of Technology, 1991.

[MaHa89] Mazel, D. S., Hayes, M. H., *Fractal modeling of time-series data,* in: *Conference Record of the Twenty-Third Asilomar Conference of Signals, Systems and Computers*, pp. 182–186, 1989.

[MaHa90] Mazel, D. S., Hayes, M. H., *A piece-wise self-affine fractal model for discrete sequences,* in: *Proc. of the IEEE 1990 Digital Signal Processing Workshop,* New Palz, 1990.

[MaHa91] Mazel, D. S., Hayes, M. H., *Hidden-variable fractal interpolation of discrete sequences, Proceedings of ICASSP-1991 IEEE International Conference on Acoustics, Speech and Signal Processing,* pp. 3393–3396, 1991.

[MaHa92] Mazel, D. S., Hayes, M. H., *Using iterated function systems to model discrete sequences,* IEEE Transactions on Signal Processing 40,7 (1992) 1724–1734.

[Monr93a] Monro, D. M., *A hybrid fractal transform,* in: *Proceedings of ICASSP-1993 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 5, pp. 169–172, 1993.

[Monr93b] Monro, D. M., *Generalized fractal transforms: Complexity issues,* in: *Proceedings DCC'93 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Comp. Soc. Press, 1993.

[Monr93c] Monro, D. M., *Class of fractal transforms,* Electronics Letters 29,4 (1993) 362–363.

[Monr93d] Monro, D. M., *Fractal transforms: Complexity versus fidelity,* in: *Image Processing: Theory and Applications,* G. Vernazza, A. N. Venetsanopoulos and C. Braccini (eds.), Elsevier Science Publishers, 1993.

[Monr93e] Monro, D. M., *An evaluation of fractal transforms for image approximations,* in: *Proc. Technical Association of the Graphic Arts (TAGA) 45th Annual Technical Conference,* Minneapolis, April 1993.

[Monro95] Monro, D. M., *A hybrid fractal transform,* in: *Signal Processing Technology and Applications,* J. G. Ackenhusen (ed.), IEEE technology update series, 1995.

[MoDu92a] Monro, D. M., Dudbridge, F., *Fractal approximation of image blocks,* in: *Proceedings of ICASSP-1992 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 3, pp. 485–488, 1992.

[MoDu92b] Monro, D. M., Dudbridge, F., *Fractal block coding of images,* Electronics Letters 28,11 (1992) 1053–1055.

[MoDu95] Monro, D. M., Dudbridge, F., *Rendering algorithms for deterministic fractals,* IEEE Computer Graphics and Applications 15,1 (1995) 32–41.

[MoNi94] Monro, D. M., Nicholls, J. A., *Real time fractal video for personal communications,* Fractals 2,3 (1994) 391–394.

[MoNi95] Monro, D. M., Nicholls, J. A., *Low bit rate colour fractal video, Proc. ICIP-95 IEEE International Conference on Image Processing,* Washington, D.C., 1995.

[MoWiNi93] Monro, D. M., Wilson, D. L., Nicholls, J. A., *High speed image coding with the Bath fractal transform,* in: *IEEE International Symposium on Multimedia Technologies and Future Applications,* Southampton, April 1993.

[MoWo94a] Monro, D. M., Woolley, S. J., *Fractal image compression without searching,* in: *Proceedings of ICASSP-1994 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 5, Adelaide, 1994.

[MoWo94b] Monro, D. M., Woolley, S. J., *Rate/distortion in fractal compression: order of transform and block symmetries,*in: *Proc. ISSIPNN'94 Intern. Symp. on Speech, Image Processing and Neural Networks,* pp. 168–171, Hong Kong, 1994.

[MSKKK95] Moon, Y. H., Son, K. S., Kim, H. S., Kim, Y. S., Kim, J. H., *Fractal image compression using human visual system,* in: *Proceedings from IS&T/SPIE 1995 Symposium on Electronic Imaging: Science & Technology,* Vol. 2418: Still-Image Compression 1995.

[NaHa94a] Naemura, T., Harashima, H., *Data compression & interpolation of a multi-viewpoint image set using iterated contractive image transformation,* in: *Proceedings of the International Picture Coding Symposium PCS'94,* Sacramento, California, Sept. 1994.

[NaHa94b] Naemura, T., Harashima, H., *Fractal coding of a multi-view 3-d image,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[NeGa94] Nettleton, D. J., Garigliano, R., *Evolutionary algorithms and a fractal inverse problem,* Biosystems 33,3 (1994) 221–231.

