Fractals, Vol. 14, No. 4 (2006) 1–11

World Scientific
www.worldscientific.com

IMAGE COMPRESSION USING AFFINE FRACTAL INTERPOLATION ON RECTANGULAR LATTICES

V. DRAKOPOULOS,* P. BOUBOULIS†,‡ and S. THEODORIDIS†,§
Department of Informatics and Telecommunications
* *Theoretical Informatics*
† *Telecommunications and Signal Processing*
University of Athens, Panepistimioupolis 157 84, Athens, Hellas, Greece
**vasilios@di.uoa.gr*
‡*macaddic@otenet.gr*
§*stheodor@di.uoa.gr*

Received December 30, 2005
Accepted May 30, 2006

Abstract

Two methods for representing discrete image data on rectangular lattices using fractal surfaces are proposed. They offer the advantage of a more general fractal modeling compared to previous one-dimensional fractal interpolation techniques resulting in higher compression ratios. Theory, implementation and analytical study of the proposed methods are also presented.

Keywords: Fractal Interpolation; Reconstruction; IFS; Fractal Image Coding; Fractal Surface.

1. INTRODUCTION

The theory of image coding using an *iterated function system*, or *IFS* for short, was first proposed by Barnsley (see, for example, Ref. 1). With the help of IFS's along with a collage theorem, he laid the foundation of the fractal-based image compression. A set of contractive affine transformations can approximate a real image and so, instead of storing the whole image, it is enough to store the relevant parameters of the transformations reducing memory requirements and achieving high compression ratios.

The effectiveness of *fractal image compression*, or *FIC* for short, has been demonstrated by Jacquin,[2] Barnsley and Hurd[3] and Fisher[4] by showing that a well-designed fractal compressor yields comparable

compression ratios and image quality to the JPEG algorithm. Moreover, FIC has the unique property of *resolution-independence*, that is, the same fractal representation can be decoded to various output devices in the best resolution for each of them. However, the computational requirements of the compression algorithm are orders of magnitude greater than those of the decompressor. An overview of the variety of schemes that have been investigated can be found in Wohlberg and de Jager.[5] The books by Lu[6] and Fisher[4] combine introductory material with an in-depth discussion of many aspects of fractal coding.

Barnsley introduced a class of functions (see, for example, Ref. 1) which he called *fractal interpolation functions*, or *FIF's* for short. He worked basically with *affine FIF's*, in the sense that they are obtained using affine transformations. After that, FIF's can be used as a variation of fractal image compression methods. Mazel and Hayes[7] first introduced two methods based on FIF's for modeling single-valued discrete-time sequences: the *self-affine* and the *piecewise self-affine fractal model*. A few years later, in 1994, Ali and Clarkson[8] applied the self-affine model in order to compress static image frames. Our intention is to further extend the aforementioned models so that the resulting attractor lies entirely in $\mathbb{R}^3$ and not to be the projection on $\mathbb{R}^2$, like in the *piecewise hidden-variable fractal model*, which was an extension of the piecewise self-affine IFS model from $\mathbb{R}^2$ to $\mathbb{R}^3$ by Mazel[9]. Furthermore, the model that we propose is also piecewise self-affine. This extension permits the representation of static images with a fractal model that is more general, flexible and computationally efficient compared with the hidden-variable model.

2. THE 1D FRACTAL INTERPOLATION MODEL

2.1. Basic Concepts of FIC Using IFS's

An IFS is defined as a pair consisting of a complete metric space (X, ρ), e.g. $(\mathbb{R}^n, \|\cdot\|)$ or a subset, and a finite set of continuous mappings $w_i\colon X \to X$, $i = 1, 2, \ldots, M$. It is often convenient to write an IFS formally as $\{X; w_1, w_2, \ldots, w_M\}$ or, somewhat more briefly, as $\{X; w_{1-M}\}$. If w_i are contractions with respective *contractivity factors* $s_i, i = 1, 2, \ldots, M$, the IFS is termed *hyperbolic*. The *attractor* of a hyperbolic IFS is the unique set $A_\infty = \lim_{k\to\infty} W^k(A_0)$ for every starting set A_0, where

$$W(A) = \bigcup_{i=1}^{M} w_i(A) \quad \text{for all } A \in \mathcal{H}(X),$$

and $\mathcal{H}(X)$ is the metric space of all non-empty compact subsets of X with respect to some metric, e.g. the Hausdorff metric.

The fundamental principle of FIC consists of the representation of an image by an IFS of which the *fixed point* is close to that image. Therefore, the encoding process is first to find an IFS and then a suitable transformation W whose fixed point is close to the given image. The usual approach is based on the next theorem.

Theorem 1 (The Collage Theorem). *If* $B \in (\mathcal{H}(X), h)$, *where* h *is a metric, obeys*

$$h\left(B, W(B)\right) \le \varepsilon,$$

then

$$h(B, A_\infty) \le \frac{\varepsilon}{1-s},$$

where $s = \max\{s_i : i = 1, 2, \ldots, M\}$.

The Collage Theorem provides a measure of the goodness of fit of an attractor associated with an IFS and a given non-empty compact set. An attractor that is close to a given set is one with an associated IFS such that the union of all the maps applied to the given set is close to the given set. The closer the union is to the given set, the closer the attractor of the IFS will be to the given set. Therefore, in order to test the closeness of an attractor to a given set, one need not compute the attractor itself. The collage theorem, however, is not constructive, it does not indicate how to find a set of proper maps, but rather, it provides a way to test an IFS without need for computation of the attractor.

