Genetic Algorithm based Fractal Image Compression

Mahesh G. Huddar
Lecturer, Dept. of CSE,Hirasugar Institute of Technology, Nidasoshi, India

Abstract:*Fractal Image Compression is a lossy compression technique that has been developed in the early 1990s. Fractal image compression explores the self-similarity property of a natural image and utilizes the partitioned iterated function system (PIFS) to encode it. The fractal image compression problem had three major requirements: speeding up the compression algorithm, improving image quality and increasing compression ratio. So far, several methods have been proposed in order to speed-up fractal image compression. The time is essentially spent on the search of the similar domain block. This paper aims to present a method that uses Genetic algorithms to speed up computation time in fractal image compression with acceptable image quality and high compression rate. These improvements are obtained by encoding all regions in the image with different size blocks.*

Keywords:*Fractal image compression; genetic algorithm.*

I. Introduction

Deterministic Fractals have the intrinsic property of having extremely high visual complexity while being very low in information content, as they can be described and generated by simple recursive deterministic algorithms. They are mathematical objects with a high degree of redundancy in the sense that they are recursively made of transformed copies of either themselves or parts of themselves.Fractal image compression (FIC) was introduced by Bernesly [1] and Jacquin [2]. Since then, many researches have improved the original approach in various ways [3]. The image compression problem puts forward three major requirements: speeding up the compression algorithm, improving image qualityafter compression/decompression or increasing compression ratio. The method based on the theory of partitioned Iterated Function Systems (PIFS) [3] has received a lot of attention in the last ten years. To encode an image according to the self-similarity property, each block must find the most similar domain block in a large domain pool.

In fractal image compression, the original image is partitioned into range blocks [8] and for each range block, a suitable domain block [2] D is searched, so it exists a transformation,

$$T: Dom(I) \rightarrow Ranges(I);$$
$$T \text{ must guaranty } \forall i,\ \exists j \,/ T(D_j) \approx R_i$$

A transformation is associated to each R_i, it codes the D_j coordinates and parameters of the transformation.

The associated parameters for each R_i are: the isometric flip Rotation $\pi/2,\ \pi, 3\pi/2$, the horizontal flip, the vertical flip, the transposed of *Dom(I)*, the rotation π of the transposedof *Dom(I)*, the luminance and contrast.

A. General Structure of FIC Algorithm

A general structure for most proposed fractal compression algorithms, for both coding and decoding images can be given by:

```
Step 1:  Encoding of an Image I
   •  Set t = some tolerance level
   •  Partitioning I into uncovered ranges $R_i$'s.
   •  For each uncovered range $R_i$do
        ▪  Search over all $D_i$'s in the pool domains:
        ▪  If ($\exists w_i$ such that$d(R_i,w_i(D_i))< t$)
             ◦  Report $w_i$and compress it using adaptivearithmeticCoding (or any other lossless compression
                scheme.)
           Else
        ▪  Split $R_i$ into sub-ranges and add them to the list of ranges to be covered.
     ◦  If the range can no longer be partitioned, return the minimum $d(R_i, wi(D_i))$.Remove $R_i$ from the uncovered list.

Step 2:  Decoding of a map $w = \cup w_i$
   •  Choose any image $I_0$, and then compute the image$w_n(I_0) = \cup w_{ni}(I_0)$. When n is big enough, $w_n(I0) \approx I_w \approx I$.
```

The major problem of this method is time consuming compared with others methods of image compression.In this paper, a new Genetic Algorithm forimage compression is proposed, that speed up this method when findinga LIFS [6] whose attractor is close to a given image.

