International Journal of Future Generation Communication and Networking
Vol. 10, No. 3 (2017), pp. 41-48

Flower Pollination Technique for Fractal Image Compression

KanimozhiRajasekaran, P.D. Sathya
rs.kaniraj@gmail.com, pd.sathya@yahoo.in
Department of Electronics and Communication Engineering
Annamalai University, India-608002

ABSTRACT

Metaheuristic Optimization techniques are used to overcome the problems inherent in iterative simulations. These methods have certain remarkable characteristics and abilities, which is that it requires only less computation time and memory to find better solutions i.e., near optimal solutions without using any composite derivatives. Over the years, varieties of meta-heuristic optimization algorithms have been introduced and used to compress the images with minimal loss of information. The objective of image compression is to reduce the redundancy of the image and to store or transmit data in an efficient form. One such type of image compression is Fractal Image Compression (FIC). These FIC techniques commonly use the optimization techniques to find the optimal best solution. The aim of the FIC is to divide the image into pieces or sections and then finds self-similar ones. It produces high compression ratio, fast decompression in short amount of time. But the major disadvantage of Fractal Image Compression is that, it requires high computational cost and retrieves image with poor qualities. In this paper, Flower Pollination Based Optimization approach is used for fractal image compression. This optimization technique effectively reduces the encoding time while retaining the quality of the image. Here, Flower pollination algorithm (FPA) is compared with genetic algorithm (GA)and their performances are analyzed in terms of compression ratio, encoding time and PSNR (Peak Signal-to Noise Ratio) value.

1. Introduction

Compression and decompression technology of digital image has become an important aspect in the storing and transferring of digital image in information society. Most of the methods in use can be classified under the head of lossy compression. This implies that the reconstructed image is always an approximation of the original image. Fractal image coding introduced by Barnsley and Jacquin [1-3] is the outcome of the study of the iterated function system developed in the last decade. Because of its high compression ratio and simple decompression method, many researchers have done a lot of research on it. But the main drawback of their work can be related to large computational time for image compression. The first practical fractal image compression scheme was introduced in 1992 by Jacquin. One of the main disadvantages of using exhaustive search strategy is the low encoding speed. Therefore, improving the encoding speed is important for FIC.

Fractal compression is a lossy compression method for digital images, based on fractals. The method is best suited for textures and natural images, relying on the fact that parts of an image often resemble other parts of the same image. Fractal image compression is attractive because of high compression ratio, fast decompression and multi-resolution properties. The two major advantages of changing images to fractal data are, 1) the memory size required to store fractal codes is extremely smaller than the memory required to store the original bitmap information, 2) the image can be scaled up or down a size (zooming) easily without disrupting the image details as the data becomes mathematical on conversion of image to fractals [4]. In FIC, encoding process is more time consuming than decoding as it is difficult to find an appropriate approximation of an image by a set of affine contractive maps. Of late, one of the most active research areas in FIC is reducing the search complexity of matching between range block and domain block [5]. Numerous techniques have been proposed in order to fasten fractal image coding. Lately, many researchers have looked into a fast-encoding algorithm to speed-up the fractal encoding process [6-8]. To overcome this drawback, many optimization algorithms such as Genetic Algorithm, Flower Pollination Algorithm were introduced and used.

ISSN: 2233-7857 IJFGCN

In the present work, Flower pollination algorithm is compared with Genetic algorithm. This paper will describe how the performance of flower pollination algorithm is better compared to Genetic algorithm.

GAs are member of a wider population of algorithm, Evolutionary Algorithm (EA). The idea of evolutionary computing was introduced in the year 1960 by I. Rechenberg in his work "evolution strategies" ("Evolution strategies" in original). GA was invented by John Holland. Genetic algorithms (GA's) are a stochastic global search method that mimics the process of natural evolution. The genetic algorithm starts with no knowledge of the correct solution and depends entirely on responses from its environment and evolution operators to arrive at the best solution. Instead of searching one point at a time, GA's use multiple search points. GA's attempt to find near-optimal solutions without going through an exhaustive search mechanism. Thus, GA's can claim significant advantage of large reduction in search space and time.A few investigations have been carried out in application of GA to fractal image compression. GA is an efficient means of investigating large combinational problems. But it also suffers from major disadvantages such as, itis computationally expensive, sensitive to initial parameters and not guaranteed to find an optimal solution. To overcome these disadvantages, this paper uses Flower Pollination algorithm (FPA).