While a few impressive examples of image modeling were generated by the original approach devised by Barnsley, no automated encoding algorithm was found. FIC became a practical reality with the introduction by Jacquin of the Partitioned (or Local) IFS (PIFS) which differs from an IFS in that each of the individual transformation operates on a subset of the image, rather than the entire image. Since the *image support* is tiled by "range blocks" each of which is mapped from one of the "domain blocks" as depicted in Fig. 1, the combined

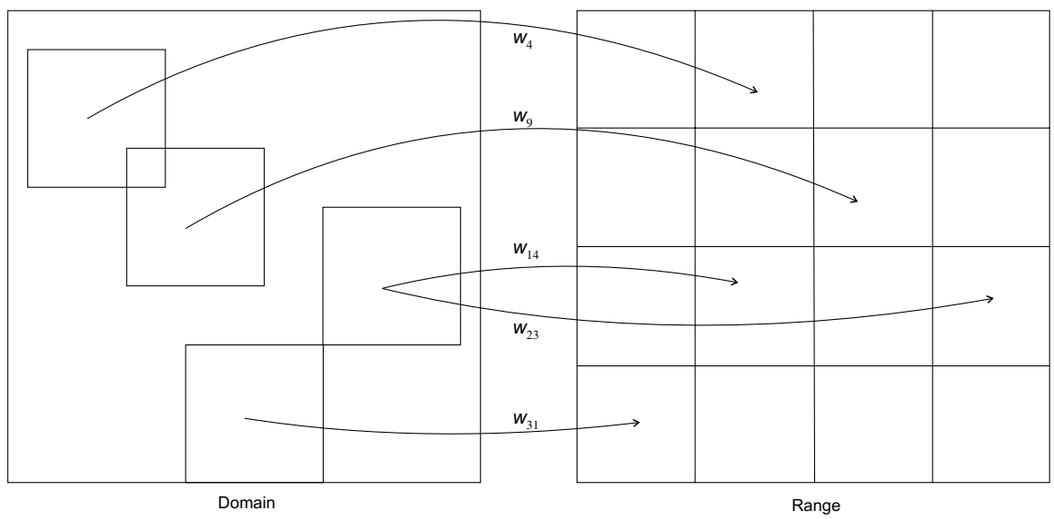


Fig. 1 Each range block is constructed by a transformed domain block.

mappings constitute a transformation on the image as a whole. The transformation minimizing the collage error within this framework is constructed by individually minimizing the collage error for each range block, which requires locating the domain block which may be made closest to it under an admissible block mapping.

2.2. Image Compression Using Affine Fractal Interpolation Functions

It is well known (see, for example, Ref. 1) that the attractor of an appropriately chosen IFS is the graph of some continuous (usually fractal) function which interpolates the *data points*. By choosing the *interpolation points*, say $\{(x_m, y_m) \in I \times \mathbb{R} : m = 0, 1, \ldots, M\}$ where x_m is strictly increasing and $I = [x_0, x_M] \subset \mathbb{R}$, as a subset of the data points we seek a continuous function $f: I \to \mathbb{R}$ (with fractal characteristics) that interpolates the above-mentioned points according to $f(x_m) = y_m$ for every $m = 0, 1, \ldots, M$. A *section* is defined as the function values *between* interpolation points. Affine fractal interpolation produces a self-affine (interpolation) function by mapping the entire (graph of the) function to each section of the function. The interpolation function is constructed with M affine mappings of the form

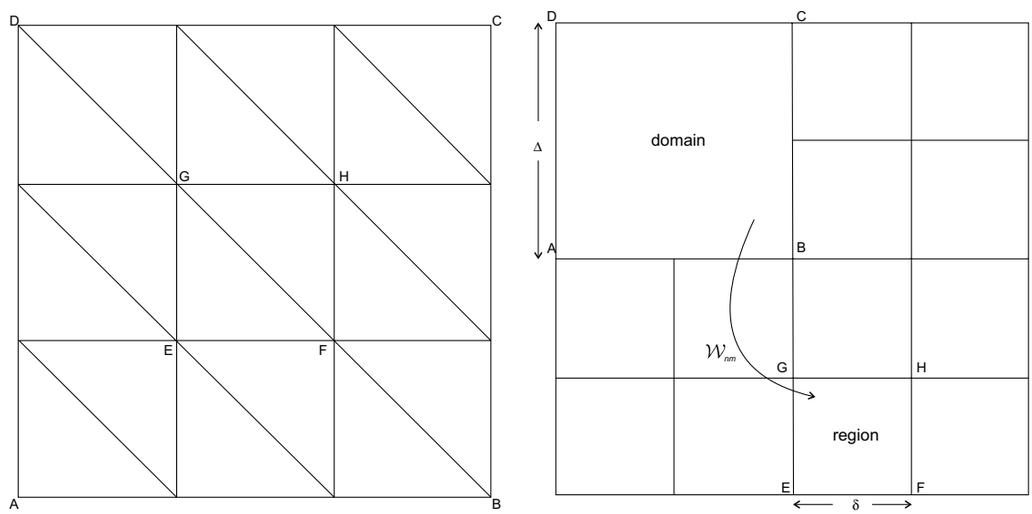
$$w_i \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_i & 0 \\ c_i & s_i \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} d_i \\ e_i \end{pmatrix} \tag{1}$$

with section endpoint constraints

$$w_i \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} x_{i-1} \\ y_{i-1} \end{pmatrix} \quad \text{and}$$
$$w_i \begin{pmatrix} x_M \\ y_M \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix}, \tag{2}$$

for all $i = 1, 2, \ldots, M$. We refer to s_i as the *contractivity factors*. With each s_i constrained to lie in the interval $(-1, 1)$, the collection of affine mappings defined by (1)–(2) form a hyperbolic IFS. The Collage Theorem assures the goodness of fit of such a function.

The affine fractal interpolation was expanded in Mazel and Hayes[7] so that a pair of data points, which are called *addresses*, is now associated with each affine mapping. The pair of addresses associated with the ith map is called *domain* and denote the interval of the function that is mapped between the interpolation points. The piecewise self-affine fractal model produces a function by mapping domains of the function to sections of the function. In addition, the width of each interval associated with a map is constrained to be strictly greater than the width of the section associated with that mapping. Note that points within a given section are not necessarilly contained within any domain. The piecewise self-affine fractal model is a generalization of the affine fractal interpolation model and has its mathematical roots embedded in *recurrent IFS*, or *RIFS* for short, theory.[10]



(a) (b)

Fig. 2 Domains for fractal interpolating surfaces over rectangular lattices using RIFS on **(a)** triangular tiling and **(b)** rectangular tiling.

