Diffusion Limited Aggregation VS Iteration Function Technique for Fractal Graphics.

Yahia AL - HALABI
Sabah M. A. MOHAMAD
Jinan A. W. FAIDHI
Computer Sience Dept., Amman University

ملخص

الصُور غير المكتملة تُزوِّد الصور الطبيعية المُجسَّمة بِتَقنيةٍ لا يُمكن تواجدها بالهندسة الاقليدية.

يُركز هذا البحث على تنفيذ تَقَنيَتين ترتكزان على الصور غير المكتملة في تصوير الأشياء الطبيعية.

أسلوب الاسهاب المحدد يُحاكي الصُوَر الطبيعية المجَسمة عن طريق عملية تطوير ونُمو مستمر للصورة، بينما يُحاكي الأسلوب التكراري الوظيفي في تمثيل الصُور الطبيعية عن طريق تقديم أفكار مُجرَّدة باستخدام المعاملات المعروفة.

كما يُقدم البحث حلولاً لِمشكلة اسلوب التَوازن والقياس وتكبير الصور الطبيعية.

ABSTRACT:

Fractal Graphics provides mechanisms for modelling natural images that can not be represented with Euclidean Geometry. This research paper concentrates upon implementing two mechanisms based on Fractal Graphics for modelling natural

objects. The diffusion limited aggregation method model natural objects by a growth process while the iterated function technique model natural images by representing their abstraction using parameters. Moreover this paper presents solutions to the problem of natural objects scaling and zooming techniques.

KEYWORDS: Fractal Objects, Diffusion Limited Aggregation, Iterated Functions.

(1) FRACTAL GEOMERTY FOR MODELLING NATURAL PATTERNS:

What is a fractal? Generally, it is a class of graphics objects or a set of points [2]. Fractals in physics explains phenomena such as how some solids crystalize and how air bubbles move in fluids. Understaning fractals is indeed quite important to other fields, especially for material engineering and several manufacturing techniques. Although fractals are non - euclidean objects, they do posses a perfect symmetry. Actually it is the "increasing amount of evidence suggests that nature's love fractal shapes.

Traditional computer graphics encodes images in terms of simple geometrical shapes: points, line segments, boxes, circles, and so on. More advanced systems use three dimensional elements, such as spheres and cubes, and add color and shading to the description. Graphics systems founded on traditional geometry are great for creating pictures of man - made objects, such as bricks, wheels, roads, buildings, and cogs. However, they don't work well at all when the problem is to encode a sunset, a tree, a lump of mud, or the intricate structure of a black

spleenwort fern.

Indeed one can achieve good results by digitizing any natural picture, using several available software packages (e. g. Freehand Painting, Macpaint, Fullpaint, Icon Editor) based on techniques such as spraying, smudging, cut and past [2]. Think about using a standard graphics system to encode a digitized picture of a cloud: You'd have to tell the computer the address and color attribute of each point in the cloud. But that's exactly what an uncompressed digitized image is a long list of addresses and attributes.

To escape this difficulty, we need a richer library of geometrical shapes. These shapes need to be flexible and controllable so that they can be made to conform to clouds, mosses, feathers, leaves, and faces, not to mention waving sunflowers and glaring arctic wolves. Fractal geometry provides just such a collection of shapes. For a hint of this, glance at the pictures in The Fractal Geometry of Nature by Benoit Mandelbrot, who coined the term fractal to describe objects that are very "fractured" [3]. Mandelbort objects posses certain repetition of a pattern and that pattern is scale - inveriant: at each stage any part of the pattern that has one third of the diameter of the whole looks exactly like the whole. Scale - inverience is a symmetry of fractals. Just as round objects are symmetric under rotation. Fractals are symmertic under changes of scale.

Using fractals to simulate landscaped and other natural effects is not new; it has been a primary practical application. For instance, through experimentation, you find that a certain fractal generates a pattern similar to tree bark. Later, when you want to

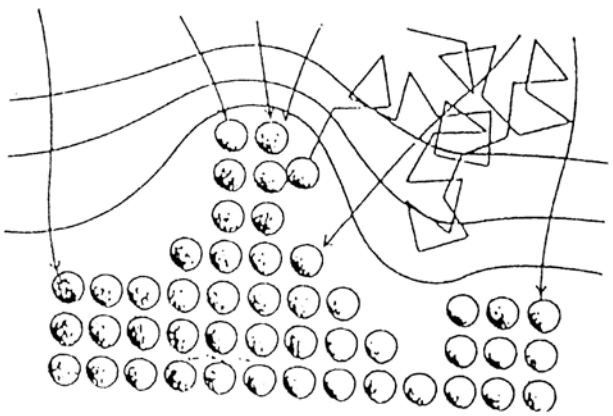
render a tree, you put the tree bark fractal to work. Indeed, what is new is the ability to extract from a natural image the fractals that will imitate it to any desirable degree of accuracy. By doing that, we should achieve a mechanism for fractals modelling and simulation as well as rending up with a highly compressed data set for reconstructing the original image. In this paper, we present two methods to achieve this aim; namely, the diffusion limited aggregation and the iterated function system.

