

Crowding Optimization Method to Improve Fractal Image Compressions Based Iterated Function Systems

Shaimaa S. Al-Bundi

Department of Mathematics-College of
Education for pure Sciences- Ibn Al-
Haitham-Baghdad University
Bagdad, Iraq

Nadia M. G. Al-Saidi

Applied Sciences Department-
University of Technology
Baghdad, Iraq

Neseif J. Al-Jawari

Department of Mathematics-College of
Sciences-Al-Mustansiriah University
Baghdad, Iraq

Abstract—Fractals are geometric patterns generated by Iterated Function System theory. A popular technique known as fractal image compression is based on this theory, which assumes that redundancy in an image can be exploited by block-wise self-similarity and that the original image can be approximated by a finite iteration of fractal codes. This technique offers high compression ratio among other image compression techniques. However, it presents several drawbacks, such as the inverse proportionality between image quality and computational cost. Numerous approaches have been proposed to find a compromise between quality and cost. As an efficient optimization approach, genetic algorithm is used for this purpose. In this paper, a crowding method, an improved genetic algorithm, is used to optimize the search space in the target image by good approximation to the global optimum in a single run. The experimental results for the proposed method show good efficiency by decreasing the encoding time while retaining a high quality image compared with the classical method of fractal image compression.

Keywords—Fractal; Iterated Function System (IFS); Genetic algorithm (GA); Crowding method; Fractal Image Compression (FIC)

I. INTRODUCTION

Fractal image compression (FIC) is produced from Barnsley's research IFS system [1] and the fractal image block coding suggested by Jacquin [2]. In 1988, Barnsley [3] used FIC based on the theoretical IFS system to represent computer graphics and compress the aerial image. Using this approach, Barnsley obtained a compression ratio of 1000:1, but the approach requires manual interference. Thereafter, Jacquin suggested a new FIC method that depends on image block and can behave automatically without manual interference. This method has become a perfect representation for this research direction, in which FIC theory is realized. At present, FIC has obtained extensive interest from the research community, because of its novel concept, high compression ratio, independent resolution, and fast image decoding. This technique is based on the fractal inverse problem and aims to find an IFS, in which the attractor is close to a query image.

The emerging technique for image compression that based on fractal theory is fully different form traditional image compression techniques. It is focused on two main problems:

the first one is how to find the IFS mappings and the second is finding of an efficient algorithm to find those mappings, such that, they can approximate the original image. Toward solving these problems, Jacquin [2] proposed an efficient technique by partitioning of a given image M into non-overlapping range blocks and an overlapping domain blocks, the IFS parameters is achieved by finding the best corresponding domain block for each range block. Therefore, as a result of this encoding process, we obtain a different transformation for each range block. If we composed all the transformations of all range blocks and iterated starting with the initial image, the attractor (fractal) that approximate the original image is produced, it is also called the fixed point of the transformations. This type of representation is called partitioned IFS [4] or local IFS [5].

Many researchers have emphasized on overlapping of an efficient and reliable image compression technique based on fractal. It is firstly presented by Barnsley and Sloan [6] in 1988, when they introduced of finding an *IFS*, whose attractor approximate the given image and the *IFS* is sent instead of sending the image itself over the channel. In 1992, A. Jacquin [7] Barnsley's student improved *IFS* theory and introduced the concept of local *IFS* through presenting the concept of fractal image coding. In 1994, Y. Fisher [4] made many improvements on Barnsley's algorithm. He combined his idea in a very famous book in this field. Since Jacquine's publication of the original fractal coding scheme, several papers try to popularize his work both in practical and theoretical [8], among others, however none of these attempts in general have been proven to be efficient. Therefore, many efforts are highlighted towards employing of evaluative algorithms. Numerus optimization models have been proposed to represent a normal evolution mechanism [9]. Genetic algorithms [10,11] is one of these models. In these algorithms, the population represents as an IFS models and it is responsible of making adjustments toward the optimum through a random process that used for selection of genetic operators called crossover and mutation.

GAs that are used to address an optimization problem are required to solve multimodal and multidimensional problems, through which a large search space with different optima can be obtained. These problems do not have deterministic algorithms to obtain the global optimum; if they do exist, however, the algorithm is an inclusive search along the

solution space that, in turn, leads to exponential time and machine resources using algorithms of this kinds in solving the problem described above. Therefore, the algorithms used to solve various complex problems can show their respective capacities. The GAs work with population of individuals that are iteratively adjusted towards the optimum by means of a random operation of selection restructure and mutation [11].

