Map Search Strategies for IFS Image Compression Algorithms *

Greg Vines
Georgia Institute of Technology
School of Electrical Engineering
Atlanta, GA 30332 USA

and

Monson H. Hayes, III
Georgia Tech Lorraine
2-3 rue Marconi
57070 Metz, France

RÉSUMÉ

Dans cet article, nous proposons et évaluons trois méthodes de recherche de blocs de domaine nécessaires lors du codage d'images utilisant les sytèmes de fonctions itérées. La première approche utilise les transformations de proximité qui se fondent sur la corrélation entre blocs adjacents. La deuxième stratégie utilise la dimension fractale locale de blocs du domaine et du destination afin de diminuer la recherche systématique, et la dernière technique utilise une approche hiérarchisée de mise en correspondance des blocs afin de trouver le meilleur block de domaine pour chaque bloc de destination. Chacune de ces stratégies de recherche a été implémentée sur un codeur à dimension adaptive des blocs. Les résultats sont comparés sur l'image de test standard LENA.

ABSTRACT

In this paper we propose and evaluate three methods for searching for the domain blocks required when coding an image using an Iterated Function System. The first approach uses proximity maps, which are based on the correlation between adjacent blocks in an image. The second strategy uses the local fractal dimension of the range and domain blocks to restrict the search, and the final technique uses a hierarchical block matching approach to find an appropriate domain block for each range block. Each of these search strategies was implemented in an adaptive block size coder and results are compared for the standard LENA test image.

1 Introduction

Iterated Function Systems (IFS's) have recently received a great deal of attention in the literature as a new technique for image coding. Much of this interest stems from the fact that an IFS is simple in form and yet capable of producing complicated images, many of which closely resemble those found in nature [1].

Image coding with an IFS is a block coding technique. The image is first divided into a set of blocks, then each block is coded with a map. There are three main tasks in determining the set of maps to represent an image. The first is how to divide the image into blocks, which form the ranges for the maps. Secondly an appropriate domain must be found for each map. The final task, determining the map parameters, may be accomplished through an error minimization technique such as least squares. Once a set of maps has been determined, the reconstruction process is straightforward [2, 3].

This paper will first give some background on IFS theory, followed by a description of an adaptive block size coder which has been shown to work well [4]. Three search-based algorithms are then introduced for determining the domains for the maps of the IFS. Results are provided comparing these algorithms using the standard LENA test image.

*This work was supported by the Joint Services Electronics Program, Grant No. DAAH-0493G0027.

2 Background

An IFS is a finite set of contraction mappings, w_i, defined on a complete metric space, $\mathcal{U}$, with a distance function ρ, i.e.,

$$\mathrm{w}_i : \mathcal{U} \rightarrow \mathcal{U} \tag{1}$$

for $i = 1, 2, \ldots, P$ with

$$\rho\left[\mathrm{w}_i(\mathbf{x}), \mathrm{w}_i(\mathbf{y})\right] \leq s_i \cdot \rho[\mathbf{x}, \mathbf{y}] \tag{2}$$

for all $\mathbf{x}, \mathbf{y} \in \mathcal{U}$ with $0 \leq s_i < 1$. The notation $\{\mathcal{U} : \mathrm{w}_i, i = 1, 2, \ldots, P\}$ will be used to represent the IFS, and the transformation by all of the maps will be written:

$$W(B) = \bigcup_{i=1}^{P} \mathrm{w}_i(B), \tag{3}$$

where $B \subset \mathcal{U}$.

Every IFS has a unique fixed point given by[1]

$$A = \lim_{n \rightarrow \infty} W^{\circ n}(B) \tag{4}$$

for any $B \subset \mathcal{U}$. This fixed point is called the attractor and is defined below.

Definition 2.1 (Attractor) *The fixed point, $A = W(A)$, is called the attractor of the IFS.*

[1]The notation $W^{\circ n}(A)$ means the n^{th} iterate of A by the function $W()$. For example: $W^{\circ 3}(A) = W(W(W(A)))$.