[NiMo95] Nicholls, J. A., Monro, D. M., *Adaptive fractal video,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[Nova93a] ∗Novak, M., *Attractor coding of images,* Licentiate Dissertation, Dept. of Electrical Engineering, Linköping University, May 1993.

[Nova93b] ∗Novak, M., *Attractor coding of images,* in: *Proceedings of the International Picture Coding Symposium PCS'93,* Lausanne, March 1993.

[NoNaWa92] Novak, M., Nautsch, H., Wadstrømer, N., *Fractal coding of images,* in: *Symposium on Image Analysis,* Uppsala, pp. 101–104, March 1992.

[ObCaKu95] Obrador, P., Caso, G., Kuo, C-C. J., *A fractal-based method for textured image compression,* in: *Proceedings from IS&T/SPIE 1995 Symposium on Electronic Imaging: Science & Technology,* Vol. 2418: Still-Image Compression 1995.

[Oien93] Øien, G. E., *L_2 Optimal Attractor Image Coding with Fast Decoder Convergence*, PhD Thesis, The Norwegian Institute of Technology, Trondheim, Norway, April 1993.

[Oien94] Øien, G. E., *Parameter quantization in fractal image coding,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[OBLMK94] ∗Øien, G. E., Baharav, Z., Lepsøy, S., Malah, D., Karnin, E., *A new improved collage theorem with applications to multiresolution fractal image coding,* in: *Proceedings of ICASSP-1994 IEEE International Conference on Acoustics, Speech and Signal Processing,* Adelaide, 1994.

[OiHaSa96] Øien, G. E., Hamzaoui, R., Saupe, D., *On the limitations of fractal image texture coding,* in: *Proceedings of NORSIG'96 1996 IEEE Nordic Signal Processing Symposium,* Espoo, Sept. 1996.

[OiLe94a] 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.), Springer-Verlag, New York, 1994.

[OiLe94b] Øien, G. E., Lepsøy, S., *Fractal-based image coding with fast decoder convergence,* Signal Processing 40 (1994) 105–117.

[OiLeRa91] Øien, G. E., Lepsøy, S., Ramstad, T., *An inner product space approach to image coding by contractive transformations,* in: *Proceedings of ICASSP-1991 IEEE International Conference on Acoustics, Speech and Signal Processing,* pp. 2773–2776, 1991.

[OiLeRa92] Øien, G. E., Lepsøy, S., Ramstad, T., *Reducing the complexity of a fractal-based image coder,* in: *Proceedings of the Vth European Signal Processing Conference EUSIPCO'92,* pp. 1353–1356, 1992.

[PaHa94] Paul, B.-B., Hayes, M. H., *Fractal-based compression of motion video sequences,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* pp. 755–759, Austin, Texas, Nov. 1994.

[PaHa95] Paul, B.-B., Hayes, M. H., *Video Coding based on iterated function systems,* in: *Proceedings of ICASSP-1995 IEEE International Conference on Acoustics, Speech and Signal Processing,* Detroit, 1995.

[PeHo91] Pentland, A., Horowitz, B., *A practical approach to fractal-based image compression,* in: *Proceedings from SPIE Visual Communications and Image Processing,* Vol. 1605, pp. 467–474, 1991.

[PoDu95] Polidori, E., Dugelay, J.-L., *Zooming using IFS,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[PoDiYa95] Popescu, D., Dimca, A., Yan, H., *Generalized square isometries - An Improvement for fractal image compression,* in : *Proceedings of 8th International Conference on Image Analysis and Processing,* San Remo, Sept. 1995.

[PoYa93] Popescu, D. C., Yan, H., *MR image compression using iterated function systems,* Magnetic Resonance Imaging 11 (1993) 727–732.

[PoYa95] Popescu, D., Yan, H., *Fractal-based method for color image compression,* Journal of Electronic Imaging 4,1 (1995) 23–30.

[RaKa93a] Raittinen, H., Kaski, K., *Image compression with affine transformations,* in: *IEEE Winter Workshop on Nonlinear Digital Signal Processing,* Tampere, Finland, Vol. 2, pp. 2.1–2.6, Jan. 1993.

[RaKa93b] Raittinen, H., Kaski, K., *Fractal based image compression with affine transformations,* in: *Proceedings DCC'93 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Comp. Soc. Press, 1993.

[RaLe93] Ramstad, T. A., Lepsøy, S., *Block-based attractor coding: Potential and comparison to vector quantization,* in: *Conference Record of the Twenty-Seventh Asilomar Conference on Signals, Systems and Computers,* pp. 1504–1508, 1993.