Meanwhile, crowding is a technique that is applied in GAs to maintain variety in the population and prohibit early convergence to local optima. This technique involves the combination of both the offspring and the identical individual from the present population in this process, which is called coupling phase; determining which of the two will remain in the population is a process called alternation phase [12]. The current work depends on the alternation phase of crowding, which is applied by using one of the following three approaches: deterministic [12, 13], probabilistic [14, 15], and simulated annealing [16]. In our work, we used an improved crowding method to achieve the aim with a shorter time and good quality. We achieved our goal by selecting the chromosome for a maximum of three times to prevent repetitive selection and provide an opportunity to check another chromosome that may obtain a better result.

The rest of the paper is presented as follows; section 2 presents the theoretical background of fractal, fractal inverse problem, and *PIFS*. The detailed explanations on the fractal image compression, collage theorem, and Jacquin approach for fractal image coding are discussed in sections 3. The GA and its relationship with the fractal image compression is introduced in section 4. Crowding method and its improved version is introduced in section 5. The implementation and the analysis of the results is discussed in section 6. Finally, the work is concluded in section 7.

II. BASIC CONCEPTS OF FRACTAL IMAGE CODING

The theory of self-affine transformation and self-similarity is the bases that fractal image coding depends on. In this section, we introduce the theoretical basis for fractal image compression, such as the *IFS*, contraction mapping, and fixed point theorem.

A. Self-similarity Property

One of the base properties of fractal image is self-similarity. A typical image is said to be self-similar if the image looks “almost” the same on any scale. However, all images do not contain this kind of self-similarity found in fractals and actually contains different sort of similar parts (Distasi et al. [17], Truongx et al. [18]). Figure 1 shows an example of this fractal image.

Self-similar parts in the Lena image are shown in Figure 2, as can be seen in part of her shoulder and the reflection in the mirror with her hat [19]. In this type of image, only a portion of an image is self-similar, whereas, in Figure 1, the whole image is self-similar.

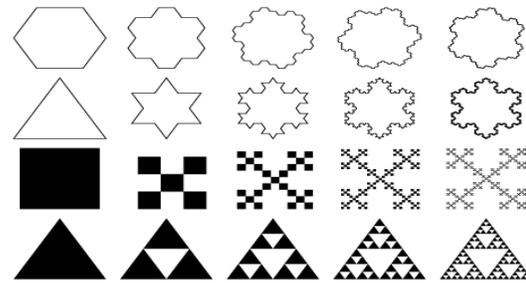


Fig. 1. Fractal image repeated at different locations



Fig. 2. Self-similarity in the Lena image

Now, let (X, d) be a metric space and a sequence (X_n) is called a Cauchy sequence, if for any given $\epsilon > 0$, we have $d(X_m, X_n) < \epsilon$ for all $m, n \in N$ (natural numbers). (X, d) is called complete if every Cauchy sequence in X converges to an element of X .

Readers that are interested in greater detail can refer to [3,20].

Definition 1: Let $f: R^2 \rightarrow R^2$ be a transformation of the form $w(x_1, x_2) = (ax_1 + bx_2 + e, cx_1 + dx_2 + f)$, where a, b, c, d, e , and f are real numbers. This transformation is called a (two-dimensional) affine transformation. The following equivalent notations have been used:

$$w(x) = w_i \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} e_i \\ f_i \end{pmatrix} = Ax + l \quad (1)$$

where $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is a two-dimensional, 2×2 real matrix, and l is the column vector $\begin{pmatrix} e \\ f \end{pmatrix}$, such as $(e, f) \in R^2$.

The matrix A can always be written as follows:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} r_1 \cos \theta_1 & -r_2 \sin \theta_2 \\ r_1 \sin \theta_1 & r_2 \cos \theta_2 \end{pmatrix}.$$

Definition 2: let (X, d) be a metric space, a transformation $f: X \rightarrow X$ is called contractive mapping if $d(f(x), f(y)) \leq s \cdot d(x, y)$ for all $x, y \in X$, where $0 \leq s < 1$ is called contractivity factor of f .