3. IMAGE COMPRESSION USING 2D PIECEWISE SELF-AFFINE FRACTAL MODELS

In this section we introduce two piecewise self-affine fractal models suitable for 2D signals that can be viewed as a generalization of the one mentioned in Mazel.[9] This means that sections and domains do not lie on $\mathbb{R}$, but on $\mathbb{R}^2$ instead, yielding squares of width δ and Δ, respectively. A 2D section is called *range*. When we wish to refer to an image, we refer to the function $z = f(x, y)$ that gives the grey level at each point (x, y).

Let the discrete data $\{(x_i, y_j, z_{ij} = z(x_i, y_j)) \in \mathbb{R}^3 : i = 0, 1, \ldots N; j = 0, 1, \ldots, M\}$ be known. Each affine mapping that comprises the hyperbolic IFS $\{\mathbb{R}^3; w_{1-N,1-M}\}$ is given by

$$w_{nm}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a_{nm} & b_{nm} & 0 \\ c_{nm} & d_{nm} & 0 \\ e_{nm} & g_{nm} & s_{nm} \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} h_{nm} \\ k_{nm} \\ l_{nm} \end{pmatrix}, \tag{3}$$

with $|s_{nm}| < 1$ for every $n = 1, 2, \ldots, N$ and $m = 1, 2, \ldots, M$. The condition

$$\left\| \begin{matrix} a_{nm} & b_{nm} \\ c_{nm} & d_{nm} \end{matrix} \right\| < 1$$

ensures that

$$u_{nm}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_{nm} & b_{nm} \\ c_{nm} & d_{nm} \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} h_{nm} \\ k_{nm} \end{pmatrix}$$

is a similitude and the transformed surface does not vanish or flip over. The distances between the nodes are δ_x and δ_y, whereas between the vertices A, B, C, D are Δ_x and Δ_y (see Fig. 2a). Constraints are made to map the vertices of the respective domain to the vertices of the region (see Fig. 2b). Since an image is modeled as a function $f(x, y)$, we can apply w_{nm} to an image f by $w_{nm}(f) \equiv w_{nm}(x, y, f(x, y))$. The coordinates of the region vertices are $E(x_{n-1}, y_{m-1}, z_{n-1,m-1})$, $F(x_n, y_{m-1}, z_{n,m-1})$, $G(x_{n-1}, y_m, z_{n-1,m})$ and $H(x_n, y_m, z_{n,m})$.

3.1. Triangular Tiling

The interpolating nodes form NM rectangles (see Fig. 2a or Ref. 11). By affine transformations one can transform only triangle to triangle (there are only nine parameters to be determined). Therefore, each rectangle must be subdivided into two triangles satisfying the relations $A \to E, B \to F, C \to H$ and $D \to G$. Now, an affine transformation can be performed for each triangle separately, i.e.

$$w_{nm}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a_{nm} & 0 & 0 \\ 0 & d_{nm} & 0 \\ e_{nm} & g_{nm} & s_{nm} \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} h_{nm} \\ k_{nm} \\ l_{nm} \end{pmatrix} \tag{4}$$

and

$$\overline{w}_{pq}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \overline{a}_{pq} & 0 & 0 \\ 0 & \overline{d}_{pq} & 0 \\ \overline{e}_{pq} & \overline{g}_{pq} & \overline{s}_{pq} \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} \overline{h}_{pq} \\ \overline{k}_{pq} \\ \overline{l}_{pq} \end{pmatrix}. \quad (5)$$

The coordinates of the vertices A, B, C, D are $A(x_0, y_0, z_0), B(x_0 + \Delta_x, y_0, z_1), C(x_0 + \Delta_x, y_0 + \Delta_y, z_2), D(x_0, y_0 + \Delta_y, z_3)$, respectively. The unknown parameters are

$$a_{nm} = \frac{\delta_x}{\Delta_x}, \quad h_{nm} = x_{n-1} - \frac{\delta_x}{\Delta_x}x_0, \quad d_{nm} = \frac{\delta_y}{\Delta_y},$$

$$k_{nm} = y_{m-1} - \frac{\delta_y}{\Delta_y}y_0,$$

$$e_{nm} = \frac{1}{\Delta_x}[(z_{n,m-1} - z_{n-1,m-1}) - s_{nm}(z_1 - z_0)],$$

$$g_{nm} = \frac{1}{\Delta_y}[(z_{n-1,m} - z_{n-1,m-1}) - s_{nm}(z_3 - z_0)],$$

$$l_{nm} = z_{n-1,m-1} - s_{nm}z_0 - g_{nm}y_0 - e_{nm}x_0,$$

for relation (4) and

$$\overline{a}_{pq} = \frac{\delta_x}{\Delta_x}, \quad \overline{h}_{pq} = x_{p-1} - \frac{\delta_x}{\Delta_x}x_0, \quad \overline{d}_{pq} = \frac{\delta_y}{\Delta_y},$$

$$\overline{k}_{pq} = y_{q-1} - \frac{\delta_y}{\Delta_y}y_0,$$

$$\overline{e}_{pq} = \frac{1}{\Delta_x}[(z_{p,q} - z_{p-1,q}) - \overline{s}_{pq}(z_2 - z_3)],$$

$$\overline{g}_{pq} = \frac{1}{\Delta_y}[(z_{p,q} - z_{p,q-1}) - \overline{s}_{pq}(z_2 - z_1)],$$

$$\overline{l}_{pq} = (z_{p-1,q} - z_{p,q} + z_{p,q-1}) - \overline{s}_{pq}(z_3 - z_2 + z_1) - \overline{g}_{pq}y_0 - \overline{e}_{pq}x_0,$$

for relation (5). The free parameters are s_{nm} and $\overline{s}_{pq}$. This gives 2^{NM} different interpolants.