II. Literature Survey

A fully automated fractal-based image compression technique of digital monochrome image was first proposed by Jacquin [2]. The encoding process consists of approximating the small image blocks, called range blocks, from the larger blocks, called domain blocks, of the image, through some operations. In the encoding process, separate transformations for each range block are obtained. The scheme also uses the theory of vector quantization [4] to classify the blocks. The set consisting of these transformations, when iterated upon any initial image, will produce a fixed point (attractor) that approximates the target image. This scheme can be viewed as partitioned iterative function system (PIFS). One such scheme, using PIFS, to store fewer number of bits (or to increase the compression ratio) was proposed by Fisher et al. [7].

III. Fractal Image Compression Using Genetic Algorithm

There are many algorithms of optimization used for different domains. I have chosen Genetic Algorithm [7] [8] [9]to accelerate our fractal image compression algorithm.For each range domain R_i, the set of all possible domain blocks is genetically browsed until we find an appropriate solution. The genetic algorithm search space parameters are the domain block coordinates and the isometric flip.

A. Chromosome Attributes

A chromosome is constituted by 5 genes, from which only 3 genes are submitted to genetic modification, the two others are computed by the RMS equation.

- X_{dom},Y_{dom}, flip: which are optimised by genetic search?
- Contrast O, and scaling S: which are computed directly by RMS equation.

X_{dom}	Y_{dom}	Flip	O_{opt}	S_{opt}

Figure 1 Chromosome Representation

B. Genetic Operators

The crossover and mutation operators ensure the production of offspring. These genetic operators must be defined according to the chromosome specification. With these basic components, a Genetic Algorithm works as follows: The first procedure is to generate the first population represented with string codification (Chromosome) that represents possible solution to the problem. Each individual is evaluated, and according to its fitness, an associated probability to be selected for reproduction is assigned.

(1) Crossover Operator

The crossover operator combines two individuals in the current population, to produce two offspring individuals included in the new generation. The main role of this operator is to create good new solutions based on the characteristics of their parents. To perform this operation, individuals from current population are chosen randomly and proportionally to their fitness value. This operator with a probability value fixed as a parameter for the algorithm. Experimental results have shown that a value of 0.7 is good to ensure quick convergence of the algorithm. The result coordinates for the offspring individuals are obtained by a linear combination of the parents coordinates. A random number 'a' is generated in the interval [0, 1], then the new coordinates are calculated according to the following formula:

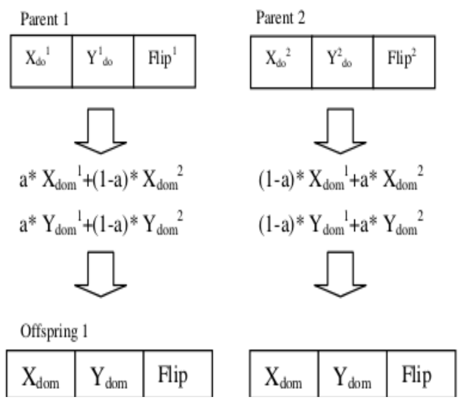
For the first offspring

$$X_{dom}=a^* X_{dom1}+(1-a)^* X_{dom2}$$
$$Y_{dom}=a^* Y_{dom1} +(1-a)^* Y_{dom2}$$

For the second offspring

$$X_{dom}=(1-a)^* X_{dom1}+(1-a)^* X_{dom2}$$
$$Y_{dom}=(1-a)^* Y_{dom1}+(1-a)^* Y_{dom2}$$

This crossover pattern is very efficient. It allows to explore all the image area if the two parents are in separated regions, and to explore the nearest neighbourhood if they are in the same region. Figure 2 illustrates the described crossover pattern.