[Reus93] ∗Reusens, E., *Sequence coding based on the fractal theory of iterated transformations systems,* in: *Proc. of Visual Communications and Image Processing VCIP,* Boston, 1993.

[Reus94a] ∗Reusens, E., *Overlapped adaptive partitioning for image coding based on the theory of iterated function systems,* in: *Proceedings of ICASSP-1994 IEEE International Conference on Acoustics, Speech and Signal Processing,* Adelaide, 1994.

[Reus94b] ∗Reusens, E., *Partitioning complexity issue for iterated function systems based image coding,* in: *Proceedings of the VIIth European Signal Processing Conference EUSIPCO'94,* Edinburgh, Sept. 1994.

[RiCa94] ∗Rinaldo, R., Calvagno, G., *An image coding scheme using block prediction of the pyramid subband decomposition,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[RiCa95] ∗Rinaldo, R., Calvagno, G., *Image coding by block prediction of multiresolution subimages,* IEEE Trans. on Image Proc. 4,7 (1995) 909–920.

[RiZa92] Rinaldo, R., Zakhor, A., *Inverse problem for two-dimensional fractal sets using the wavelet transform and the moment method,* in: *Proceedings of ICASSP-1992 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 4, San Francisco, California, 1992.

[RiZa93] Rinaldo, R., Zakhor, A., *Fractal approximations of images,* in: *Proceedings DCC'93 Data Compression Conference,* J. A. Storer and M. Cohn (ed.), IEEE Comp. Soc. Press, 1993.

[RiZa94] Rinaldo, R., Zakhor, A., *Inverse and approximation problem for two dimensional fractal sets,* IEEE Trans. Image Processing 3,6 (1994) 802–820.

[RoDu95] Roche, S., Dugelay, J.-L., *Improvement in i.f.s. formulation for its use in still image coding,* in: *Proceedings of 1995 IEEE Workshop on NSIP,* Halkidiki, June 1995.

[RoMoDu96] Roche, S., Molva, R., Dugelay, J. L., *Multi resolution access control algorithm based on fractal coding,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[Saup94a] ∗Saupe, D., *Breaking the time complexity of fractal image compression,* Technical Report 53, Institut für Informatik, Universität Freiburg, 1994.

[Saup94b] Saupe, D., *From classification to multi-dimensional keys,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[Saup95a] ∗Saupe, D., *Accelerating fractal image compression by multi-dimensional nearest neighbor search,* in: *Proceedings DCC'95 Data Compression Conference,* J. A. Storer and M. Cohn (eds.), IEEE Comp. Soc. Press, March 1995.

[Saup95b] ∗Saupe, D., *Fractal image compression via nearest neighbor search,* in: Conf. Proc. NATO ASI *Fractal Image Encoding and Analysis,* Trondheim, July 1995, Y. Fisher (ed.), to appear in Springer-Verlag, New York, 1995.

[Saup96a] ∗Saupe, D., *Lean domain pools for fractal image compression,* in: *Proceedings from IS&T/SPIE 1996 Symposium on Electronic Imaging: Science & Technology – Still Image Compression II,* Vol. 2669, to appear Jan. 1996.

[Saup96b] Saupe, D., *A new view of fractal image compression as convolution transform coding,* IEEE Signal Processing Letters 3,7 (1996).

[Saup96c] ∗Saupe, D., *The futility of square isometries in fractal image compression,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[SaBa94] Saupe, D., Bayer, K., *Visualizing fractal image compression,* in: *Visualization in Scientific Computing,* M. Göbel, H. Müller and B.Urban (eds.), Springer-Verlag, Wien, 1995.

[SaHa94a] ∗Saupe, D., Hamzaoui, R., *A guided tour of the fractal image compression literature,* in: *New Directions for Fractal Modeling in Computer Graphics,* J. Hart (ed.), ACM SIGGRAPH'94 Course Notes 13, 1994.

[SaHa94b] Saupe, D., Hamzaoui, R., *A review of the fractal image compression literature,* Computer Graphics 28,4 (1994) 268–279.

[SaHa94c] ∗Saupe, D., Hamzaoui, R., *Complexity reduction methods for fractal image compression,* in: *I.M.A. Conf. Proc. on Image Processing; Mathematical Methods and Applications,* Sept. 1994, J.M. Blackledge (ed.), to appear with Oxford University Press, 1996.

[SaHaHar96] ∗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.

[SaHar96a] ∗Saupe, D., Hartenstein, H., *Lossless acceleration of fractal image compression by fast convolution,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[SaRoFi95] Saupe, D., Rombach, M., Fischer, H., *Fuzzy clustering for fractal image compression with applications to digital angiography,* in: *Proc. The Third European Congress on Intelligent Techniques and Soft Computing EUFIT'95,* Aachen, Aug. 1995.