(2) DIFFUSION LIMITED AGGREGATION PROCESS FOR FRACTAL MODELING:

Although "T. A.' Witten" of the Exxon Reserch and Engineering Company had proposed in 1983 a mechanism for fractal modelling and simulation, in which he calls diffusion - limited aggregation, not very much work has been done to implement his proposal [4]. According to Witten's modelling technique a particular kind of fractal can result from a disorderly and irreversible growth process. This technique seems to be attractive to implement, since it is conceptually simple and explains how vaiety of fractals may from. In this paper we present an algorithm or graphics approoach to implement Witten's technique which has not been done before.

The basic mathematical idea behind constructing the Mandelbort object by diffusion limited aggregation can be illustrated schematically in Figure 1. Typically one begins with smooth cluster onto which diffusion particles aggregate. Tiny bumps and holes form onto the surface owing to noise or random statistics of the incoming particles. This process can be captured

by simulation. Particles are modelled as colored pixels of the screen monitor and by executing an iterative feedback procedure to produce sequences of semi - random numbers x and y (of type REAL) which will aggregate towards forming the fractal shape. As we proceed with this iteration, a count variable I (or type INTEGER) keeps track of how many steps you have performed. Now suppose your screen has graphics resolution of A times B points. Also let k + 1 be the number of distinct colors which your micro can display. Number these colors from 0 through to k.



* The fuzzy lines shows the random path of the incoming particles.
* The contour lines showing the density level of particles.
* The arrow lines show the streamlines of the average flow of particles.

Figure 1 : Schematic Diagram for the Fractal Forming Process Using the idea of Diffusion Limited Aggregation.

The fractal forming algorithm then can be described by the following steps:

Procedure Diffusion - Limited - Aggregation:

begin

Set the viewing dimmensions (H versus V) in a complex plane;

for $P = \phi$ *to* H *do*

for $Q = \phi$ *to* V *do*

begin

generate a unique point R using P and Q using the formula

$R_{i+1}(X,Y) = R_i(X,Y) + C_i(P,Q)$ where $X_i = X_i^2 - Y_i = 2*X*Y$

if R > threashold then

print white color since the point lies in the complex plane

else print black color since the point is outside the complex plane

end

end

The forming process is shown on the screen step by step in which it will last for long time. This process can generates a fern - like fractal (see figure 2), it is considered to be generic as variety of fractals can be generated by altering some iterative steps. This process use no inputs and it depends on the iterative increase of both P and Q to genertate paints within the viewing dimmension.

Figure 2 : Afern - like fractal

(3) ITERATED FUNCTION SYSTEMS FOR FRACTAL MODELLING:

The essence of a fractal is to have detail at all scales. One way to achieve this characteristic is through self - similarity. An object is self - similar if small pieces of itself are identically chapped versions of the complete object, only on a smaller scale. One method to generate such fractals is called Iterative Function Systems.

The theory behind generting the IFS codes is called the IFS theory introduced by Barnsley [5]. This theory uses affine transformations, explained below, to express relations between parts of an image. Using only these relations, it defines and conveys intricate pictures. With IFS theory, we can describe a cloud as clearly as an architect can describe a house.

Affine transformations can be described as combinations of rotations, scalings, and translations of the coordinat exes in n -

dimensional space.

The general form for an affine transformation is:

$$W \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

$$x' = ax + by + e$$

$$y' = ex + dy + f$$

Where the coefficients a, b, c, d, e, and f are real numbers.

If we know in advance the translations, ratations, and scalings that combine to produce W, we can generate coefficient values as follows:

$$a = r \cos p, \; b = -s \sin q.$$

$$c = f \sin p, \; d = s \cos q.$$

Where r is the scaling factor on x, s is the scaling factor on y, p is the angle of rotation on x, q is the angle of rotation on y, e is the translation on x, and f is the translation on y.

