Adaptive Delaunay Triangulation for Attractor Image Coding

F. Davoine and J.-M. Chassery

Laboratoire TIMC-IMAG, Equipe Infodis
Institut Albert Bonniot, Faculté de médecine, Domaine de la Merci
38706 LA TRONCHE Cedex, FRANCE
GDR 134 TDSI CNRS

Abstract

The principle of attractor image coding presented in this paper is based on the theory of IFS (Iterated Function Systems). The algorithm exploits piece-wise similarities between blocks of different sizes. To improve the algorithms based on regular and square blocks, we propose an adaptive Delaunay triangulation of the image support. The originality of the method is to map the triangles on specific parts of the image in order to have lots of similarities between them.

1 Introduction

The main purpose of attractor image compression [1] is to find resolution independant models of images. A.E. Jacquin first proposed an automatic algorithm to code "real-world" images. (He published recently a review on fracal image coding [5]). His main idea was to state that an image is formed of transformed copies of parts of itself. He showed that partitioning images into square blocks, and designing discrete transformations acting blockwise, approximates the original image by a self-similar one. He refered to this approach as fractal block coding. Many researchers subsequently compressed images with a similar approach [4]. Another possibility is to compute the parameters of the transformations directly from local invariant features of the image. This last idea has been succesfully implemented on a pixel row of an image [6]. The originality of our approach is the use of Delaunay tessellation to partition the image support. The approach benefits from properties of triangulation, such as adaptivity, and non-rigidity. It is possible to map triangles on specific parts of the image, in order to cover edge, shade, or midrange regions. It is possible to map triangles on specific parts of the image, in order to cover edge, shade, or midrange regions.

2 Overview of the encoding principle

Let us consider a grey level image A to be encoded. We see it as the attractor of a contractive operator W, obtained by $\lim_{n\to\infty} W^{\circ n}(B) = \mathcal{A}_t, \forall B \in X$. The collage theorem, proposed by M.F. Barnsley, makes it possible to find an operator W whose fixed point is close to a given one. It states that if it is possible to cover the image A by transformed parts of itself such that the result is close to the image A, then the operator W approximately defines the image A. The operator W returns a result close to the original image A, starting with any image as explained above. The covering being not exact, we have an approximation of the original image during the decoding phase. A.E. Jacquin partitioned the image into non-overlapping blocks R_i and defined the operator W as: $W(A) = \bigcup_{i=1}^{N} \omega_i(A \cap D_i)$, where $A \cap D_i$ is the restriction of A to the part D_i of the image A. W is composed of a collection of local contractive transforms ω_i and returns the union of transformed parts of the image A. Our encoding algorithm consists in using a two-level triangulation of the image support. One of the two levels returns blocks D_i (partition D), and the other returns blocks R_i of a smaller or equal area (partition R). The partition R generates non-overlapping blocks. More details will be given on their construction in Section 4. The algorithm finds, for each block R_i, a transformed domain block D_i which is very close to R_i. The block D_i can be found anywhere in the partition D. Then, if we perform the collage of the transformed blocks D_i on the blocks R_i, we have $W(A)$ nearly equal to A. In this case, the collage theorem is satisfied and the operator W encodes the image.
The mappings ω_i used in our implementation can be written as:

$$\omega_i \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} e_i \\ f_i \\ o_i \end{pmatrix}.$$

The similarity (in terms of grey level) between the block R_i and the transformed block D_i is measured with the square error (SE) given by: $d(A \cap R_i, \omega_i(A \cap D_i)) = \sum_{n=1}^{n_0}(s_i.d_n + o_i - r_n)^2$ where d_n and r_n are respectively the intensities of the pixels in the blocks D_i and R_i, and n_0 the number of pixels included in $A \cap R_i$. The contractivity of the transform W is in that case controled by the parameters s_i. They have to be less than 1, in order to insure contractivity.

3 Decoding from fractal code

The fractal code for the compressed image A is composed of a collection of N contractive transforms ω_i(x,y,z). The decoding consists in iterating the operator W, starting with any initial image B. The reconstructed image is given by $A \simeq \lim_{n\to\infty} W^{o_n}(B)$. One iteration consists in scanning the blocks R_i (in the same order as during the coding step) and in applying the affine transformation ω_i on their corresponding domain block D_i. We usually need 8 to 12 iterations to converge to the original image. The number of iterations depends on the resolution of the image.