Definition 3: Let $f: X \rightarrow X$ be a transformation on a metric space (X, d) , a point $x_f \in X$, such that $f(x_f) = x_f$ is called the fixed point of the transformation f . The fixed point is highly important; it represents that the part of the shape in which we are interested that is not affected by the transformation.

The Hausdorff metric is an important concept in fractal theory. Therefore, many mathematicians have discussed and proven basic concepts and results of this space [21,22]. The Hausdorff metric is known as the space of fractals and is denoted by $H(X)$. It is generated from the complete metric space X comprising elements that are the compact sets in X . The distance that is defined in $H(X)$ is given as follows.

Definition 4: Let (X, d) be a complete metric space, for the space of fractal $H(X)$, The Hausdorff distance h is defined on this space as follows:-

$$h(A, B) = d(A, B) \vee d(B, A),$$

for any points A and $B \in H(X)$

where $d(A, B) = \max \{d(x, B): x \in A\}$
and $d(x, B) = \min \{d(a, B): a \in B\}$.

The IFS is the most important concept of fractal theory. The IFS was developed by Hutchinson (1981), and then by Barnsley and other researchers [1, 6]. These systems of mapping have been widely discussed and used in many applications, such as image compression. The general formula for IFS is introduced as follows.

Definition 5: Let (X, d) be a complete metric space. An IFS on I is a finite set of contractive self-mappings $w_i: X \rightarrow X$ with respective contractivity factors s_i for $i = 1, 2, \dots, N$, such that $s = \max\{s_i, i = 1, 2, \dots, N\}$. IFS is based on the affine transformations given by

$$w(x) = w \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} r_1 \cos \theta_1 & -r_2 \sin \theta_2 \\ r_1 \sin \theta_1 & r_2 \cos \theta_2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$$

Definition 6: In IFS, any compact subset (fixed point) $A \in H(X)$ is called an attractor for IFS if

$$A = \bigcup_{n=1}^N w_n(A).$$

The fixed point observes existence and uniqueness based on the contraction mapping theorem. The iteration process of the IFS based on any starting image the attractor, which is fully known by the parameters of W .

III. FRACTAL IMAGE COMPRESSION

The FIC approach is an important search area with many possible application fields. This approach is focused on finding fractal code that generates given objects. Barnsley [3] introduced this concept with the well-known collage theorem. When the object is considered as an image, FIC is often involved, which is also known as fractal image coding. The foundation for FIC is the IFS. This problem has been studied by many authors, and a method has been proposed by Jacquine [7] to solve this type of inverse problem. The major problem of standard fractal image coding is its time consumption compared with other image coding methods. Some time is spent in searching for a similar domain block. Therefore, new techniques to solve this problem and accelerate this method are in great demand.

The problem of finding IFS's that used to generate fractal is called an inverse problem. However, if the given set is self-similar, then the required construction is almost straightforward. The IFS can easily be found by conducting mathematical translation of the property of self-similarity. This solution is verified in the collage theorem, which is the first step towards solving the inverse problem.

A. The Collage Theorem

This theorem states the process of obtaining the set of transformations that represent an accurate approximation of a fixed image. It is stated as follows.

Let (X, d) be a complete metric space. let $\{X; w_n, n = 1, 2, \dots, N\}$ be an IFS with contractivity factor s , $0 \leq s < 1$ and let L be a closed subset of X such that

$$h(L, \bigcup_{n=1}^N w_n(L)) < \epsilon,$$

for some $\epsilon > 0$, and h is the Hausdorff distance. Then

$$h(L, A) \leq \frac{\epsilon}{1-s},$$

where A is the attractor of the IFSs

B. Jacquin Approach for Fractal Image Compression

FIC depends on the self-similarity property in an image. The main idea comes from the partitioned iterated function system (PIFS), which is an expansion of IFS theory. The difference between the two concepts appears in the application domain. Thus, the main difference is that instead of dealing with the whole image, a specific part is used to obtain the PIFS parameters.

For an original image M of size $m \times m$, it is partitioned into $(m/n)^2$ blocks which are non-overlapping to form a set of range blocks each of them is of size $m \times m$. To comply with the contractive point theorem, the domain block is twice the size of the range block. Hence, a set of $(m-2n+1)^2$ elements, each of them is a block of size $2n \times 2n$ is constructed from M and known as domain blocks. In this case, the partitioned is overlapping. In each search for similarity between the range and domain blocks, two types of blocks emerge from the same image, as shown in Figure 3. As an example, for M of size 128×128 if the size of the each range block is 8×8 , then we have $(128/8)^2 = 256$ range blocks and $(128-2 \times 8+1)^2 = 12,769$ domain blocks of size 16×16 .