The latest nature inspired algorithm is Flower Pollination Algorithm which was proposed by Xin-She Yang in 2012 [9]. This is based on the pollination of flowers. Flower Pollination Based Optimization is nature inspired algorithm which decreases the search complexity of matching between range block and domain block. Also, the optimization technique has effectively reduced the encoding time while retaining the quality of the image. Flower pollination is a process associated with transferring flowers pollens. The main actors of performing such transfer are birds, bats, insects, and other animals. There exist some flowers and insects that have made what we can call a flower-pollinator partnership. These flowers can only attract the birds that are involved in that partnership, and these insects are considered the main pollinators for these flowers. The pollination is a result of fertilization and it is must in agriculture to produce fruits and seeds [10]. Flower pollination process can occur at both local and global levels. But in reality, neighborhood flowers have higher chances of getting pollinated by pollen from local flowers than those which are far away. Flower pollination process is achieved through cross-pollination or self-pollination.

2. Fractal Image Compression

Iteration Function System (IFS) is the basic idea of fractal image compression in which the governing theorems are the Collage Theorem and the Contractive Mapping Fixed-Point Theorem [12]. The encoding unit of FIC for given grey level image of size N x N is (N/L)2 of non-overlapping range blocks of size L x L which forms the range pool *R*. For each range block *v* in *R,* one search in the (N - 2L + 1)2 overlapping domain blocks of size 2L x 2L which forms the domain pool *D* to find the best match. The parameters describing this fractal affine transformation of domain block into range block form the fractal compression code of *v*.

The parameters of fractal affine transformation isΦ of domain block into range block having domain block coordinates (t_x, t_y), Dihedral transformation-*d,* contrast scaling-*p,* brightness offset-*q.*

$$\Phi\begin{bmatrix} x \\ y \\ u(x,y) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ 0 & 0 & p \end{bmatrix}\begin{bmatrix} x \\ y \\ u(x,y) \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ q \end{bmatrix} \quad (1)$$

Where the 2 x 2 sub-matrix $\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ is one of the Dihedral transformations in (2)

$$T_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ T_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},\ T_2 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix},\ T_3 = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix},$$

$$T_4 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ T_5 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix},\ T_6 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix},\ T_7 = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}. \quad (2)$$

The above parameters are found using the following procedure:

1. The domain block is first down-sampled to L x L and denoted by *u.*

2. The down-sampled block is transformed subject to the eight transformations T_k: k = 0,. . . ,7 in the Dihedral on the pixel positions and are denoted by u_k, *k* = 0,1, . . . ,7, where u_0 = *u*. The transformations *T*1 and *T*2 correspond to the flips of *u*along the horizontal and vertical lines, respectively. *T*3 is the flip along both the horizontal and vertical lines. *T*4, *T*5, *T*6,and *T*7 are the transformations of *T*0, *T*1, *T*2, and *T*3 performed by an additional flip along the main diagonal line, respectively.

3. For each domain block, there are eight separate MSE computations required to find the index d such that, the eight transformed blocks are denoted by u_k,k = 0,1, . . . ,7, where u_0 = u. The transformations T1 and T2 correspond to the flips of u along the horizontal and vertical lines, respectively. T3 is the flip along both the horizontal and vertical lines. T4, T5, T6, and T7 are the transformations of T0, T1, T2, and T3 performed by an additional flip along the main diagonal line, respectively. In fractal coding, it is also allowed a contrast scaling p and a brightness offset q on the transformed blocks. Thus, the fractal affine transformation U of u(x,y) in D can be expressed as

$$d = \arg\min\{MSE((p_k u_k + q_k), v): k = 0,1,\ldots,7\} \quad (3)$$

where $MSE(u,v) = \frac{1}{L^2}\sum_{i,j=0}^{L-1}(u(i,j=0) - v(i,j))^2$ (4)

Here, $p_k$and $q_k$can be computed directly as

$$p_k = \frac{\left[L^2\{u_k, v\} - \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} u_k(i,j)\sum_{i=0}^{L-1}\sum_{i=0}^{L-1} v(i,j)\right]}{\left[L^2(u_k, u_k) - \left(\sum_{i=0}^{L-1}\sum_{j=0}^{L-1} u_k(i,j)\right)^2\right]} \quad (5)$$

$$q_k = \frac{1}{L^2}\left[\sum_{i=0}^{L-1}\sum_{i=0}^{L-1} v(i,j) - p_k\sum_{i=0}^{L-1}\sum_{j=0}^{L-1} u_k(i,j)\right] \quad (6)$$

4. As *u* runs over all of the domain blocks in *D* to find the best match, the terms t_x and t_y can be obtained together with *d* and the specific *p* and *q* corresponding this *d*, the affine transformation (1) is found for the given range block *v*.

To decode, the compression codes to obtain a new image, and proceeds recursively by chooses any image as the initial one and makes up the (N/L) 2 affine transformations. According to Partitioned Iteration Function Theorem (PIFS), the sequence of images will converge. The final image is the retrieved image of fractal coding.