4 Adaptive partition of the image support: Delaunay triangulation

To construct the fractal code, we need to partition the image support. Different partitions have been proposed, using regular squares, quadtrees, rectangles and triangles. This chapter is concerned by Delaunay triangulation [8] which offers good properties of regularity. The unconstrained orientation of triangles makes it possible to have a data dependant partition. Essentially there must be a high density of triangles in regions including details in the image. This principle can be understood in terms of variance, or gradients computed on pixels belonging to the interior of triangles. In terms of data structure, a graph environment is used in order to facilitate the manipulation of the triangles. The algorithm (an incremental approach)

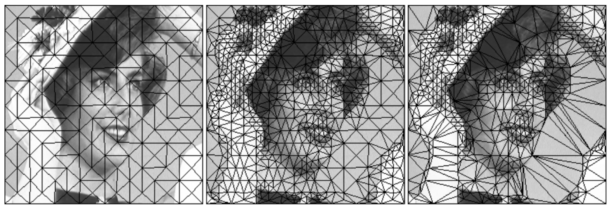

Figure 1: From left to right: the initial known regular triangulation, the end of the split step and the merge step

works by local modifications of the triangulation when a new triangle is inserted. It proceeds in two steps which are called Split and Merge [2, 3].
We start with a small number of points (triangles vertices) regularly distributed on the image support (Fig. 1). The split step consists in adding a point on the barycenter of each non homogeneous triangle (variance or gradient criteria). The split process continues until convergence. Thus it stops when either the triangles are homogeneous or the surfaces of the triangles are less than a given threshold. The merge step consists in deleting neighboring triangles with similar mean grey levels. Those triangles are suppressed from the graph. We note that the number of triangles is inferior to the number of squares or rectangles obtained in previous algorithms.

5 Compression Ratio

To encode an image we need to store the coefficients of the N mappings ω_i composing the operator W. One mapping is needed for each block R_i in the partition R. The Delaunay tessellation is coded in a graph environment.
Thus, for one transformation ω_i, it is only necessary to code:
• the position of the triangle D_i in Delaunay graph
• the orientation for the collage (6 possibilities to map a triangle onto another)
• the scale (s_i) coefficient
• the offset (o_i) coefficient
It has been verified that 6 bits to code s_i and 6 bits for o_i are sufficient in the general case to provide a visualy good reconstructed image. The orientation for the collages needs 3 bits to be coded.
In addition, we must code the way to obtain the image adapted partitions R and D. Since we always start with a known regular distribution of points to construct the Delaunay triangulation, it is only usefull to code the split and merge process, with 1 bit for a split of a triangle and 1 bit for a merge of a triangle: the division of a triangle during the split steps is coded with a "1", the non-division with a "0". The obtained string of binary codes can moreover be compressed with classical techniques like the Run Length Coding. For example, the image "femme" partition R in Fig. 2 is coded by an uncompressed string of 1564 bits.
The total number of bits necessary to code the operator W is thus given by: $N_{tot} = N(6 + 6 + 3 + M) + X$ with N = total number of blocks R_i in the partition R, X = number of bits to code the two partitions processes and $M = \log_2$(nb. of blocks D_i) = number of bits to code the position of the block D_i in the graph of triangles D.

6 Results

We have presented results on $256 * 256$, 8 bpp grey level images "femme" and "peeper" (fig. 2). First we constructed the operator W of the image on a Delaunay triangulation, using the split and merge approach. The resulting tessellation returns irregular triangles which are image constrained. Large triangles appear in homogeneous parts of the image. There are 1024 triangles in the partition R for the image "femme", 1225 triangles for the image "peeper", and 576 triangles in the partitions D for the two images. The search for the matching of range blocks is done among a limited number of triangles D_i and not anywhere in the image. In order to preserve a good fidelity of reconstructed image with respect to the original, the two partitions have to present a maximum number of similar blocks, and also to be image dependant. The unconstrained orientation of the triangles gives good results. The decoded image does not present discontinuities, like blocking artifacts as observed with fixed square block based methods.