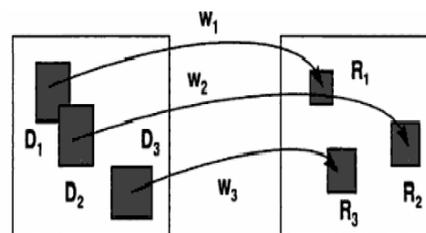


Fig. 3. Clarification of PIFS

The image is equally partitioned by the range blocks, which resulted that each pixel of the image is included in one of the range blocks. However, since the domain block is overlapping, this may cause losing of some pixels. The aim of

this process is to find an approximate domain block for each range block.

By the PIFS technique, the third dimension is appeared that represent the pixel z . after shrinking the domain block to the size of the range block, the eight transformation is applied to resulted in eight different blocks $z_k, k=0, 1, \dots, 7$. These transformations $T_k, k=0, 1, \dots, 7$ can be represented in (2).

$$T_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad T_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad T_2 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}, \\ T_3 = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad T_4 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad T_5 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \\ T_6 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad T_7 = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \quad \dots \quad (2)$$

T_1 and T_2 correspond to the flips of z along the horizontal and vertical lines, respectively. The flip of z along the horizontal and the vertical lines is denoted by T_3 , whereas, an additional flip along the line of the main diagonal is performed by the transformations T_4, T_5, T_6 , and T_7 which are correspond to T_0, T_1, T_2 , and T_3 . Finally $T_0(z) = z$.

In fractal coding, a contrast scaling s and a brightness offset o on the transformed blocks occur, so the fractal affine transformation becomes three-dimensional as shown in (3).

$$W_i \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ o_i \end{bmatrix} \quad \dots \quad (3)$$

We let a_1, a_2, \dots, a_n and b_1, b_2, \dots, b_n be two squares containing n pixel from D_i and R_i , respectively. Here, we minimize the quantity of s and o between of the coded range block X_D and its corresponding coordinates of the domain block Y_D , such as;

$$s = \frac{n^2 \sum_{i=1}^{n^2} a_i b_i - (\sum_{i=1}^{n^2} a_i)(\sum_{i=1}^{n^2} b_i)}{n^2 \sum_{i=1}^{n^2} a_i^2 - (\sum_{i=1}^{n^2} a_i)^2}$$

$$o = \frac{1}{n^2} \left[\sum_{i=1}^{n^2} b_i - s \sum_{i=1}^{n^2} a_i \right]$$

where the minimum distance between R_i and D_i is found by the RMS such that;

$$RMS = \frac{1}{n} \left[\sum_{i=1}^{n^2} b_i^2 + s \left(s \sum_{i=1}^{n^2} a_i^2 - 2 \sum_{i=1}^{n^2} a_i b_i + 2 \cdot o \sum_{i=1}^{n^2} a_i \right) + o \cdot \left(o \cdot n^2 - 2 \sum_{i=1}^{n^2} b_i \right) \right] \quad \dots \dots \dots (4)$$

When this error is less than the predefined threshold, the search will be finished.

The execution time of fractal compression implementation is the main problem that most standard algorithms. Fisher's algorithm [4], involves a classification pattern that has been greatly accelerated; however, the resulting image quality is extremely poor because of the search space reduction from the classification used by Fisher. To overcome these problems, some evolutionary algorithms is used to serve in solving these problems, its methodology is introduced in the following section.

IV. GENETIC ALGORITHM

Using a GA is important for obtaining solutions to complicated search problem. In this section, we discuss the relationship between the processing of GA and its operation when dealing with the fractal inverse problem. Holland's 1975 book [10] introduced GAs as a summary of biological evolution and showed the theoretical structure of GAs.

A. Genetic Algorithm for Fractal Image Compression

Searching processes begin after dividing the image into range and domain blocks as in standard Jacquine approach image compression. For each range block, the domain block and identical transformations that best cover the range block is specific. In general, for best correspondence, transformation codes are set, including contrast and brightness. Searching succeeds when the domain block is appropriate for the suitable range block. Eventually, mapping data are stored. Then, we use GAs to attain the intended outcome via fractal compression. Usually, GAs are employed to find the near optimal solution, and thus the GA for fractal compression of images is shown below [10].