Here's an example of an IFS of three transformations:

$$W1 \quad \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} .5 & .0 \\ .0 & .5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$W2 \quad \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} .5 & .0 \\ .0 & .5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$W3 \quad \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} .5 & .0 \\ .0 & .5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} .25 \\ .5 \end{bmatrix}$$

Each transformation must also have an associated probability, p1, determining its "importance" relative to the other transformations. In the present case we might have p1, p2, and p3. Notice that the probabilities must add up to 1. That is, $p1+p2+p3 = 1$. Of course, the above notation for an IFS is cumbersome. Table 1 expresses the same information in tabular form.

W	a	b	c	d	e	f	p
1	.5	0	0	.5	0	0	.33
2	.5	0	0	.5	1	0	.33
3	.5	0	0	.5	.5	.5	.34

Table 1 : IFS cosdes for a Sierpinski triangle.

Now let's see how to decode an arbitrary IFS code using an iteration method. Remember that in general an IFS can obtain any number, say m, of affine transformations, W1, W2, W3,... Win, each with an associated probability. The following algorithm summarize the method:-

Proceduse IFS:

begin

Intalize $X = \phi$, $Y = \phi$:

for $n = 1 + 0$ *large do*

input affine transformation parameters a, b, c, d, e, associated with their probability pi;

apply the transformation Wi to the poin (x, y) to obtain (x';y').

Set (x, y) equal to the new point $x = x-$, $y = y-$:

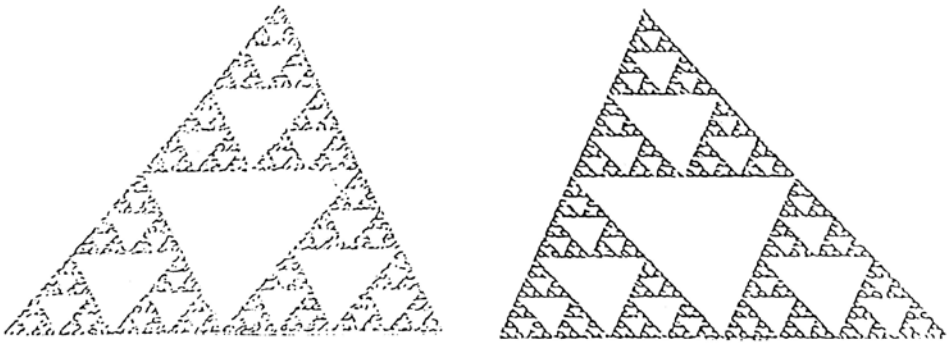
if $n >$ *step then plot (x, y)*

end

end

Applying this procedure to the transformation in Table 1 produces the figure shwn in figure 4 (a) fractal - known as the Sierpinski triangle. Increasing the number of iterations n adds points to the image, figure 4 (b) for 5000 iteration. Indeed this technique presents a new method for image compression if compared to the traditional methods of image digitizing as well as a technique for increasing / decreasing the image resolution by changing the number of iterations. Moreover, this technique performance is quite superior compared to the diffusion limited aggregation method from the point of view of computer plotting time (about 95% improvement compared to the Diffusion

method) of the fractal model point as well as being a parametric method for modelling fractals.



Sierpinski Triangte Level =3 Sierpinski Triangle Level=6

Figure 4 : Applying the iteration algorithm to the IFS code in Table 1.

(4) FRACTAL SCALING AND ZOOMING TECHNIQUES:

Fractal scaling differs from an ordinary scaling in that it is not expressed as a whole number but as a fraction. To determine the precies fractal dimension of an object one counts the average number N of fundamental units of repetition found within a sphere of a certain radius r centered somewhere on the object. According to the Eculidean geometry, the number of fundamental units equals a constant C multiplied by the radius raised to the value of the dimension D (N = C*r D). An example illustrating the scaling process is given in figure 5.

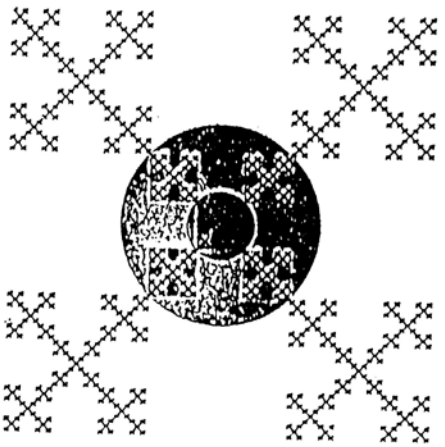


Figure 5 : An Example of Fractal Scaling.