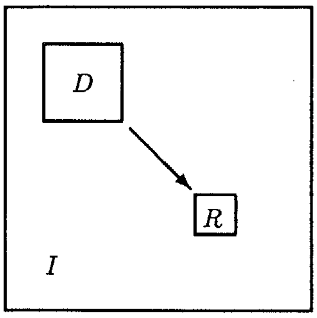

Figure 1: IFS mapping

Thus, to code an image, we wish to find a set of maps whose attractor closely approximates the image. The image is thus coded in terms of the map parameters. The reconstruction of the image from the maps is then straightforward.

Previous work in image coding using IFS's has proceeded by dividing the image into uniform-sized blocks and either, classifying the blocks to reduce the search for domains [5], or creating a library of possible domain blocks [6]. Alternatively, using a fixed domain for each range block has been considered in order to avoid the search entirely [7]. In many cases, in order to increase the flexibility of the map in modeling a block of the image, the domain data is allowed to be transformed by one of eight possible isometries [5].

In this paper we evaluate three search-based strategies. The first uses proximity maps and is based on the notion that the correlation between blocks in an image should be the largest when the blocks are in close proximity to one another. The second strategy uses the local fractal dimension of the range and domain blocks to restrict the search. The final technique uses a hierarchical block matching approach to find an appropriate domain block for each range block. Each of these search strategies was implemented in an adaptive block size coder that is described in the next section.

As in previous implementations, the maps have a fixed contraction factor of 0.5 along each axis. The maps are constructed to transform a domain region, D_i, of the image, I, onto a range region, R_i, as illustrated in Figure 1. The i^{th} map is implemented with the following equation:

$$\begin{aligned} &w_i(x,y) \qquad (5)\\ &= K_{i0} + K_{i1}\cdot x + K_{i2}\cdot y + d_i \cdot I[x_{Di}+2x, y_{Di}+2y], \end{aligned}$$

for $x = 0, \ldots L_x - 1$, and $y = 0, \ldots L_y - 1$, where (x_{Ri}, y_{Ri}) and (x_{Di}, y_{Di}) are the upper left corner of the domain and range blocks respectively, and L_x, L_y are the lengths of the range block along the x and y axis respectively. The same isometries considered in [5] were used with the appropriate changes to equation (5).

For each map, the following parameters must be stored: the three offsets, K_{i1}, K_{i2}, K_{i3}, the scaling factor d_i, the isometry, and the domain address (x_{Di}, y_{Di}). The final constraint to insure a contraction mapping is to restrict the scaling factor such that $|d_i| < 1.0$. The offsets, as well as the scaling factor, are determined using a least squares minimization of the distance between the map's range and the transformed domain data,

$$\xi_i = \sum_{x=0}^{L_R-1}\sum_{y=0}^{L_R-1} \left(I[x_{R_i}+x, y_{R_i}+y] - w_i(x,y)\right)^2. \qquad (6)$$

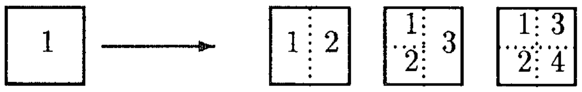

Figure 2: Three ways that a range block may be divided into two to four subblocks. With all possible rotations, there are a total of seven distinct divisions.

3 Adaptive Block Size Coder

Since different regions of an image will vary in detail, it may be expected better coding results will be obtained by allowing the block size to vary. Typically, smaller blocks should be used in those regions of the image that contain information or detail. Therefore, range block are allowed to vary from 4 × 4 to 16 × 16 pixels in size. It has been found that block sizes larger than 16 × 16 tend to result in objectionable blocking artifacts.

The image is initially divided into a uniform grid of 16×16 pixel range blocks. Each range block is then coded and the error determined using equation (6). If the error for any range block exceeds a predetermined threshold, then it is divided. The decision whether or not to divide a block is based on the Mean Square Error (MSE) of each of the four quadrants in the range block. If two adjacent quadrants have an error that is below the threshold, then they are grouped together to form a larger block while subblocks having an error that exceeds the threshold are coded separately. In subsequent divisions, the blocks are never allowed to become smaller than 4 × 4 pixels. This differs from the standard quadtree algorithm in that each block is not necessarily divided into four smaller blocks. Therefore if only half of the block contains data that is difficult to model, it may be separated and coded independently. For example, a 16 × 16 block may be divided into a 16 × 8 and two 8 × 8 blocks. Figure 2 illustrates three different ways that a range block may be divided using from two to four subblocks. Coupled with all possible rotations of this subdivision, there are a total of seven different ways to divide a range block using. In Figure 3, the LENA image has been partitioned using this method and the concentration of smaller blocks can be seen around the more complex portions of the image.