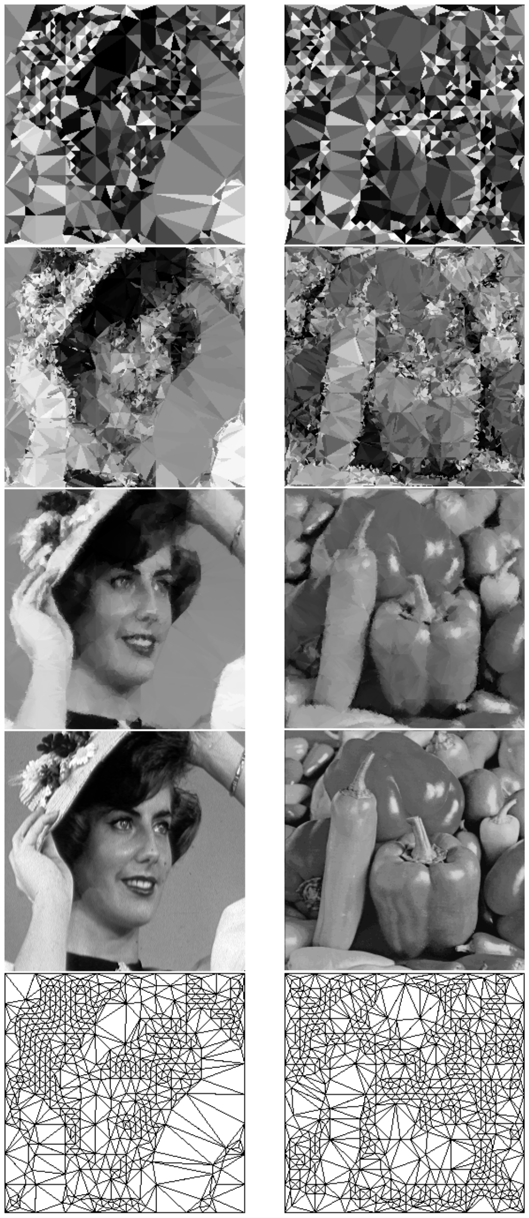
Figure 2: Decoding results. From top to down: first, second and tenth iteration starting with a white image, the original image, and the associated partition R. Left: Compression ratio = 18.3, PSNR = 27.4 dB. Right: Compression ratio = 15.6, PSNR = 27.3 dB

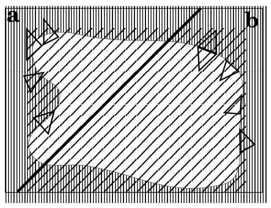

Figure 3: a: triangles obtained with the Split and Merge algorithm, b: containt triangles

7 Discussions about perspective works

The coding algorithm can be improved by a classification scheme of the triangles. The classification creates different types of triangles. The use of a contour image allows separation of triangles in two classes: edge triangles and others. Thus, the search, for a given edge range block, is done through a list of edge domain blocks.
The split and merge algorithm as presented returns triangular patches covering homogeneous and edge regions of the image. This particularity is due to the use of a variance criteria during the split process. It would be more cunning to force the triangles edges onto, or close to the image edges, in order to increase the similarities between blocks. It has been verified that if the edge of a transformed triangle $w_i(D_i)$ is close to an edge, this triangle is often mapped onto another triangle R_i close to the same edge, with a similar gradient vector. On figure 3.a we notice that if a triangle D_i covers an edge, its neighbours are not similar in terms of minimizing the distance between the transformed block $w_i(D_i)$ and its target block R_i. When we visually compare two triangles, only the gray level surface inside these triangles is important, and not the grey level mean or variance. In consequence, when the triangles edges are close to the image edges (Fig. 3.b), we obtain a set of similar blocks [7]. The collages are more efficient. In a future work, to construct this partition, we propose to start with a regular distribution of points on the image support and to add other points on the image edges. Next, the split process runs on the initial triangles with a constraint to let unchanged the triangles close to an edge.

References

[1] M.F. Barnsley and L.P. Hurd. *Fractal Image Compression.* AK Peters Ltd., Wellesley, 1993.

[2] E. Bertin, F. Parazza, and J.-M. Chassery. Segmentation and measurement based on 3D Voronoi diagram: Application to confocal microscopy. *Computerized Medical Imaging and Graphics*, 17:0–8, 1993.

[3] X. Chen and F. Schmitt. Split-and-merge image segmentation based on Delaunay triangulation. In *Proc. of The 7th Scandinavian Conf. on Image Analysis, Aalborg, Denmark*, pages 910–917, aug. 13-16 1991.

[4] Y. Fisher (Ed.). Fractal Compression: Theory and Application to Digital Images. Springer Verlag, New York, 1994, with contributions from I. Baharav, B. Bielefeld, R.D. Boss, K. Culik II, F. Dudbridge, Y. Fisher, B. Jacobs, J. Kari, E.D. Karnin, S. Lepsøy, L.M. Lundheim, D. Malah, S. Memlove, G.E. Øien, D. Saupe, G. Vines.

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

[6] W.G. Kropatsch, M.A. Neuhausser, I.J. Leitgeb, and H. Bischof. Combining Pyramidal and Fractal Image Coding. In *Proc. 11th ICPR, The Hague, The Netherlands*, 3:61–64, 1992.

[7] M. Melkemi, and J.-M. Chassery. Edge-Region segmentation process based on Generalized Voronoi diagram representation. In *Proc. 11th ICPR, The Hague, The Netherlands*, 3:323–326, 1992.

[8] D.F. Watson. Computing the n-dimensional Delaunay tesselation with application to Voronoi polytopes. *Comput. J.*, 24(2):167–172, 1981.