1) Chromosomes

Given that, the GA works on the chromosomes, producing chromosomes from the range and domain blocks is a crucial step in using the GA for *FIC*. The transformation parameters obtained for each block are coded on a set of a fixed number of bits. These parameters are then stored as chromosomes. By encoding the parameters of an image, a chromosome comprises N genes that are equal to the number of the non-coded parts of an image. These genes are generated from parameters X_D and Y_D , which refer to the coordinates of the domain block, and the flip, refers to the transformation isometrics. Figure 4 show the chromosomes

Range Block									
Block1			Block2			Block N		
X_D^1	Y_D^1	Flip ¹	X_D^2	Y_D^2	Flip ²	X_D^N	Y_D^N	Flip ^N

Fig. 4. Image representing a chromosome

2) Fitness Function

Fitness function is a specific task for each chromosome, which refers to the capability of each chromosome to survive and proliferate. We denote fitness as the value of error between the coded range and domain blocks that are assigned by the transformation with analogous luminance and contrasting values. The error is computed using the root mean square equation (4).

3) Genetic Operators

Crossover and mutation are two basic operators that are used in all implementations of genetic algorithms. These operators are described as follows.

- **Crossover Operator:** The crossover operator selects two parents based on their fitness, and then attempts to produce a new child with the best possible quality. A high fitness value provides the crossover operator with a high probability of selection. The crossover operator changes the genes of the parent. Given that a random number a is produced in the interval $[0, 1]$, the new coordinates are computed using the equation below.

$$\begin{aligned} \text{First offspring} \quad X_D &= a * X_{D1} + (1 - a) * X_{D2} \\ Y_D &= a * Y_{D1} + (1 - a) * Y_{D2} \quad \dots(5) \end{aligned}$$

$$\begin{aligned} \text{Second offspring} \quad X_D &= (1 - a) * X_{D1} + (1 - a) * X_{D2} \\ Y_D &= (1 - a) * Y_{D1} + (1 - a) * Y_{D2} \end{aligned}$$

- **Mutation Operator:** The mutation operator changes the value of one or more genes in the chromosome, thereby adding completely new gene values to the gene pool. The GA may achieve a better solution using these new values. The mutation operator also introduces the verity in the chromosomes. The information changes randomly based on the mutation rate.

B. Genetic Algorithm for Fractal Image Compression

Fractal image compression algorithm

1. Decompose the input image M into blocks according to Jacquine's technique
2. Begin with FIC parameters, such as range block size, fitness function, error limit, and number of iterations;
3. Begin with GA parameters, such as mutation rate and crossover rate;
4. Set t = some tolerance level;
5. Partition image M into non-overlapping ranges R_i 's and overlapping domain D_i 's;

For each range block R_i in the range, do

- The transformations (a random population of chromosomes) is generated

while number of populations is not the maximum and the optimal domain is not found, Do

- The fitness value is computed for all individuals to be used for search for the optimal domain in the domain pool using the fitness function;
 - when the optimal domain block is found;
 - apply the crossover operator on individuals;
 - apply the mutation operator on individuals; and
 - generate the new population;

end while

The obtained transformation parameters from the search is written in the transformation W

end for

V. CROWDING METHOD

De Jong [23] introduced crowding as a general technique for maintaining population variety and early convergence. Crowding is often used to determine survival of genetic algorithms in order to determine the individuals in the present population and identify the offspring that will pass to the next generation. It is divided into two principal phases, namely, coupling and alteration. In the coupling phase, the offspring individuals are coupled with individuals in the present population based on a likeness metric. Meanwhile, in the alteration phase, the pairs of offspring and individuals that will remain in the population are selected. The main crowding scheme of De Jong [23] involves the random selection of offspring individuals from the present population. The identical selected of individual is used to replace the selected

offspring. Which makes crowding is an improved genetic algorithm is that;

1) In the crowding method, parent selection is not commonly used, therefore, the individuals are randomly paired in the present population. However, in the population, each individual becomes a parent.

2) In the crossover operator, for each pair (P_1, P_2), the parents are recombined with probability P_c . In the mutation operator, the two producing children (c_1, c_2) are mutated with probability P_m , where P_c denotes crossover probability, P_m denotes mutation probability, and M denotes population size [13].

3) The population of the next offspring includes one of the two parents that complete with each child.