[SaRu96a] ∗Saupe, D., Ruhl, M., *Evolutionary fractal image compression,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[Sign95] ∗Signes, J., *Geometrical interpretation of IFS based image coding,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[Simo95a] Simon, B., *Fractal transformations and the wavelet multiresolution representation,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[Simo95b] ∗Simon, B., *Explicit link between local fractal transform and multiresolution transform,* in: *Proc. ICIP-95 IEEE International Conference on Image Processing,* Washington, D.C., 1995.

[Simo96] Simon, B., *Image coding using overlapping fractal transform in the wavelet domain,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[Scho91] van Schooneveld, D., *Fractal coding of monochrome images,* Report WBB-M, Dept. of Mathematics, Delft University of Technology, 1991.

[SciAm88] *Fractal shorthand,* Scientific American 258,2 (1988) 28.

[SiMa93] Sirgany, W. N., Mazel, D. S., *Correlation effects of fractal compression,* in: *Conference Record of the Twenty-Seventh Asilomar Conference on Signals, Systems and Computers,* pp. 1524–1528, 1993.

[Skar93] Skarbek, W., *Metody reprezentacji obrazow cyfrowych (Digital Image Representation Methods),* Akademicka Oficyna Wydawnicza PLJ, Warsaw, 1993.

[Skar94a] Skarbek, W., *Banach constructor in fractal compression,* Machine Graphics & Vision 3,1-2 (1994) 431–441.

[Skar94b] Skarbek, W., *Choice of Banach space for fractal compression of digital images,* in: *Proceedings of the 5th Microcomputer School Computer Vision and Graphics,* Zakopane, 1994.

[Skar96] Skarbek, W., *On convergence of affine fractal operators,* Image Processing and Communications 1,1 (1996) 33–41.

[SkIg96] Skarbek, W., Ignasiak, K., *Asynchronous nonlinear fractal operators and their applications,* to appear in Image Processing and Communications, 1996.

[StMaJa96] Stapleton, W. A., Mahmoud, W., Jackson, D. J., *A parallel implementation of a fractal image compression algorithm,* in: *Proc. of the Twenty-Eigth Southeastern Symposium on System Theory,* March-April, 1996.

[TKOFK96] Tanimoto, M., Katsuyama, S., Ohyama, H., Fujii, T., Kimoto, T., *A new fractal image coding employing blocks of variable shapes,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[Thao96] Thao, N. T., *A hybrid fractal-DCT coding scheme for image compression,* in: *Proc. ICIP-96 IEEE International Conference on Image Processing,* Lausanne, Sept. 1996.

[ThDe93] Thomas, L., Deravi, F., *Pruning of the transform space in block-based fractal image compression,* in: *Proceedings of ICASSP-1993 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 5, pp. 341–344, 1993.

[ThDe95] Thomas, L., Deravi, F., *Region-based fractal image Compression using heuristic search,* IEEE Transactions on Image Processing (1995) 4,6 832–838.

[VeRu94] ∗Vences, L., Rudomin, I., *Fractal compression of single images and image sequences using genetic algorithms,* Manuscript, Institute of Technology, University of Monterrey, 1994.

[Vine93] ∗Vines, G., *Signal Modeling with Iterated Function Systems,* PhD Thesis, Georgia Institute of Technology, Atlanta, 1993.

[Vine94] Vines, G., *Orthogonal basis IFS,* in: *Fractal Image Compression — Theory and Application,* Y. Fisher (ed.), Springer-Verlag, New York, 1994.

[ViHa92a] Vines, G., Hayes, M. H., *Using hidden variable fractal interpolation to model one-dimensional signals,* in: *Proceedings of the Vth European Signal Processing Conference EUSIPCO'92,* 1992.

[ViHa92b] Vines, G., Hayes, M. H., *Fast algorithm to select maps in an iterated function system fractal model,* in: *Proceedings from SPIE Visual Communications and Image Processing,* Vol. 1818, pp. 944–949, 1992.

[ViHa93a] Vines, G., Hayes, M. H., *Adaptive IFS image coding with proximity maps,* in: *Proceedings of ICASSP-1993 IEEE International Conference on Acoustics, Speech and Signal Processing,* Vol. 5, pp. 349–352, 1993.

[ViHa93b] Vines, G., Hayes, M. H., *Nonlinear address maps in a one-dimensional fractal model,* IEEE Trans. on Signal Processing 41,4 (1993) 1721–1724.