For a geometric approach to determine the contractivity factors in practice, we let $|\mu|$ be the maximum absolute vertical distance of the function measured from the plane connecting the three vertices of the domain, where the sign of μ is taken as positive or negative depending on whether the maximum distance occurs above the plane or below it, respectively. Let ν be the vertical distance between a function value and the plane connecting the three vertices of the region, where ν is taken positive, if the function value is above the plane and negative otherwise. Then the contractivity factor is given by the ratio ν/μ.

3.2. Rectangular Tiling

The under examination rectangle must satisfy the relations $A \to E, B \to F, C \to H$ and $D \to G$ (see Fig. 2b). The coordinates of the vertices A, B, C, D are $A(x_A, y_A, z_0), B(x_A + \Delta_x, y_A, z_1), C(x_A + \Delta_x, y_A + \Delta_y, z_2), D(x_A, y_A + \Delta_y, z_3)$, respectively. Then, the following system of linear equations arises,

$$z_{n-1,m} = e_{nm}x_A + g_{nm}(y_A + \Delta_y) + s_{nm}z_3 + l_{nm} \quad (6)$$

$$z_{n,m} = e_{nm}(x_A + \Delta_x) + g_{nm}(y_A + \Delta_y) + s_{nm}z_2 + l_{nm} \quad (7)$$

$$z_{n,m-1} = e_{nm}(x_A + \Delta_x) + g_{nm}y_A + s_{nm}z_1 + l_{nm} \quad (8)$$

$$z_{n-1,m-1} = e_{nm}x_A + g_{nm}y_A + s_{nm}z_0 + l_{nm}. \quad (9)$$

The unknown parameters are

$$a_{nm} = \frac{\delta_x}{\Delta_x}, \quad h_{nm} = x_{n-1} - \frac{\delta_x}{\Delta_x}x_0,$$

$$d_{nm} = \frac{\delta_y}{\Delta_y}, \quad k_{nm} = y_{m-1} - \frac{\delta_y}{\Delta_y}y_0.$$

Subtracting Eq. (6) from Eq. (7) we have

$$z_{n,m} - z_{n-1,m} = e_{n,m}\Delta_x + s_{nm}(z_2 - z_3)$$

and subtracting Eq. (9) from Eq. (8) we have

$$z_{n,m-1} - z_{n-1,m-1} = e_{n,m}\Delta_x + s_{nm}(z_1 - z_0).$$

Subtracting the above two equations we have

$$(z_{n,m-1} - z_{n-1,m-1}) - (z_{n,m} - z_{n-1,m}) = s_{nm}[(z_1 - z_0) - (z_2 - z_3)].$$

If $(z_1 - z_0) - (z_2 - z_3) \neq 0$, then

$$s_{nm} = \frac{z_{n,m-1} - z_{n-1,m-1} + z_{n-1,m-1} - z_{n,m}}{z_1 + z_3 - z_2 - z_0}$$

and

$$e_{nm} = \frac{1}{\Delta_x}[z_{n,m} - z_{n-1,m} - s_{nm}(z_2 - z_3)].$$

Finally, subtracting Eq. (9) from Eq. (6) we can determine the value of g_{nm} and l_{nm} by

$$g_{nm} = \frac{1}{\Delta_y}[z_{n-1,m-1} - z_{n-1,m} - s_{nm}(z_0 - z_3)]$$

$$l_{nm} = z_{n-1,m} - e_{nm}x_A - g_{nm}y_A - s_{nm}z_3.$$

We must now consider the case where $(z_1 - z_0) - (z_2 - z_3) = 0$. Then, if $z_{n,m-1} + z_{n-1,m} - z_{n-1,m-1} -$

$z_{n,m} = 0$, the system is indefinite. This means that we can choose any value for s_{nm}, and the values of the other parameters are determined from the above equations. But, if $z_{n,m-1} + z_{n-1,m} - z_{n-1,m-1} - z_{n,m} \neq 0$, then the system is unsolvable. This is a degenerate case which will be arranged inside the proposed algorithm.

In the 1D methods introduced in Mazel and Hayes,[7] the contractivity factors were computed either *geometrically* or *analytically*, with both methods requiring large amount of computations. In the indefinite case, instead of choosing any arbitrary value, we compute the value of s_{nm} using one of the two methods mentioned in Mazel and Hayes[7] properly modified: According to the geometric approach, the value of μ is deducted from the maximum absolute vertical distances between the function (signal) values and the least-square plane of the four domain vertices, while the value of ν is computed from the vertical distances between the function values and the least-square plane of the four section vertices. We use the least-square plane, because the four vertices of a domain or a region are not necessarily coplanar. There is no need to care about the discontinuities between the boundaries of the surfaces, because we are not interested about the surface itself but only about its vertices. The least-square plane of N points is determined by $z = \alpha x + \beta y + \gamma$, where

$$\beta = \left\{ \sum_{i=1}^{N} X_i^2 \cdot \sum_{i=1}^{N} Y_i Z_i - \sum_{i=1}^{N} X_i Y_i \cdot \sum_{i=1}^{N} X_i Z_i + \left[\left(\sum_{i=1}^{N} X_i^2 \cdot \sum_{i=1}^{N} Y_i - \sum_{i=1}^{N} X_i Y_i \cdot \sum_{i=1}^{N} X_i \right) \right.\right.$$
$$\left.\left. \cdot \left(\sum_{i=1}^{N} x X_i \cdot \sum_{i=1}^{N} X_i Z_i - \sum_{i=1}^{N} Z_i \; cdot \sum_{i=1}^{N} X_i^2 \right) \Big/ \left(3 \cdot \sum_{i=1}^{N} X_i^2 \right) \right] \right\}$$
$$\Big/ \left\{ \sum_{i=1}^{N} X_i^2 \cdot \sum_{i=1}^{N} Y_i^2 - \left(\sum_{i=1}^{N} X_i Y_i \right)^2 - \left[\left(\sum_{i=1}^{N} X_i^2 \cdot \sum_{i=1}^{N} Y_i - \sum_{i=1}^{N} X_i Y_i \cdot \sum_{i=1}^{N} X_i \right) \right.\right.$$
$$\left.\left. \cdot \left(\sum_{i=1}^{N} Y_i \cdot \sum_{i=1}^{N} X_i^2 - \sum_{i=1}^{N} X_i \cdot \sum_{i=1}^{N} X_i Y_i \right) \Big/ \left(3 \cdot \sum_{i=1}^{N} X_i^2 \right) \right] \right\},$$