4) The distance between two individuals i_1, i_2 is denoted by $d(i_1, i_2)$.

If $d(p_1, c_1) + d(p_2, c_2) < d(p_1, c_2) + d(p_2, c_1)$

$p_1 \leftarrow$ win the emulation between p_1 and c_1 .

$p_2 \leftarrow$ win the emulation between p_2 and c_2 .

Else

$p_1 \leftarrow$ win the emulation between p_1 and c_2 .

$p_2 \leftarrow$ win the emulation between p_2 and c_1 .

For survival, each offspring oriented to fight with its most identical parent. Other variants exist when more than two parents and children are selected before applying the resemblance metric [26]. This idea is the basis of several widely applied modern crowding approaches. The difference between these approaches is used to determine the winner in each competition.

B. The Proposed Crowding Algorithm for FIC

The crowding method [23] is proposed to eliminate the selection process and introduce a preselecting process. This will cause in a very fast GA to be used for multidimensional optimization problem. By reducing the selection process, the individuals are mutate randomly with any other population individuals. During the replacement process the pairing between the offspring and one of the parents is performed first. This operation is done with probability P_c . This pairing process is happened according to the similarity between them. In the evaluation step, the fitness function which represented the least square error between the offspring and the parents is responsible for deciding about which individual of the population is allowed to stay.

In this section, we proposed an improved crowding method in order to be applied to improve FIC . With this method the diversity is preserved in the population with the opportunity for each individual to be a parent. What distinguish our proposed method from original crowding method [23] and Mahfoud method [13] is some technical differences in the main phases of the algorithm. The population set $\{T_1, T_2, \dots, T_n\}$ is constructed by finding all construction mapping T_i that resulted from the similarity measure between the range block and domain block of the query image. This set is calculated using Jacquine approach [7]. Each individual T_i is assigned a fitness value $f(T_i)$ as its weight, where $f_i \in \{f_1, f_2, \dots, f_n\}$ represents the minimum distance

between R_i and D_j , $j=1, \dots, m$. This value is used in the selection process of the parents $\{P_1, P_2\}$ and controlled by a chosen factor known as crowding factor that determine the number of the maximum selection of this individual as shown in the following diagram.

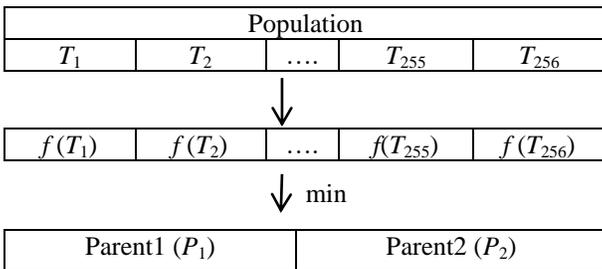


Fig. 5. Selection process

Crowding algorithm

Input: Image I, M % M is known as a crowding factor
 p_c % crossover probability
 p_m % mutation probability
Output: New optimized population $cp = \{T'_1, T'_2, \dots, T'_n\}$

Begin
 Genetic crowding population $cp = \{T_1, T_2, \dots, T_n\}$
 Evaluate (cp) % calculate the fitness for cp
 while terminated () do
 $P = \text{selection}(cp, M)$ % select parents $P = \{T: T \in TP, \text{ such that, } RMS(T(i), i) < \epsilon\}$
 Offspring = recombination (cp, p_c, P)
 Offspring = mutation (offspring, p_m)
 Compare offspring with parents and add the best one to the cp
 end while
 end.

In the selection phase, the parents are chosen according to the minimum fitness values, where the maximum number of selection of each individual is equal M (crowding factor).

Selection function

Input: c_p % crowding population
 M % maximum number the selection
Output: P % parents

Begin
 for $i = 1$ to n % n is the size of c_p
 $p_i = \frac{f_i}{\sum_{j=1}^n f_j}$ % f is the fitness
 end
 while $z = \text{false}$ do
 Select the best two individual P from c_p according to p_i and call it P
 If $p_c \cdot f \leq M$ then
 $Z = \text{true}$
 end if
 end while
 end

In the recombination phase, the offspring is generated according to two logical values as shown in the following algorithm.