Parent 1
X_{do}^1 | Y^1_{do} | $Flip^1$
Parent 2
X_{do}^2 | Y^2_{do} | $Flip^2$
$a^* X_{dom}^1+(1-a)^* X_{dom}^2$
$a^* Y_{dom}^1+(1-a)^* Y_{dom}^2$
$(1-a)^* X_{dom}^1+a^* X_{dom}^2$
$(1-a)^* Y_{dom}^1+a^* Y_{dom}^2$
Offspring 1
X_{dom} | Y_{dom} | Flip
X_{dom} | Y_{dom} | Flip
Figure 2Crossover Operator Pattern

(2) Mutation operator

Mutation operator is used in all implementations of genetic algorithms to introduce diversity in each population. Mutation is applied to individuals by changing pieces of their representations. These individuals are randomly selected according to their fitness value. At each position in the individual, we decide randomly, using a small mutation probability, whether the gene at this position is to be changed. This genetic operator allows the algorithm to explore new areas of the search space, and find new possible optimal solution. Here mutation operator is applied to some selected IFSs from the current population, one of the 3 genes X_{dom}, Y_{dom} and flip is selected randomly, and changed with a random generated value. Figure 3 shows the general pattern of the mutation operator in our algorithm.

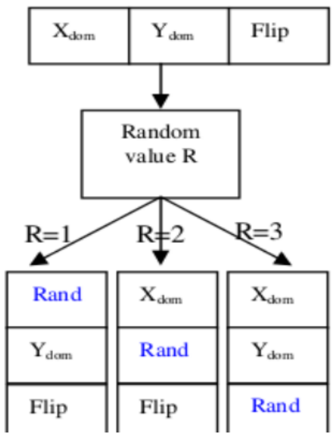


Fig. 3 Mutation Operator Pattern

C. Selection Process

To avoid the premature convergence effect, linear scaling is applied to each individual fitness then, the Roulette wheel method is used as a selection process.

D. Termination Criteria

Any genetic algorithm must find the optimal solution for a given problem in a finite number of steps. Two criteria can cause the termination of the algorithm when applied to a given range block:

- An acceptable value of fitness for the best in individual in the population is reached.
- A maximum predefined count of generations is reached.

This maximum count is a predefined parameter of the algorithm; it was determined experimentally and fixed to 20.

E. Parameters of the Algorithm

The set of parameters of genetic control is given by:

(1) **Population size**: specify the number of individuals in each generated population (constant during all steps);
(2) **Crossover rate**: specify the probability used to select individuals submitted to crossover operator;
(3) **Mutation rate**: specify the probability used to select individuals submitted to mutation operator;
(4) **Maximum generations**: specify the maximum number of generations to evaluate before assuming that a particular run has converged prematurely, and should be aborted.

F. The Fitness Function

In the case of IFS, the measure of quality for a given transformation is given by the RMS error between the coded range block, and the domain block determined by the transformation coordinates X_{dom} and Y_{dom}, and transformed with corresponding luminance and contrast values. This error is calculated using the root mean square equation. The fitness function is defined by the value of this error, which is inversely proportional to the efficiency of the corresponding individual.

G.The Genetic Fractal Image Compression Algorithm

Based on all these elements, the geneticcompression algorithm operate on an input image to the following general steps:

```
(Input I: NxNgrey scale image [Image would be square]
(Output W: Coded IFS);
(Region Size) = 16;
(Fixed Error) = X * (Region Size)/4;
Decompose the input image into (Region Size) blocks;
While Exist (Regions not coded)
    -Scale the Domain Blocks;
    -Generate a random population of chromosomes;
    While Exist (Regions not coded) and (Last generation not reached)
```

```
            -Compute fitness for all regions;
            -When optimal domain block found write obtained transformation parameters to the output W;
            -Generate new population {Apply Crossover and Mutation operators};
        Wend
        (RegionSize) = (RegionSize)/2;
        (FixedError) = X * (RegionSize)/4;
        If Regions size > 4
            Decompose the rest region not coded into (Range Size) blocks;
        Else
            (FixedError) = (Fixed Error) +X;
            Code all rest Regions;
        IEnd
    Wend
```

IV. Experimental Results

A. Std. Algorithm with Quad tree Partitioning

Image	RMS Limit	Execution Time	Quality (dB)	Compression Ratio
Lena	5.0	1:11:10	32.01	11.48 :1
Boat	5.0	1:27: 04	29.56	8.88:1
Peppers	5.0	1:05:17	33.07	24.56:1
Barb	5.0	1:22:15	33.07	10.85:1