$$\gamma = \frac{-\left(\sum_{i=1}^{N} Y_i \sum_{i=1}^{N} X_i^2 - \sum_{i=1}^{N} X_i \sum_{n=1}^{N} X_i Y_i \right) \beta - \left(\sum_{i=1}^{N} X_i \sum_{i=1}^{N} X_i Z_i - \sum_{i=1}^{N} Z_i \sum_{i=1}^{N} X_i^2 \right)}{3 \sum_{i=1}^{N} X_i^2},$$

$$\alpha = \frac{\sum_{i=1}^{N} X_i Z_i - \gamma \cdot \sum_{i=1}^{N} X_i - \beta \cdot \sum_{i=1}^{N} X_i Y_i}{\sum_{i=1}^{N} X_i^2}.$$

3.3. The Proposed Methods, Their Compression Ratio and a Variation of Them

If we are given $w_1, w_2, \ldots, w_M : I^3 \to I^3$, where $I = [0, 1]$, then we can let $D_m \subset I^2$ denote the domain of w_m and $R_m \subset I^2$ its range. Constraints on the model parameters similar to Ref. 7 on p. 1730 are placed. The first is that the distance between the interpolation points along the horizontal and vertical direction is a constant, i.e. $\delta_x = \delta_y = \delta$. The second is that the distance between the address points along the horizontal and vertical direction is constant, i.e. $\Delta_x = \Delta_y = \Delta$, with the address domains just touching, i.e. that each D_m is exactly equal to a union of some ranges. So, taking $N = M$, we set

$$x_m - x_{m-1} = \delta = y_m - y_{m-1}, \quad m = 1, 2, \ldots, M$$

and

$$\tilde{x}_{m,2} - \tilde{x}_{m,1} = \Delta = \tilde{y}_{m,2} - \tilde{y}_{m,1}, \quad m = 1, 2, \ldots, M,$$

where $\Delta > \delta$ and δ, Δ are chosen *a priori*, for example $\Delta = a\delta$. The number of interpolation regions, MN, is greater than the number of distinct

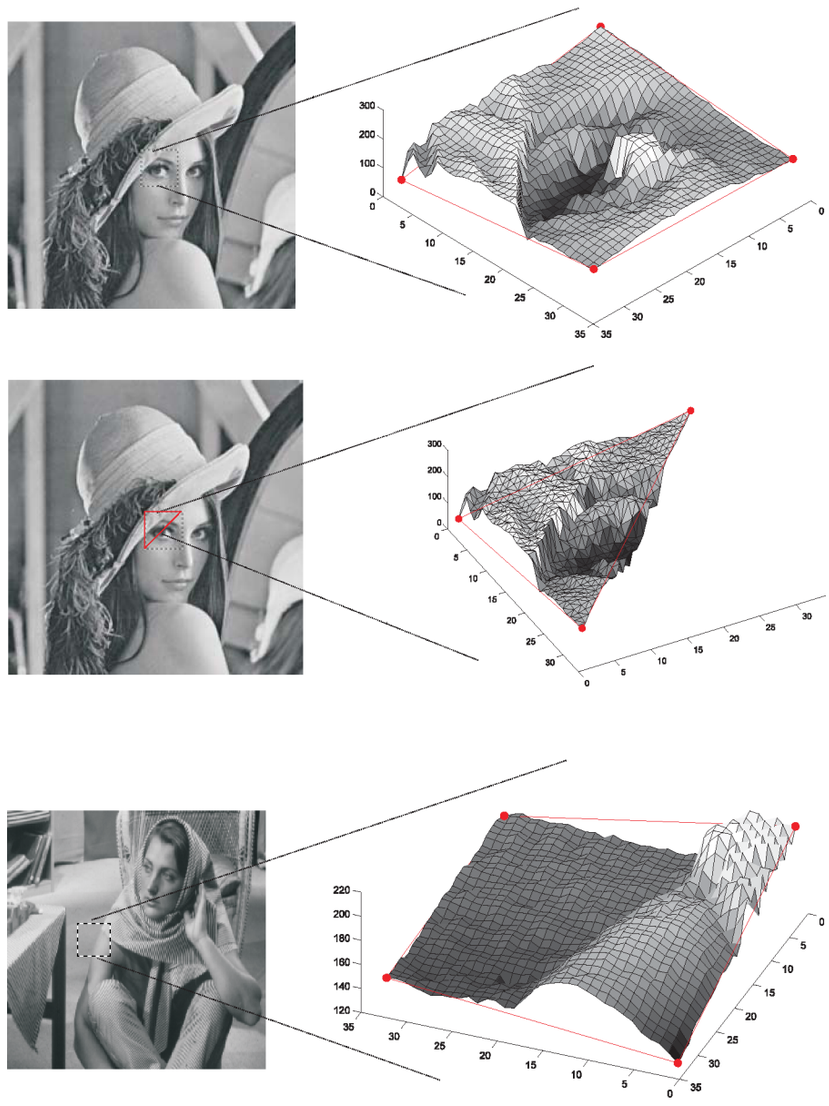


Fig. 3 The image of Lena using rectangular and triangular tiling as well as the image of Barbara using rectangular tiling.

address domains, say M_1. An iterative algorithm for finding the *model parameters*, i.e. the interpolation points (x_i, y_j, z_{ij}), the contractivity factors and the addresses associated with each region of the function is presented in Bouboulis *et al.*[12] Figure 3 illustrates each of the two proposed tiling schemes.

