Local Fractal Compression: an Approach for making Real Fractal Compression

Jacques Blanc-Talon
CTME-GIP/PRO,
16bis, av. Prieur de la Côte d'Or,
94114, Arcueil, France
blanc@etca.fr

Abstract *A preprocessing for giving fractal compression algorithms their full power is proposed. Assuming that IFSs should work much better on self-affine images, it consists in segmenting an image according to the fractal dimension in order to compress subimages showing a constant dimension. First, a dimension map of the image is computed by using a Fast Dimension Transform algorithm. Regular polygons approximate the computed regions. Secondly, the polygons are mapped to square boxes by using conformal mappings which preserve angles. Thirdly, the boxes may be compressed by using a classical fractal technique. Results are shown in the conclusion.*

Keywords: fractal techniques, collage theorem, fractal dimension, conformal map

1 Introduction

Three essential concurrent problems have to be solved at the same time to turn a possible approach into an effective compression technique: namely compressing within *a reasonable time* (according to the application) an image with *a ratio* as great as possible while keeping the *image quality* of the uncompressed file as close as possible to the original one.

Fractal Image Compression, as introduced by Barnsley *et al* [1], made use of some basic properties of fractal geometry which forcast better compression ratios than any other technique. These theoretical possibilities of the fractal way look very promising and it is clear that they have been understood as a promise. Yet, up to now, this promise has not been really kept and, to call a spade a spade, fractal techniques do not offer ratios and qualities as good as those yielded by other techniques. For instance, we tested a software product based on wavelet functions whose current color image compression ratio reach 1,000 with a good quality; such a ratio is still far from fractal algorithms capabilities. Moreover, if the decompression time of the fractal code is good since it only involves linear algebra, it is indecent to talk about the compression time which may be up to 5,000 greater (depending on the approach).

There might be a couple of reasons to that. About the compression time, the parameter space to explore is huge and the possibility tree is enormous, especially when Vrscay's algorithm ([2, 3]) is used. It seems rather difficult to prune the tree significantly even if recent advances have been made (for instance [4],[5],[6]).

About the compression ratio, one has to admit that block-coding compression scheme [7] for instance does not take into account any geometrical property of the original image at all. In other words, real images are rarely fractal in the block-coding scheme sense whenever simple transformations are used. The converse could also be true: nature seems to reject the same kind of patterns at both macroscopic and microscopic levels. Thus, the introduction of Generalized Square Isometries [8] seemed to be a good step in the direction of a deeper understanding of the kind of transforms to be used in fractal compression.

Our conclusion is that real images are *not* fractal enough. If it was possible to *fractalize* an image without loosing its content, the moment approach should work much better. The approach described in this paper consists in compressing homogeneous regions (according to the fractal dimension) separately in order to justify the use of IFSs. In a first step, the image is segmented into regions (polygons) whose self-similarity dimension is assumed to be constant. These polygons are mapped into the unit square by using a conformal map. The compression algorithm is then applied to the collection of squares. Decompressing an image consists in reversing the process and gluing the puzzle.

2 Self-Affine Fractal Segmentation of Images

The fractal segmentation of an image consists here in splitting the image into regions whose respective fractal dimension is as constant as possible. For practical reasons, several assumptions are made; as a result, the regions are replaced by triangles whose edges approximate the region boundaries. The whole segmentation is achieved in 4 steps.

First, the dimension of every pixel within the image is computed. In a recent study, Soille and Rivest [9] showed how dependent a measure of the dimension D is with respect to the content of the image. However, an accurate estimate of D is not needed here since i) regions must be approximated by polygons and ii) the spatial derivative of the dimension is the only information being used. Instead of using a sophisticated algorithm, we used the following one called *Fast Dimension Transform*.

Let I be a square image of size $M = 2^\Gamma$ and $I(x,y)$ its grey values (in the general case of a rectangular image, the greatest square one within is selected). $D(p)$ denotes the *local self-affine dimension* of pixel $p = (x,y)$ and $D(I)$ the *dimension transform* of the image, i.e. the collection of all local values. $D(p)$ is computed in a square neighborhood of size $m = 2^\gamma$ around p (and so is $D(I)$ in the greatest square image included in I) by means of the log-log fit formula:

$$D(p) = 2 - \frac{\gamma\Sigma(\epsilon S) - \Sigma(\epsilon)\Sigma(S)}{\det R} \qquad (1)$$

with in particular $\Sigma(u) = \sum_{i=1}^{i=\gamma} \log u_i$ and $\det R = \gamma\Sigma(\epsilon^2) - (\Sigma(\epsilon))^2$ is the determinant of the regression matrix R. In this notation, ϵ is the sequence of grid sizes $2^k (k = 0, 1, \cdots, \gamma-1)$ and S the one of related surface estimations of the image whose supports are $((x,y), (x+2^k, y), (x+2^k, y+2^k), (x, y+2^k))$.