Table 1. Result for different images with Quadtree blocks decomposition

B. Standard algorithm improved with Classification (Y. Fisher approach)

Image	RMS Limit	Execution Time	Quality (dB)	Compression Ratio
Lena	5.0	0:02:55	30.05	9.73 :1
Boat	5.0	0:03:22	25.86	7.9 :1
Peppers	5.0	0:02:45	29.36	10.16 :1
Barb	5.0	0:04:12	21.07	8.62:1

Table 2. Results for different images with Quad tree blocks decomposition using Fisher's classification

C. Genetic Algorithm

The genetic compression algorithm was used with Quad tree partitioning. Different parameters were used for each test, and the obtained results are given in both table forms and graphical forms. Examples of reconstructed images are also given to illustrate reconstruction quality.

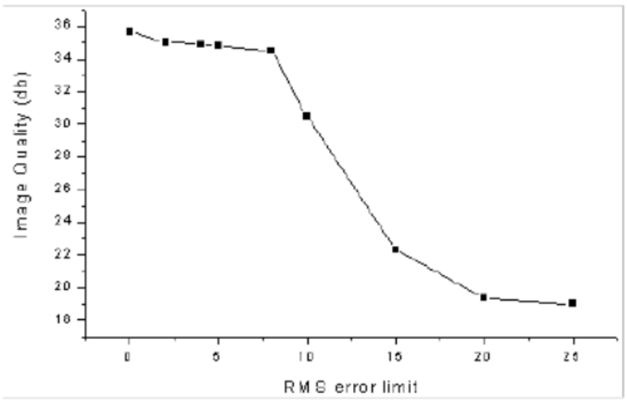
The following table shows different performances with different value of RMS error limit using fixed value for other parameters: population size=100, mutation rate=0.1, crossover rate=0.7 and maximum generations count =20.

(1)Effect of parameters controlling fractal compression

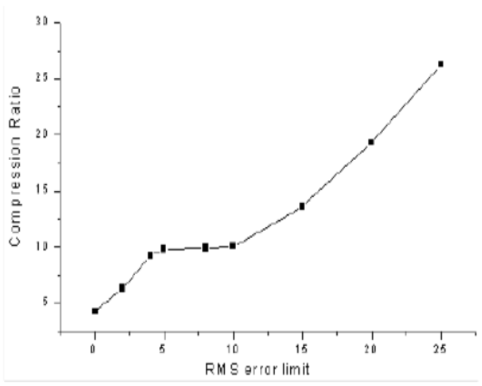
We can see from the Graphs 1 and 2, that image quality is inversely proportionate to RMS error limit. And compression rate is proportionate to that value. The compromise value of this parameter is 5.0 and it gives very acceptable performances.

RMS Limit	Execution Time	Quality (dB)	Ratio	Ranges count
0.0	2 m 44 s	35.66	4.29:1	4096
2.0	1 m 56 s	35.03	6.35:1	2770
4.0	49 s	34.89	9.28:1	2023
5.0	43 s	34.80	9.82:1	1792
8.0	36 s	34.50	9.95:1	1768
10.0	33 s	30.50	10.05:1	1750
15.0	21 s	22.33	13.66:1	1288
20.0	14 s	19.36	19.34:1	910
25.0	13 s	19.01	26.25:1	670

Table 3. Different compression results of Lena image while applying different values ofRMS error limit