3. Genetic Algorithm

The searching process used by genetic algorithm is similar to that in nature, where successive generations of organisms are reproduced and raised until they themselves can reproduce. To use a genetic algorithm, initialize the genetic algorithm with a set of solutions represented by chromosomes called a population. Each solution can be represented as either real valued numbers or a binary string of ones and zeros. These solutions are known as individuals. In these algorithms the fittest among a group of individuals survive and are used to form new generations of individuals with improved fitness vales. The fitness of an individual is a measure of how well the individual has performed in the problem domain.

To illustrate the working process of genetic algorithm, the steps to realize a basic GA are listed:

Step 1: Represent the problem variable domain as a chromosome of fixed length; choose the size of the chromosome population N, the crossover probability Pc and the mutation probability Pm.

Step 2: Define a fitness function to measure the performance of an individual chromosome in the problem domain. The fitness function establishes the basis for selecting chromosomes that will be mated during reproduction.

Step 3: Randomly generate an initial population of size N: $x_1, x_2, ..., x_N$

Step 4: Calculate the fitness of each individual chromosome: $f(x_1), f(x_2), ..., f(x_N)$

Step 5: Select a pair of chromosomes for mating from the current population. Parent chromosomes are selected with a probability related to their fitness. High fit chromosomes have a higher probability of being selected for mating than less fit chromosomes.

Step 6: Create a pair of offspring chromosomes by applying the genetic operators.

Step 7: Place the created offspring chromosomes in the new population.

Step 8: Repeat Step 5 until the size of the new population equals that of initial population, N.

Step 9: Replace the initial (parent) chromosome population with the new (offspring)population.

Step 10: Go to Step 4, and repeat the process until the termination criterion is satisfied.

A GA is an iterative process. Each iteration is called a generation. A typical number of generations for a simple GA can range from 50 to over 500. A common practice is to terminate a GA after a specified number of generations and then examine the best chromosomes in the population. If no satisfactory solution is found, then the GA is restarted.

4. Flower Pollination Algorithm

Pollination is a process of transfer of pollen from the male parts of a flower called another to the female part called stigma of a flower. The reproduction in plants happens by union of the gametes. The pollen grains produced by male gametes and ovules borne by female gametes are produced by different parts and it is essential that the pollen has to be transferred to the stigma for the union. This process of transfer and deposition of pollen grains from anther to the stigma of flower is pollination. The process of pollination is mostly facilitated by an agent.

In cross-pollination, pollens are transferred from a different plant. The biotic and cross-pollinations occur at long distances, so they are performed by insects that can fly for long distances such as bees, birds, and bats. The previously mentioned flies are considered as global pollinators. Birds and bees usually follow their behavior in the Levy flight, from this phenomenon, we can consider their moves as discrete jumps that obey the Levy distribution. The second type of pollination, by which the fertilization is achieved, is the self-pollination. In self-pollination, pollens from the same flower or the same type of the flower are responsible for the fertilization process. Self-pollination usually needs no pollinators.

The flower pollination algorithm, inspired by the flow pollination process of flowering plants. The FPA has been extended to multi-objective optimization. For simplicity, the following four rules are used.

- Biotic cross-pollination can be considered as a process of global pollination, and pollen carrying

pollinators move in a way that obeys Lévy flights (Rule 1).

- For local pollination, abiotic pollination and self-pollination are used (Rule2).
- Pollinators such as insects can develop flower constancy, which is equivalent to a reproduction probability that is proportional to the similarity of two flowers involved (Rule 3).
- The interaction or switching of local pollination and global pollination can be controlled by a switch probability *p* in [0, 1], slightly biased towards local pollination (Rule 4).

To formulate the updating formulas, these rules have to be changed into correct updating equations. The main steps of FPA, or simply the flower algorithm is illustrated below:

min or max objective *f(x), x = (x1, x2 ,. . . , xd)*

Initialize *n* flowers or pollen gametes population with random solutions Identify the best solution (*g**) in the initial population
Express a switch probability *p* in [0, 1] While (*t* < Max Generation)
for *i* = 1 : n (all *n* flowers in the population) ifrand <*p*,
Draw a (*d*-dimensional) step vector *L from a* Levy distribution Global pollination via $X_i^{t+1} = X_i^t + \gamma L\,(g^* - X_i^t)$,
else
Draw $\in$ from a uniform distribution in [0,1]
Do local pollination via
$X_i^{t+1} = X_i^t + \in (X_j^t - X_k^t)$,
end if
Evaluate new solutions
If new solutions are better, update them in population end for
Find current best solution end while
Output the best solution obtained

In principle, flower pollination process can happen at both local and global levels. But in reality, flowers in the neighborhood have higher chances of getting pollinated by pollen from local flowers than those which are far away.

To simulate this feature, a proximity probability p (Rule 4) can be commendably used to switch between intensive local pollination to common global pollination.