Recombination function

Input: c_p % crowding population
 p_c % crossover probability
 p_m % mutation probability
output: offspring

begin
 select $\{p_1[x], p_2[x]\}, \{p_1[y], p_2[y]\}, \{p_1[f], p_2[f]\}$
 case 0 0
 $p_2[x] = p_1[x], p_2[y] = p_1[y], p_2[f] = p_1[f]$
 case 0 1
 $p_1[x] = p_2[x], p_1[y] = p_2[y], p_1[f] = p_2[f]$
 case 1 0
 $p_1[x], p_2[x], p_1[y] = p_2[y], p_1[f] = p_2[f]$
 case 1 1
 $p_2[x] = p_1[x], p_2[y] = p_1[y], p_2[f] = p_1[f]$
 end select
 end

After the recombination phase, the resulting offspring are competed with their parents the mutation phase for surviving. The decision of winning is taken based fitness value (the similarity measure between the offspring and the parents) in order to decide the one that should in the new population, such that: If $RMS(P, C) < \epsilon$ then C is the winner of the competition

else P is the winner of the competition

Mutation function

Input: P_m % mutation probability
 Offspring

Output: offspring

begin
 $x, y, f = \text{rand}()$ % generate random number
 if $x > y$ and $x > f$ then
 $x = \text{rand}()$
 else if $y > x$ and $y > f$ then
 $y = \text{rand}()$
 else if $f > x$ and $f > y$ then
 $f = \text{rand}()$
 end if
 end

The termination value of the algorithm is deduced according to learning process on a sample of different images to determine the best that can satisfy the compromising between the optimum solution and the execution time.

VI. IMPLEMENTATION AND ANALYSIS

A. Implementation

The proposed system was established using Matlab Ver.8.2 and then tested on an pc with cor i7, 2.5 GHz and 8 GB RAM, windows 10 pro. The proposed system was tested on five 8-bit gray images of size 512×512. We tested the proposed system on three ranges, namely, 2, 4, and 8. The RMS of the decoded image partitioned by the 8×8, 4×4 and 2×2 block sizes. A smaller block size indicated a smaller

appropriate error for the affine transformation. The partitions of range blocks were calculated according to $(m/n)^2$, while the partitions of domain blocks were calculated according to $(m-2n+1)^2$. Table 3 illustrates the coding, decoding, time, and compression ratio of the selection images using the proposed technique, while Table 4 illustrates the peak signal-to-noise ratio (PSNR) and MSE of some selection images using the proposed technique.

B. Analysis

The results of the abovementioned algorithms in terms of compression ratio, quality, and implementation time (coding time) are compared for the genetic and crowding *FIC* algorithms with the standard *FIC* algorithm as shown in Tables 1-6. The comparison was performed on an image with a range pool containing 16384 range blocks of size 4x4 and a domain pool containing 255025 domain blocks of size 8x8. Table 5 presents the comparison results.

In the chosen images, the *PSNR* is inversely proportional to *MSE*, and the compression ratio is proportional to that value. A small range block size resulted in a higher compression ratio. The time for producing the image depends on how much error is allowed in the transformations. The employ of the image determines the required amount of compression and the image quality. The predefined number *M* is used as an indicator that determine the number of times for selecting the individual as a parent. This modification in the selection in new generation. This diversity in the population that achieved by the crowding method resulted in some advantages, which are:-

- 1) Through the search, different local maxima can be achieved.
- 2) The diversity is maintained.
- 3) For different crowding factor, the subpopulation is almost stable.

The replacement process is responsible about picking the new individual to construct new population.

By applying the proposed crowding method the following results is obtained. Tables 1-6 represent the analysis of the results for some chosen images.

TABLE III. CODING, DECODING TIME, AND COMPRESSION RATIO BASED ON THE PROPOSED METHOD FOR DIFFERENT RANGE SIZE

Images	Range	Coding Time	Decoding Time	Compression Ratio
	2x2	5.66	2.10	3.12
	4x4	0.99	0.26	7.04
	8x8	0.18	0.18	12.38
	2x2	5.98	2.84	2.33
	4x4	0.895	0.278	4.33
	8x8	0.26	0.19	7.41
	2x2	6.09	2.97	3.56
	4x4	0.878	0.265	5.72
	8x8	0.29	0.19	7.4
	2x2	5.90	2.02	2.29

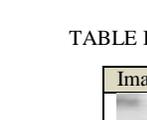
	4x4	0.895	0.269	6.03
	8x8	0.17	0.17	10.1
	2x2	6.158	2.83	3.21
	4x4	0.920	0.261	7.16
	8x8	0.22	0.18	14.06