Graph 1 Lena image quality variation according to RMS limit values

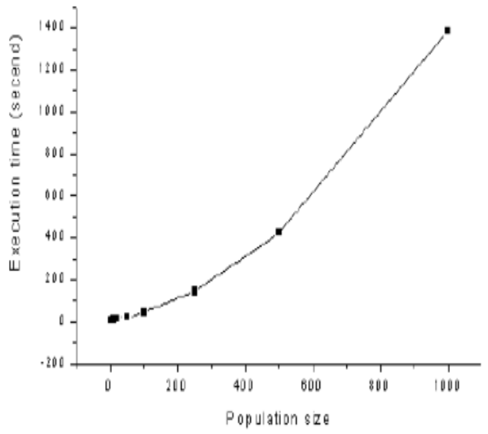


Graph 2 Lena image compression rate variation according to RMS limit values

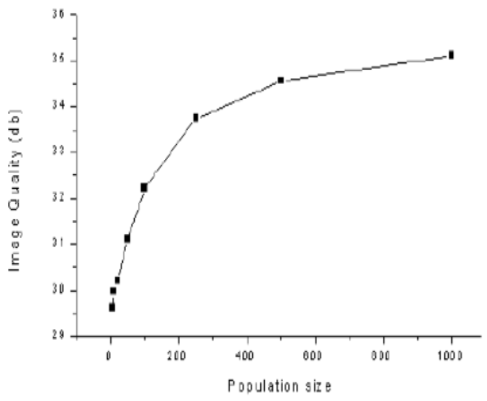
(2) Effect of parameters controlling genetic evolution

Population Size	Execution Time	Quality (dB)	Ratio	Ranges count
5	9 s	29.62	8.30:1	2119
10	11 s	29.98	8.35:1	2107
20	14 s	30.21	8.72:1	2017
50	23 s	32.11	9.36:1	1879
100	44 s	32.23	9.83:1	1789
250	2 m 24 s	33.74	10.35:1	1699
500	7 m 4 s	34.56	10.83:1	1624
1000	23m 4 s	35.12	10.97:1	1603

Table 4 Different compression results of Lena image while applying different values of population size



Graph3 Lena image compression time variation according to population size values



Graph 4 Lena image quality variation according to population size values

V. Conclusion

It is clear that the best image quality is always obtained using the standard schema, but its computation time makes it unpractical. So we must accept less quality in favour of quick compression. The main goal was to accelerate standard compression schema, without greatly decreasing both image quality and compression ratio. Furthermore this work demonstrates the genetic algorithms ability to solve complex problems.

References

[1] M. Barnsley and L. Hurt,"Fractal Image Compression", Peters, Wellesley, 1993
[2] A. E. Jaquin"Image coding based on a fractal theory of iterated contractive image transformations", IEEE Trans. on image Processing. 1, PP, 18-30, 1992.
[3] B.E. Wohlberg and G. de Jager,"A review of the fractal image coding literature", IEEE Trans. on image Processing. 8(12), PP, 1716-1729, 1999.
[4] VeenadeviS.V. And A.G.Ananth"Fractal Image Compression with Quadtrees", Signal & Image Processing: An International Journal (SIPIJ) Vol.3, No.2, April 2012.
[5] M.F. Barnsley and S. Demko. Iterated function systems and the global construction of fractals. In Proceedings of the Royal Society of London A399, pages 243 275, 1985.
[6] Arnaud E. Jacquin, "Fractal Image Coding: A Review", MEMBER, IEEE
[7] Y. Fisher, E. W. Jacbos, and R. D. Boss, "Fractal image compression using iterated transforms," in Image and Text Compression, J. A. Storer, Ed. Boston, MA: Kluwer, 1992, pp. 36–61.
[8] Y. Chakrapani and K. SoundaraRajan, "GENETIC Algorithm Applied To Fractal Image Compression", ARPN Journal of Engineering and Applied Sciences, VOL. 4, NO. 1, FEBRUARY 2009
[9] S. K. Pal, D. Bhandari, and M. K. Kundu, "Genetic algorithms for optimal image enhancement," Pattern Recognit. Lett, vol. 15, pp. 261–271, 1994.
[10] S. Forrest, Ed., Proc. 5th Int. Conf. Genetic Algorithms, San Mateo, CA, July 1993.