To compress an image using the proposed algorithm we take a set of interpolation points (the vertices of the regions) and a set of integer numbers (the addresses), one for each region, which marks the domain that is best mapped to the corresponding region. It is possible (rather certain) that we will have a set of contractivity factors from the cases in which an indefinite system occurred. To reconstruct the original image, we may use the decompression algorithm also presented in Bouboulis *et al.*[12]

Suppose we are dealing with an image with size $N_1 \times N_2$ pixels in which each pixel can be one of 256 levels of grey and we need to store the model parameters. Since their number is small, we can ignore it for a while in order to simplify our computations. The number of the domains is

$$\left(\frac{N_1 - 1}{\Delta}\right)\left(\frac{N_2 - 1}{\Delta}\right).$$

Each domain is recognized from an address fluctuating from 0 to $[(N_1 - 1)/\Delta][(N_2 - 1)/\Delta] - 1$. Thus,

to store any such address we need

$$k_1 = \log_2([(N_1-1)/\Delta][(N_2-1)/\Delta])$$

bits. We need also $[(N_1-1)/\delta+1][(N_2-1)/\delta+1]$ points of the image as interpolation points. Obviously each such point needs 1 Byte = 8 bits in order to be stored into memory. With these interpolation points the image is divided into $[(N_1-1)/\delta]\times[(N_2-1)/\delta]$ regions. For each region we have to store its relative address showing the domain that is best mapped to this region. Assuming that this address needs k_1 bits we have to store

$$\left(\frac{N_1-1}{\delta}+1\right)\left(\frac{N_2-1}{\delta}+1\right)8+\frac{N_1-1}{\delta}\frac{N_2-1}{\delta}k_1$$

bits.

Before the compression the image needs $N_1 \times N_2$ pixels $\times$ 8 bits per pixel of memory. Thus, the *compression ratio* will become

$$\frac{N_1N_2 8}{\left(\frac{N_1-1}{\delta}+1\right)\left(\frac{N_2-1}{\delta}+1\right)8+\frac{N_1-1}{\delta}\frac{N_2-1}{\delta}k_1} \approx \frac{8\delta^2}{8+k_1}$$

for large N_1, N_2.

The Peak Signal-to-Noise Ratio (PSNR) used to measure the difference between two images, is defined as

$$PSNR = 20\log_{10}\left(\frac{b}{rms}\right),$$

where b is the largest possible value of the signal (typically 255) and rms is the root mean square difference between two images.

A variation of the proposed method is to obtain a larger number of domains, thus increasing compression time and PSNR, but decreasing compression ratio. In this approach, the distance between two consecutive domain vertices instead of being equal to Δ, is equal to $a\delta$. This means that two consecutive domains are not just touching anymore but covering over each other and thus having some common pixels. In this way, we can increase the number of domains thus affecting the compression ratio. Generally, the number of the domains is

$$\left(\frac{N_1-1}{\delta}-a+1\right)\left(\frac{N_2-1}{\delta}-a+1\right)$$

each of which is needed

$$k_2 = \log_2\left[((N_1-1)/\delta-a+1)\left((N_2-1)/\delta-a+1\right)\right]$$

bits. With the second approach we have to store

$$\left(\frac{N_1-1}{\delta}+1\right)\left(\frac{N_2-1}{\delta}+1\right)8+\frac{N_1-1}{\delta}\frac{N_2-1}{\delta}k_2$$

bits. Thus the compression ratio will become

$$\frac{N_1N_2 8}{\left(\frac{N_1-1}{\delta}+1\right)\left(\frac{N_2-1}{\delta}+1\right)8+\left(\frac{N_1-1}{\delta}\right)\left(\frac{N_2-1}{\delta}\right)k_2} \approx \frac{8\delta^2}{8+k_2}$$

for large N_1, N_2.

The new value of the compression ratio is slightly smaller than the compression ratio of the previously mentioned method. Specifically, if we apply these two methods to one image with size 433×433 pixels using $\delta = 9$ and $\Delta = 27$, the compression ratios we expect (according to the above forms) will be approximately 40 for the first approach and 34 for the second approach.

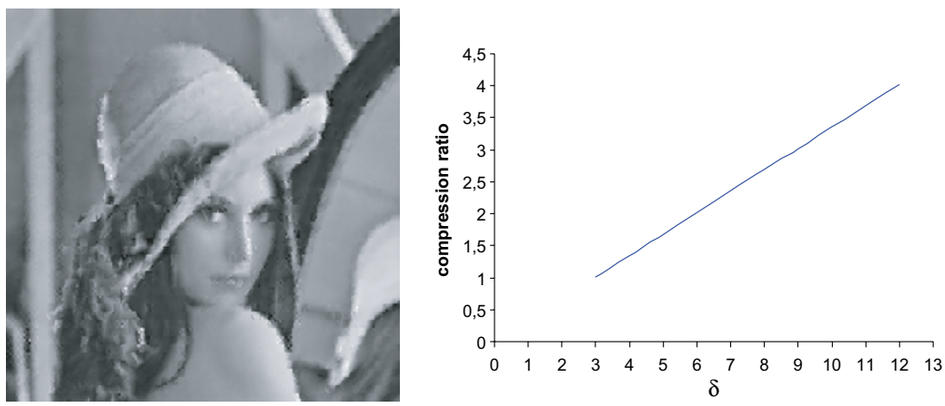
4. COMPARATIVE RESULTS AND EXAMPLES OF THE 2D PIECEWISE SELF-AFFINE FRACTAL MODEL

The original images used as our reference points in the experiments presented here are the $433\times433\times8$ bits and the $2049 \times 2049 \times 8$ bits images of Lena as well as the $2049 \times 2049 \times 8$ bits image of Barbara shown in Fig. 3. For the calculation of the contractivity factors we used the geometric approach; for their quantization we used 6 bit. In order to compute the distances described in the algorithms, we used the ρ_2 (Pythagorean) distance measure.