[Vrsc91a] Vrscay, E. R., *Iterated function systems: Theory, applications, and the inverse problem,* in: *Fractal Geometry and Analysis,* J. Belair and S. Dubuc (eds.), Kluwer Academic, 1991.

[Vrsc91b] Vrscay, E. R., *Moment and collage methods for the inverse problem of fractal construction with iterated function systems,* in: *Fractals in the Fundamental and Applied Sciences,* H.-O. Peitgen, J. M. Henriques and L. Peneda (eds), North-Holland, Amsterdam, 1991.

[Wait90] Waite, J., *A review of iterated function system theory for image compression,* in: *Proceedings IEE Colloquium: The application of fractal techniques in image processing,* pp. 1/1- 1/9, 1990.

[WaKa86] Walach, E., Karnin, E., *A fractal based approach to image compression,* in: *Proceedings of ICASSP-1986 IEEE International Conference on Acoustics, Speech and Signal Processing,* pp. 529–532, 1986.

[WaKi93] Wall, L., Kinsner, W., *A fractal block coding technique employing frequency sensitive competitive learning,* in: *Proc. of IEEE Communications, Computers and Power,* pp. 320–329, 1993.

[Wall95] ∗van der Walle, A., *Merging fractal image compression and wavelet transform methods,* in: *NATO ASI Conf. Fractal Image Encoding and Analysis,* Trondheim, July 1995, to appear in a special issue of Fractals.

[WeBl96] Wein, C. J., Blake, I. F., *On the performance of fractal compression with clustering,* IEEE Transactions on Image Processing 5,3 (1996) 522–526.

[Wils88] Wilson, D., *Fractal image compression,* Technical Report, Imperial College of Science, Technology & Medicine, University of London, 1988.

[WiNiMo94] Wilson, D. L., Nicholls, J. A., Monro, D. M., *Rate buffered fractal video,* in: *Proceedings of ICASSP-1994 IEEE International Conference on Acoustics, Speech and Signal Processing,* Adelaide, 1994.

[With89] Withers, W. D., *Newton's method for fractal approximation,* Constructive Approximation 5 (1989) 151–179.

[WoJa95] ∗Wohlberg, B. E., de Jager, G., *Fast image domain fractal compression by DCT domain block matching,* Electronics Letters 31 (1995) 869–870.

[WoJa96] Wohlberg, B. E., de Jager, G., *Fractal coding performance for first-order Gauss-Markov models,* Electronics Letters 32,5 (1996) 441–442.

[WoHsKu93] Wong, K.-J., Hsu, C.-H., Kuo, C.-C. J., *Fractal-based image coding with polyphase decomposition,* in: *Proceedings from SPIE Visual Communications and Image Processing,* Vol. 2094, pp. 1207–1218, 1993.

[WoMo94] Woolley, S., Monro, D. M., *Rate/distortion performance of fractal transforms for image compression,* Fractals 2,3 (1994) 395–398.

[WoMo95] Woolley, S. J., Monro, D. M., *Optimum parameters for hybrid fractal image coding,* in: *Proceedings of ICASSP-1995 IEEE International Conference on Acoustics, Speech and Signal Processing,* Detroit, 1995.

[XuHaMe94] Xue, M., Hanson, T., Merigot, A., *A massively parallel implementation of fractal image compression,* in: *Proc. ICIP-94 IEEE International Conference on Image Processing,* Austin, Texas, Nov. 1994.

[ZhgaHe95] Zhang, H., Gao, X., He, Zhenya, *A modified fractal transform,* in: *Proceedings of ICASSP-1995 IEEE International Conference on Acoustics, Speech and Signal Processing,* Detroit, 1995.

[ZhPo95] ∗Zhang, Y., Po, L. M., *Fractal color image compression using vector distortion measure,* in: *Proc. ICIP-95 IEEE International Conference on Image Processing,* Washington, D.C., 1995.

[ZhYa91] Zhang, N., Yan, H., *Hybrid image compression method based on fractal geometry,* Electronics Letters 27,5 (1991) 406–408.

[ZhYu94] Zhao, Y., Yuan, B., *Image compression using fractals and discrete cosine transform,* Electronics Letters 30 (1994) 474–475.

[ZhYu95] Zhao, Y., Yuan, B., *A fractal hybrid image compression based on DCT,* to appear in: Signal Processing: Image Communication.

[ZhChTi94] Zhu, X., Cheng, B., Titterington, D. M., *Fractal model of a one-dimensional discrete signal and its implementation,* IEE Proc. Vision Image and Signal Process. 141,5 (1994) 318–324.