Starting from a computed value of D in p, it is thus possible to interpolate D' in $p + \Delta p$ by using the following Taylor's expansion:

$$\Delta D = \frac{1}{\det R}\left(\gamma\frac{\partial\Sigma(\epsilon S)}{\partial p} - n\Sigma(\epsilon)\frac{\partial\Sigma(S)}{\partial p}\right)\Delta p \qquad (2)$$

Both $\Sigma(S)$ and $\Sigma(\epsilon S)$ differ in $p + \Delta p$ of the previous values in p from only functions of the bound values of $I(p)$; these values are available from (1). One must notice that $\det R$ does not depend on p provided that the image support is always mapped to the domain $[0, 2^k[\times[0, 2^k[$ whatever the position of p is. Taking all these remarks into account, the fast algorithm for computing the dimension transform is twofold. It consists in computing the local dimensions of I at nodes $\{(m/2+im, m/2+jm), 0 \le i,j \le \Gamma - \gamma$ first, then in interpolating the values between nodes by using recurrence formula 2. In our experiment, $\gamma = 4$ which yields 4 points for the log-log fit.

The next step consists in segmenting I in homogeneous regions (or, equally, in edges) with respect to the dimension, i.e. finding regions with D as constant as possible. This is very simply performed by applying a well-known Canny-Deriche operator to the image $D(I)$ (whose values $D(x,y)$ have been normalized to $[0, 255]$) so as to get connected edges. The resulting image is called $Ft(I)$.

The last step consists in finding simple polygons, and more precisely triangles, which roughly approximate the segmented regions. Polygon edges are recursively replaced by straight lines; their intersections determine triangles. The triangles must not be too small and they must not be too numerous. The first criterion is equivalent to stopping the algorithm under a minimum surface s_m and the second one over a maximum region number n_m; we chose $s_m = 64$ and $n_m = 16$ but any other values may be as convenient. The following recursive algorithm is thus performed on $Ft(I)$.

Let $C(M_1M_2)$ be an edge within $Ft(I)$. The error between the edge and the segment line $\overline{M_1M_2}$ is given by

$$E(M_1, M_2) = \sum_{M \in C(M_1M_2)} \left\| M - proj_{\overline{M_1M_2}} M \right\|$$

where *proj* is the orthogonal projection onto $\overline{M_1M_2}$. Whenever both stopping criteria are false, the edge is replaced by two sub-edges $C(M_1M_{12}), C(M_{12}M_2)$ which yields a couple of new triangles; M_{12} is the point maximizing the distance to its projection. This algorithm is called recursively until it stops and the interior of the region is split into 2 triangles.

The result of this algorithm is a kind of binary triangulation of the image I. This image is used as a logical mask for extracting the gray levels of the regions from the original image I.

3 Conformal Mapping and Local Image Compression

A conformal map is a one-to-one map acting in the complex plane $\mathbb{C}$ and whose main feature is to preserve angles, i.e. if two curves are perpendicular at a given point, they will remain perpendicular after conformal mapping. An important consequence is that the mapping of a connected set is still a connected set. According to these properties, we conjecture that *the mapping of a self-affine fractal is still a self-affine fractal.* Thus, the purpose of mapping homogeneous regions to square blocks is to feed a classical fractal compression algorithm with mere fractal regions.

Let us briefly expose the main results needed to understand the method. The conformal transformation ϕ of a triangle into a square may decomposed as the transformation $\phi_\triangle$ of a triangle into the higher semiplane ($\Im z \geq 0$), followed by the transformation ξ of the semiplane into the unit square. The inverse of the first conformal map is the Schwarz-Christoffel transformation given by:

$$\phi_\triangle{}^{-1}(z) = A \int_0^z t^{\alpha_1 - 1}(1-t)^{\alpha_2 - 1} dt + B$$

where the curvilinear integral is taken over the line segment joining the origin and z. The triplet $\{\alpha_1\pi, \alpha_2\pi, (1-\alpha_1-\alpha_2)\pi\}$ is the set of the angles of the triangle; A, B are complex values determined by the vertices and the location of the triangle. The inverse of the second conformal map is:

$$W^{-1}(z) = z_0 + \frac{e^{i\frac{\pi}{4}}}{C\sqrt{2}} \int_0^{e^{i\frac{\pi}{4}}z} \frac{dt}{\sqrt{1-t^4}}$$

where $z_0 = 0.5 + 0.5i$ and C is the real value $\frac{\Gamma(1/4)\Gamma(1/2)}{4\Gamma(3/4)}$ (Γ is the Euler function).