4 Search Strategies

Using fixed range block sizes in a tiling fashion covering the image, the remaining task in the image coder is to determining the appropriate domain block to use for each range block. In most implementations of an IFS image coding system, this search has been the most computationally intensive part of the coder. In an effort to help alleviate this bottleneck, several new search strategies have been implemented as discussed below.

5 Proximity Maps

In an image there is significant correlation between adjacent pixels as well as between adjacent image blocks. In an effort to take advantage of this correlation, it is possible to consider limiting the search to an area immediately around each range block. In limiting the search to the closest 256

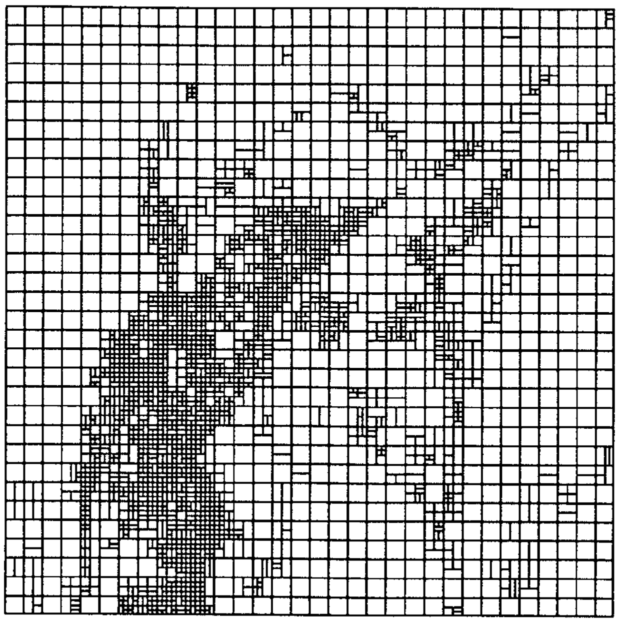

Figure 3: Adaptive block size partitioning of LENA image

domain blocks, only eight bits are required to define the domain address which may be represented in terms of an offset from the range address.

6 Fractal Dimension

In coding an image with an IFS, the image is modeled using a deterministic fractal. One characteristic of a fractal is that it will exhibit the properties of self-similarity or scale invariance [1]. Therefore, the local fractal dimension may be used to classify the domain and range blocks.

When an object is described as being one-dimensional, this means that the it may be divided into N equally sized parts, each of which is $r = \frac{1}{N}$ of the size of the entire object. Figure 4 illustrates this for several objects with integer dimensions. A two-dimensional object may similarly be divided into N pieces, each of which is $r = \frac{1}{N^{1/2}}$ of the entire object. The final example is a three dimensional object, again divided into N parts, each of which is $r = \frac{1}{N^{1/3}}$ of the original.

The trend is that a D-dimensional object can be divided into N equally sized parts that are

$$r = \frac{1}{N^{\frac{1}{D}}} \tag{7}$$

of the original. The relationship between the ratio of the size of the part to the whole, r, the number of blocks, N, and the dimension of the object, D, can be written as

$$Nr^D = 1. \tag{8}$$

Solving for D gives:

$$D = \frac{\log(N)}{\log(\frac{1}{r})}. \tag{9}$$

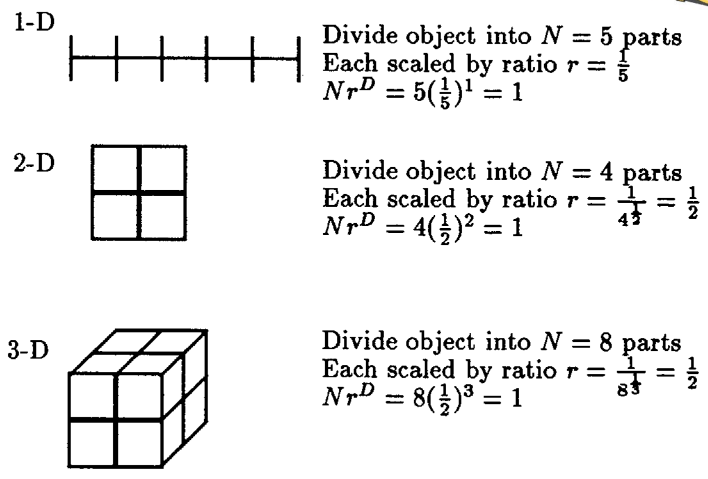
1-D
Divide object into $N = 5$ parts
Each scaled by ratio $r = \frac{1}{5}$
$Nr^D = 5(\frac{1}{5})^1 = 1$
2-D
Divide object into $N = 4$ parts
Each scaled by ratio $r = \frac{1}{4^{\frac{1}{2}}} = \frac{1}{2}$
$Nr^D = 4(\frac{1}{2})^2 = 1$
3-D
Divide object into $N = 8$ parts
Each scaled by ratio $r = \frac{1}{8^{\frac{1}{3}}} = \frac{1}{2}$
$Nr^D = 8(\frac{1}{2})^3 = 1$

Figure 4: Example of intuitive definition of dimension for objects with $D = 1, 2$ and 3. Concept can be extended to fractional dimensions.

The dimension equation (9) is the basis for many fractal dimension measurement algorithms, and is the basis for the Box-Counting Algorithm implemented to calculate the local fractal dimension to compare potential domain blocks for each range block.

By interpreting a 512×512 eight-bit gray scale image as consisting of a collection of $512 \times 512 \times 256$ cubes, it becomes a simple task to count the number of cubes intersected by the image in order to determine the fractal dimension using equation (9). This idea can be extended to compute what is referred to as the local fractal dimension. By restricting the counting of the cubes to a fixed size neighborhood around each point, a measure of the fractal dimension in the proximity of each point may be determined. Because we are interested in matching domain and range blocks, the fractal dimension is computed for $L_R \times L_R$ and $2L_R \times 2L_R$ sized blocks. It is instructive to look at the synthesized image consisting of the local fractal dimensions scaled to the range of $[0, 255]$. Figure 5 shows this image for the 512×512 version of LENA. The edges in the image have a higher fractal dimension than the smooth areas, and from this image it is easy to see why the local fractal dimension has been used as the basis for edge detection algorithms [8, 9].

The local fractal dimension is used as a basis for the domain block search by first computing the local fractal dimension for each range and domain block. Then for each range block, a search is performed over the 256 domain blocks whose local fractal dimension is close to the range block's local fractal dimension. The range of the local fractal dimension used for each range block is adjusted to ensure that the pool of domain blocks contains at least the desired 256 blocks. In this manner, each range block has the same size pool of potential domain blocks to choose from, and these blocks are similar in a fractal sense.

7 Hierarchical Block Matching

In an image, typically there are regions over which the texture will be similar. Therefore, a Hierarchical Block Matching (HBM) approach was implemented in order to search, in an iterative fashion, for a region that is similar to the range block that is being coded. This method begins by

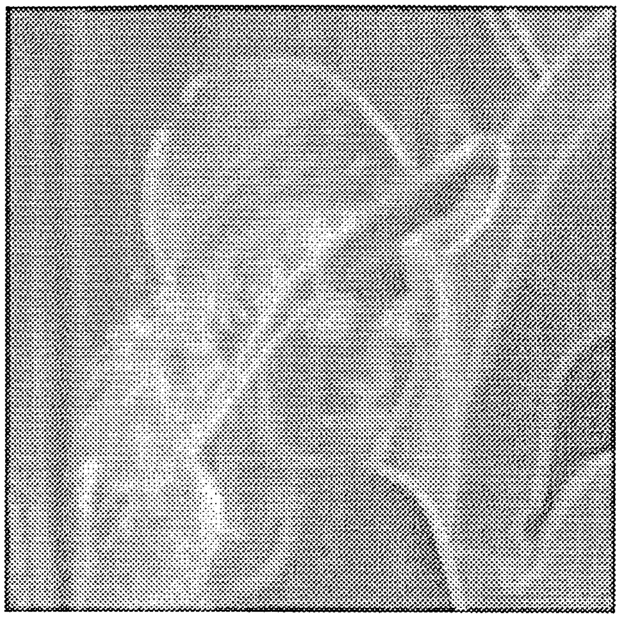

Figure 5: Local fractal dimension of LENA image

initially searching over a widely scattered collection of potential domain blocks. Then, the domain block resulting in the smallest error in coding the given range block is taken as the center for a more narrowly focused search for a domain block. This search is repeated until the desired domain block is found.