5. Results and Discussions

In the present work, the Flower pollination algorithm is compared with Genetic Algorithm. The FPA has advantages such as simplicity and flexibility. In terms of number of parameters, the FPA has only one key parameter *p* together with a scaling factor *γ*, which makes the algorithm easier to implement.

Figure 1(a). Original Lena image **Figure 1(b). Decoded Lena GA image**

Figure 1(c) Decoded Lena FPA image

The figures of 1(a),1(b) and 1(c) show the original image, the decoded GA image and the decoded FPA image. These figures shows that that the decoded FPA image has better quality than the decoded GA image. The decoded FPA image gives better quality with the increase in number of iterations. The numeric results containing encoding time and PSNR value of decoded images are given in Table 1.

Table 1. Comparison results of GA and FPA for Lena image

COMPRESSION METHOD	GA	FPA
Population size	20	20
Iteration	100	100
PSNR	21.75424	25.8004
Compression time (s)	56.1248	43.56
Decompression time (s)	51.187	29.58
Compression ratio	4.8	8.25

From simulation results, we see that the FPA gives a very good PSNR values, compression time and compression ratio for Lena image. The visual quality of the decoded FPA Lena image is good, compared to the decoded GA Lena image.

6. Conclusion

Metaheuristic approaches have been very effective and popular in recent years. In this paper, Flower Pollination Based Optimization approach is used for fractal image compression. This image compression algorithm is very efficient in terms of compression ratio and compression time and also, it retains the quality of image in terms of better PSNR value. The Flower Pollination algorithm is simple, flexible and more rapidly better to solve optimization problems. FPA can be used for solving both single objective and multi-objective optimization problems. Simulation results and tabulation have shown that the Flower Pollination algorithm is very efficient compared to genetic algorithm, i.e., the increased PSNR value indicates that FPA is better. The FPA reduces time, improves the results and the performance is better compared to other optimization techniques. FPA looks very promising and is still in budding stage and can be applied for medical image Analysis and in other areas of researches.

References

[1] M.F. Barnsley. 1988. Fractals Everywhere. New York: Academic.
[2] A.E. Jacquin. 1992. Image coding Based on a Fractal theory of Iterated contractive Image Transformations. IEEE Transactions on Image Processing. Vol. 1. pp. 18-30.
[3] A.E. Jacquin. 1993. Fractal Image coding: A Review Proc. IEEE. Vol. 81. pp. 1451-1465.
[4] Donald Walter, "Fractal and Wavelet Image Compression of Astronomical Images", URIA 2003.
[5] Lifeng Xi and Liangbin Zhang, "A Study of Fractal Image Compression Based on an Improved GeneticAlgorithm", International Journal of Nonlinear Science, Vol.3, No.2, pp. 116-124, March 2007
[6] Murray H. Loew, Dunling Li and Raymond L. Pickholtz, "Adaptive PIFS model in Fractal ImageCompression", in proc. of SPIE, Vol. 2707, No. 284, 1996.
[7] Sumathi Poobal, and G. Ravindran, "Arriving at an Optimum Value of Tolerance Factor for Compressing Medical Images", International Journal of Biological, Biomedical and Medical Sciences, Vol. 1, No. 4, pp. 250-254, 2006.
[8] Tomas and Kovacs, "A fast classification-based method for fractal image encoding", Image and vision Computing, Vol. 26, No. 8, pp: 1129- 1136, August 2008.

[9] Xin-She Yang, Mehmet Karamanoglu, XingshiHe,"Multi-objective Flower Algorithm for Optimization", ICCS 2013, Elsevier.
[10] Flower Pollination by biology.tutorvista.com/animalkingdom, Pearson,2005
[11] Y. Fisher, Fractal Image Compression: Theory and Application, Springer-Verlag, New York, 1994.
[12] Mohamed, Faraoun Kamel., Aoued, Boukeli F. 2005, "Optimization of Fractal Image Compression Based on Genetic Algorithms", 3rd International Conference: Sciences of Electronic, Technologies of Information and Telecommunications, Tunisia.
[13] S.K. Mitra, C.A. Murthy, M.K. Kundu, Technique for fractal image compression using genetic algorithm, IEEE Transactions on Image Processing 7 (1998) 586–593.
[14] Xin-She Yang, Flower Pollination Algorithm for Global Optimization, arXiv:1312.5673v1 [math.OC] 19 Dec 2013.
[15] Gaganpreet Kaur Dheerendra Singh, Manjinder Kaur, "Robust and Efficient 'RGB' based Fractal ImageCompression: Flower Pollination based Optimization", International Journal of Computer Applications (0975 –8887), Volume 78 – No.10, 2013.
[16] O.Abdel-Raouf, M. Abdel-Baset, I. El-henawy, "A New Hybrid Flower Pollination Algorithm for Solving Constrained Global Optimization Problems", International Journal of Applied Operational Research Vol. 4, No.2, pp. 1-13, Spring 2014.