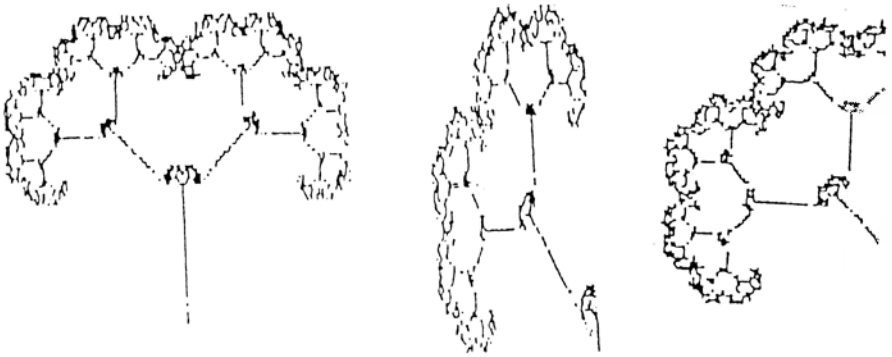
The Fractal Scale is 1.46, i. e. tripling the radius multiplies the number of units by 5.

However, Zooming primitives always helpful for understanding further details about the image, exactly as photography does. These primitives is quite simple using the diffusion limited aggregation method, in which any fractal is generated in two dimensional plane form -2.25 to 0.75 horizontally and -1.5 to 1.5 vertically. If you want to obtain zooming to a part of the fractal object in the region XMIN to XMAX and YMIN to YMAX then in step 0 you must take H = ((XMAX - XMIN) / (A- 1)) and V = (YMAX - YMIN) / (B - 1) and in step 1 the figure of -2.25 and -1.5 must be replaced by the values XMIN and YMIN that you hve chosen. An Example on some zooming effects on the Mandelbort objects is shown in Figure 6.

Step 1 Step 2 Step 3

Figure 6 : Diffusion Aggregation Fractal Zooming in Steps.

The zooming primitives are also controlled parametically using the iterated function technique. Simply by moving the XSCALE and YSCALE to any region that is needed to be analyzed further and then by changing XOFFSET and YOFFSET to any enlargment size wanted. Figure 7 (a, b, c,) presents two zooming steps performed on a tree - like fractal using the iterated function technique.



WSCALE = 400
YSCALE = 180
XOFFSET = 10
YOFFSET = 20
(A)

XSCALE = 350
YSCALE = 350
XOFFSET = 10
YOFFSET = 20
(B)

XSCALE = 350
YSCALE = 325
XOFFSET = -20
YOFFSET = -10
(C)

Figure 7 : Two Steps Zooming for Tree - Like Fractal.

(5) CONCLUSIONS:

We have implemented two particular techniques for fractal forming which help us in recognized and modeling natural objects. The diffusion limted aggregation technique suffers from being slow and the image must be simulated using a lengthy growth process. The Diffusion Limited Aggregation method is found to be complicated since fractals modeling depends on an indirect method of assigning points in complex numbers plane. That means one can not easily decide which parameters may reproduce a fractal image in this method. On the other hand the iterated function technique is quite fast resulting compact images as will as a method for presenting the image details parametically. This means instead of saving an image displayed on a screen 640 by 480 which requires 38,000 bytes in length, the IFS codes is only needed to store any fractal image, which in tern requires only quite few bytes only. This method is suitable for modeling purposes of fractals because of its parametric nature as well as it is a method for image compression [6]. Indeed, we believe that the technique of fractal graphics will open new frontiers in pattern recogition and image processing as well as in other field, especially with the availability of user defined fractals through the use of fractal languages and compilers such as the L-Systems [7].

(6) REFRENCES:

1 - L. M. Sander, 'Fractal Growth'
Scientific Averican, Vol 256, No. 1,pp 94 - 100, 1987.

2 - T. Wegner and M. Peterson, 'Fracted Creations', The Wait Group pub. co., 1991.

3 - B. Beniot and W. H. Mandelbrot, 'The Fractal Geometry of Nature', Freeman and Company, 1982.

4 - T. A. Witten and L. M. Sander, 'Diffusion limited Aggregation' Physical Review B, Vol 27, No. 9, pp 5686 - 5697, 1983.

5 - M. F. Bransely, 'Fractals EVERYWHERE', Academic press, 1988.

6 - D. Oliver 'Fractals and Data Compression',
P. C. Magazine, Vol. 1 Issuse 2, 1995.

7 - P. Prunsinkiewics and A. Lindenmager et. al. 'Beutiful pictures of plant simulations and L - Systems Theory', Spring - Verlag, 1990.