For each range block, sixteen surrounding domain blocks were tested in a grid fashion, as shown in Figure 6. The initial distance between horizontal and vertical test points was set to 128 pixels. Each inter–testpoint distance was divided by four to avoid any redundant tests. A typical series of center points for the domain maps to test is illustrated in Figure 6. Beginning with an initial step size of 128 in both x and y and performing four iterations of this algorithm, one quarter of the total number of domain blocks are reachable, with the unreachable domain blocks being those with odd numbered rows and columns. This allows far more domain blocks to be addressed with a minimal number of checks than the approaches discussed previously. In addition, only 64 domain blocks need to be evaluated for the HBM algorithm as compared to 256 for the others.

8 Results and Conclusions

Both the proximity maps and fractal dimension methods were implemented with 256 potential source domains. Thus the results can be compared fairly with regard to the computational cost. The HBM coder was implemented requiring only 64 domains to check, and consequently was faster than the previous two methods.

The results of these three algorithms are given in Table 1 for the 8-bit gray scale 512×512 LENA image. The proximity based search results in the largest SNR and also the lowest bit per pixel rate. Thus, of these methods, the proximity search based approach is the best. The one advantage of the HBM technique is that it is faster than the others because only 64 domain blocks are examined for each range block.

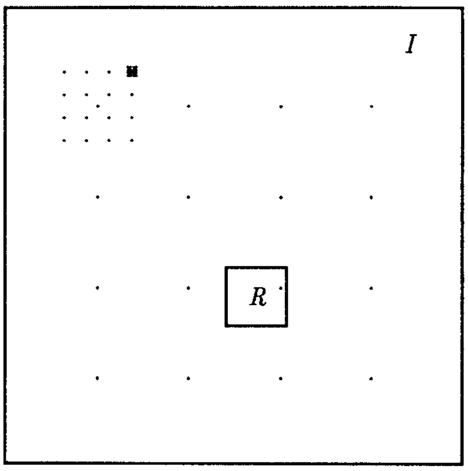

Figure 6: Hierarchical block matching example. Each point represents the center of a domain to be tested.

Search Method	BPP	PSNR
Hierarchical Block Matching	0.55	30.5
Fractal Dimension	0.62	30.5
Proximity Maps	0.47	31.5

Table 1: Image coding results

References

[1] B. B. Mandelbrot, *The Fractal Geometry of Nature.* New York, N. Y.: W. H. Freeman and Co., 1982.

[2] M. Barnsley, *Fractals Everywhere.* New York, N. Y.: Academic Press, Inc., 1988.

[3] S. Lepsøy, G. Øien, and T. Ramstad, "Attractor Image Compression with a Fast Non-Iterative Decoding Algorithm," in *Proc. ICASSP*, vol. 5, pp. 337–340, 1993.

[4] G. Vines, *Signal Modeling With Iterated Function Systems.* PhD thesis, Georgia Institute of Technology, Atlanta, GA, 1993.

[5] A. E. Jacquin, "Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations," *IEEE Trans. on Image Processing*, vol. 1, pp. 18–30, January 1992.

[6] T. A. Ramstad, G. E. Øien, and S. Lepsøy, "An Inner Product Space Approach to Image Coding by Contractive Transformations," in *Proc. ICASSP*, vol. 4, pp. 2773–2776, 1991.

[7] D. M. Monro and F. Dudbridge, "Fractal Approximation of Image Blocks," in *Proc. ICASSP*, vol. 3, pp. 485–488, 1992.

[8] A. P. Pentland, "Fractal-Based Description of Natural Scenes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, pp. 661–674, November 1984.

[9] C.-C. Chen, J. S. Daponte, and M. D. Fox, "Fractal feature analysis and classification in medical imaging," *IEEE Trans. on Medical Imaging*, vol. 8, no. 2, pp. 133–142, 1989.