Firstly, we applied to our images, which are 2D signals, slightly altered versions of the two methods described in Mazel and Hayes[7] for 1D signals. We simply split the images of size $N_1 \times N_2$ pixels into N 1D signals (one signal for each line), each one of them having M data points. Studying Table 1, one can easily observe that the self-affine

Table 1 Comparison of Different 1D Methods for the 433 × 433 Image of Lena.

Method (δ, Δ)	Enc. Time (sec)	PSNR (dB)	Comp. Ratio
Self-affine	194	20.87	1.33:1
Piecewise (3,6)	0.1	44.52	1.01:1
Piecewise (4,8)	0.1	42.8	1.35:1
Piecewise (8,16)	0.1	32.31	2.69:1
Piecewise (9,18)	0.1	30.07	3.02:1
Piecewise (12,24)	0.1	27.56	4.02:1



(a) **(b)**

Fig. 4 **(a)** The image of Lena compressed with the piecewise model for $\delta = 12$ and $\Delta = 24$, **(b)** the relation between the compression ratio and the value of δ.

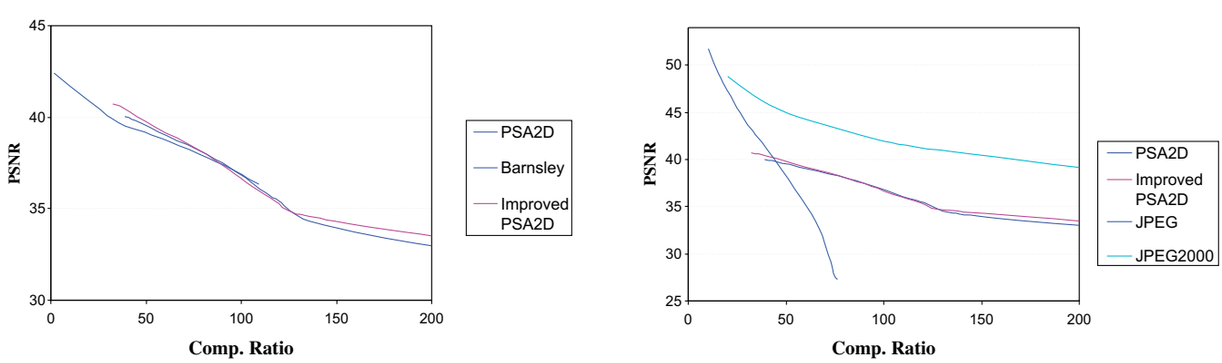
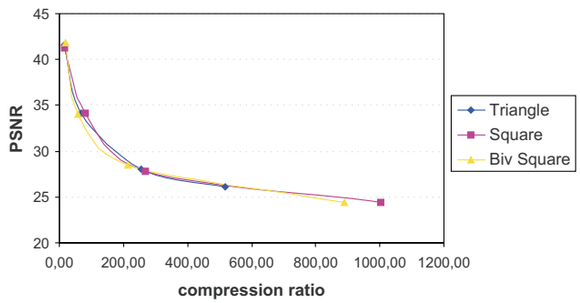


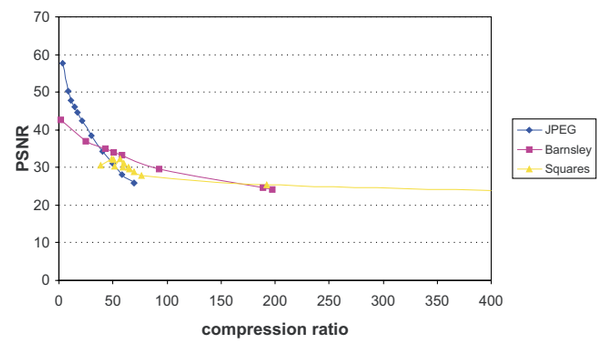
Fig. 5 Comparison of results for 2049×2049 Lena using rectangular tiling.

model needs more time than the piecewise self-affine model, offering less quality. This is expected since real images are not self-affine. On the other hand, when using the piecewise fractal model and considering the real images as 1D signals, we lose a large amount of information (see Fig. 4a). Figure 4b illustrates the linear relation between the compression ratio and the value of δ.

The methods we introduced previously partition one image into regions and domains; so it is necessary that the dimensions of the image, when using the first approach, must be multiples of Δ, while when using its variation must be multiples of δ. For the rest, we must make some modifications to the algorithms. It is obvious that, if the value of Δ is very large, only a few images will be able to be compressed using the first approach. This is one of the reasons why the variation method is preferred. We applied the two approaches to the image of Lena. Figure 5 shows PSNR versus compression ratio for the 2049×2049 Lena using (a) some fractal-based and (b) some fractal-based together with some non-fractal-based methods using rectangular tiling. "PSA2D" is the 2D piecewise self-affine model, while "Barnsley" stands for the method devoloped by Iterated Systems and described in Lu.[6] We used the "JasPer" library which contains an implementation of JPEG-2000 format defined in ISO/IEC 15444-1. For the JPEG format we used the implementation of the well-known Windows software "Corel Photo-Paint." Figure 6 is the performance graphs on the testing images — the first graph shows the difference between the two tilings as applied to Lena, while the second shows the results for Barbara. To measure the time each method needed to compress this image we used a Pentium IV PC with a 2.4 GHz CPU clock running Windows XP. The results are shown in Tables 2 to 4.



(a)



(b)

Fig. 6 **(a)** Comparison between triangular and rectangular tiling for 2049 × 2049 Lena. **(b)** Comparison of results for 2049 × 2049 Barbara using rectangular tiling.