In our application, both functions $\phi_\triangle(z)$, $W(z)$, $\phi_\triangle{}^{-1}(z)$ and $W^{-1}(z)$ are computed by using Laurent expansions. Nevertheless, it is difficult to summarize in a few lines the details; the interested reader may refer to [8]. These series yield discrete formulas involved in the practical computation of the triangle-to-square mappings, and conversely. Every pixel (x, y) is written $z = x + iy$ and its new location computed; its gray level is interpolated from the gray levels of the nearest neighbors, each neighbor having a weight inverse proportional with the square of the spatial distance.

The final step of the compression scheme is now to encode the squares by using a classical algorithm. Given a square, the accuracy of the encoding is a function of the relative surface of the initial triangle within the whole image. Thus, parameters of the encoding algorithm may be set to practical values.

4 Decompressing an Image

The decompression procedure consists exactly in the converse of the compression one. In particular, every compressed square is fitted with the coordinates of its ancestor so as to be remapped to its right position. As exemplified above, the triangle-to-square maps involved in this preprocessing are approximated by discrete expansions for computability reasons. Some oblique artefacts may appear along the borders of contiguous regions.

In the present results, local interpolations have been performed on the borders, which slightly weakens the SNR (of about 1 dB).

5 Conclusion and Results

The main feature of our approach is that the compression ratio is improved locally (for every triangle) and seems to be improved in general (of about 10 %). However, the global ratio depends on the number of squares, their respective compression ratios and the binary code needed for encoding the set of parameters of the compression algorithm. Actually, there must be an optimum for every image.

Among other intrinsic limitations, it is honest to recall that our method takes more time to decompress an image since the algorithm must be run on every box before putting them together. The problem of oblique artefacts has mainly to deal with the approximation of the maps and will be solved in the next version.

The main drawback of the approach is that it is totally useless in the case of images which do not have any significant fractal zone (for instance, the Rubik's cube image). Such *flat* images are a collection of regions poorly compressed, and a lot of time is lost in the preprocessing. However, it might be turned off by a *fractal switch* and triggered when necessary, bringing to a fractal compression algorithm the percent needed to cross the finishing line.

I would like to thank Dan Popescu from DIT, Canberra (Australia) for fruitful discussions and encouragement.

References

[1] M. Barnsley. *Fractals everywhere.* Academic Press, 1988.

[2] E.R Vrscay. Moment and collage methods for the inverse problem of fractal construction with iterated function systems. In H-O. Peitgen, J.M. Henriques, and L.F. Penedo, editors, *Fractals in the Fundamental and Applied Sciences.* Elsevier - North Holland, 1991.

[3] Bruno Forte and Edward R. Vrscay. Inverse problem methods for generalized fractal transforms. Technical report, University of Waterloo, Canada, March 1996.

[4] Dietmar Saupe and Hannes Hartenstein. Lossless acceleration of fractal image compression by fast convolution. In *International Conference on Image Processing*, Lausanne, September 1996.

[5] Eric Amram and Jacques Blanc-Talon. Fractal image compression: an efficient acceleration technique. In *CISST*, Las Vegas, NE, 1997.

[6] David John Nettleton and Roberto Gari gliano. Reductions in the search space for deriving a fractal set of an arbitrary shape. *Jour. of Mathematical Imaging and Vision*, 1996.

[7] Arnaud E. Jacquin. A novel fractal block-coding technique for digital images. In *Int. Conference on Acoustics, Speech and Signal Processing*, pages 2225–2228. IEEE, 1989.

[8] Dan Popescu, Alex Dimca, and Hong Yan. Generalized square isometries - an improvement for fractal image compression. In *8th Int. Image Conf. on Image Analysis and Processing*, San Remo (It), 1995.

[9] Pierre Soille and Jean-F. Rivest. On the validity of fractal dimension measurements in image analysis. *Visual Communication and Image Representation*, pages 217—229, September 1996.