Table 2 Various Parameters of the 2D Method as Applied on the Image of Lena (2049 × 2049).

Parameters $(\delta, \Delta, d_{\max}, \varepsilon)$	Enc. Time (sec)	PSNR (dB)	Comp. Ratio
64, 128, 4, 1	240	40.02	39.00:1
64, 128, 4, 2	150	39.85	44.48:1
64, 128, 4, 3	60	38.97	61.26:1
64, 128, 4, 4	29	37.97	81.72:1
64, 128, 4, 5	20	36.99	97.81:1
64, 128, 3, 4	42	32.86	207.81:1
64, 128, 2, 4	53	28.18	603.65:1
64, 128, 1, 4	60	24.18	1782.76:1

Table 3 Various Parameters of the 2D Variation Method as Applied on the Image of Lena (2049 × 2049).

Parameters $(\delta, \Delta, d_{\max}, \varepsilon)$	Enc. Time (sec)	PSNR (dB)	Comp. Ratio
64, 128, 4, 1	300	40.72	32.38:1
64, 128, 4, 2	210	40.53	37.83:1
64, 128, 4, 3	120	39.47	54.91:1
64, 128, 4, 4	80	38.24	77.11:1
64, 128, 4, 5	54	37.14	93.60:1
64, 128, 3, 4	60	33.7	187.60:1
64, 128, 2, 4	120	29	534.76:1
64, 128, 1, 4	244	24.94	1569.49:1

We achieved a large compression ratio (double that of the well-known JPEG format), but significantly smaller PSNR. For the quantization of the data we applied all the conditions discussed in Subsec. 3.5. The method was implemented by using C++ and we created an executable that can compress any image using the methods mentioned above.

Table 4 Various Parameters of the 2D Model as Applied on the Image of Barbara (2049 × 2049).

Parameters $(\delta, \Delta, d_{\max}, \varepsilon)$	Enc. Time (sec)	PSNR (dB)	Comp. Ratio
64, 128, 4, 1	127	30.57	39.03:1
64, 128, 4, 2	87	30.37	51.75:1
64, 128, 4, 3	77	30	59.06:1
64, 128, 4, 4	57	29.48	65.62:1
64, 128, 4, 5	49	28.87	69.52:1
64, 128, 3, 4	50	25.28	192.56:1
64, 128, 2, 4	49	23.13	592.74:1
64, 128, 1, 4	55	21.42	1788.84:1

5. CONCLUSIONS

Two methods and variations of them for constructing, in general, discontinuous fractal surfaces for data on rectangular lattices are derived. Although our experiments indicate limited applicability for engineering applications, the flexibility of our FIS's may find use in other visualization endeavors.

The 2D piecewise self-affine fractal models described here achieve compression ratios comparable to that of Barnsley's and to JPEG, although their major drawback is that they are less effective at the edges of the image. This happens due to the type of transformations used to map rectangular domains to rectangular subdomains. Choosing a smaller value for δ eliminates this drawback, the PSNR value approaches the PSNR value of the JPEG format, but the compression ratio is decreased significantly.

The compression ratio may be improved by reducing δ close to the image edges since, in the restimage, δ will be relatively large, while the qual-

ity of the reconstructed image would be significantly better. The compression ratio may also increase by observing that some domains are used more often than others and applying some sort of compression to the addresses (e.g. Huffman). Further extension of these methods to construct continuous fractal surfaces is essential (see also Ref. 13) and, when using recurrent bivariate FIS's on rectangular lattices as in Bouboulis *et al.*[14] establishes them as very competitive alternative even to the JPEG 2000 format.

REFERENCES

1. M. F. Barnsley, *Fractals Everywhere*, 2nd edn. (Academic Press Professional, San Diego, 1993).
2. A. E. Jacquin, Image coding based on a fractal theory of iterated contractive image transformations, *IEEE Trans. Image Process.* **1** (1992) 18–30.
3. M. F. Barnsley and L. P. Hurd, *Fractal Image Compression* (AK Peters, Wellesley, MA, 1993).
4. Y. Fisher (ed.), *Fractal Image Compression: Theory and Application* (Springer-Verlag, New York, 1995).
5. B. Wohlberg and G. de Jager, A review of the fractal image coding literature, *IEEE Trans. Image Process.* **8** (1999) 1716–1729.
6. N. Lu, *Fractal Imaging* (Academic Press, San Diego, 1997.)
7. D. S. Mazel and M. H. Hayes, Using iterated function systems to model discrete sequences, *IEEE Trans. Signal Process.* **40** (1992) 1724–1734.
8. M. Ali and T. G. Clarkson, Using linear fractal interpolation functions to compress video images, *Fractals* **2** (1994) 417–421.
9. D. S. Mazel, Representation of discrete sequences with three-dimensional iterated function systems, *IEEE Trans. Signal Process.* **42** (1994) 3269–3271.
10. M. F. Barnsley, J. H. Elton and D. P. Hardin, Recurrent iterated function systems, *Constr. Approx.* **5** (1989) 3–31.
11. D. M. Milošević and Lj. M. Kocić, Fractal surfaces levelling, *FACTA UNIVERSITATIS Ser. Math. Inform.* **13** (1998) 95–107.
12. P. Bouboulis, L. Dalla and V. Drakopoulos, Image compression using recurrent bivariate fractal interpolation surfaces, *Int. J. Bifurc. Chaos* **16** (2006) 1870–1880.
13. J. R. Price and M. H. Hayes III, Fractal interpolation of images and volumes, in *Proceedings of the 32nd Asilomar Conference on Signals, Systems and Computers* eds. (1998) Vol. 2, pp. 1698–1702.
14. P. Bouboulis, L. Dalla and V. Drakopoulos, Construction of recurrent bivariate fractal interpolation surfaces and computation of their box-counting dimension, *J. Approx. Theory.* **141** (2006) 99–117.