TABLE IV. PSNR AND MSE FOR DIFFERENT RANGE BLOCK

Images	Range	PSNR	MSE
	2x2	12.11	0.07
	4x4	13.18	0.026
	8x8	13.38	0.04
	2x2	7.13	0.38
	4x4	9.86	0.39
	8x8	9.86	0.51
	2x2	7.03	0.113
	4x4	12.48	0.113
	8x8	10.64	0.113
	2x2	9.06	0.039
	4x4	9.18	0.023
	8x8	10.08	0.049
	2x2	10.01	0.08
	4x4	12	0.039
	8x8	12.89	0.03

VII. CONCLUSIONS

Image compression technique is always in a continuous competition and challenge according to the fast developing of the technology fractal image compression is an emerging technology that based on the fast that most of real world images contain some redundant area that are similar to the other area in the same image. It is basic idea is how to express an image by a set of *IFSs*. The argumentative discussion about compromising between the compression ratio and the contracted image quality is motivation for new optimized technique towards this goal. Genetic algorithm is to be appropriate used to solve of a multidimensional problem that have large search space with no exact solution exist. In this study, we improve this technique by omitting of the parent selection which resulted, each individual becomes a parent. However, the selection process is specified by a pre-defined value known as crowding factor that determine the number of selection of each individual. Therefore, each offspring is randomly selected from the population, and its most identical parent. Comparing the performance of the proposed technique is accomplished through some experiments which show best result over the standard fractal compression technique and standard genetic algorithm technique as shown in tables (6) and charts (1-3). From these figures are can see that RMS error is inversely proportional to the PRNS ratio. They show a good compromise value that resulted in good performances.

REFERENCES

[1] M.F. Barnsley and S. Demko. Iterated function systems and the global construction of fractals. In Proceedings of the Royal Society of London A399, 243 - 275, 1985.

[2] A. E. Jacquin, "A fractal Theory of Iterated Markov Operators with Applications to Digital Image Coding", PhD., Georgia Institute of Technology, 1989.

[3] M. F. Barnsley, "Fractal everywhere", Second edition, Academic Press, 20-100, 1988.

[4] Y. Fisher, Fractal Image Compression: Theory and Application to digital images, Springer Verlag, New York, 1995.

[5] M.F. Barnsley and L. P. Hurd, fractal Image Compression. AK. Peters, Wellesley, Mass, 1993.

[6] M. F. Barnsley, A.D Sloan, A better way to compress images, Byte Mag. 215–223, 1988

[7] A.E. Jacquin, Image coding based on a fractal theory of iterated contractive image transformations. Image Proc., IEEE Trans. 1, 18-30, 1992.

[8] T. Abiko, M. Kawamata, IFS coding of non-homogeneous fractal images using Gröbner basis. Proc. of the IEEE International Conference on Image Processing 25–29, 1999.

[9] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, second edition, Springer-Verlag, 1994.

[10] Holland, J.H.: Adaptation in natural and artificial systems. The University of Michigan Press, 1975.

[11] D. E. Goldberg, "Genetic Algorithm in search, optimization and Machine Learning", Addison – Wesley, reading, MA, 1989.

[12] S. W. Mahfoud. Niching Methods for Genetic Algorithms. PhD thesis, Department of General Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 1995.

[13] S. W. Mahfoud. Crowding and preselection revisited. In R. MÅanner and B. Manderick, editors, Proceedings of the 2nd International Conference on Parallel Problem Solving from Nature (PPSN II), Elsevier, Amsterdam, The Netherlands, pages 27-36, Brussels, Belgium, 1992.

[14] O. J. Mengshoel. "Efficient Bayesian Network Inference: Genetic Algorithms", Stochastic Local Search, and Abstraction. PhD thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, 1999.

[15] O. J. Mengshoel and D. E. Goldberg. Probabilistic crowding: Deterministic crowding with probabilistic replacement. In W. Banzhaf, J. M. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. J. Jakiela, and R. E. Smith, editors, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), Morgan Kaufmann, San Francisco, CA., pp. 409-416, 1999.

[16] S. W. Mahfoud and D. E. Goldberg, "Parallel recombinative simulated annealing", A genetic algorithm, Parallel Computing, 21, 1-28, 1995.

[17] R. Distasi, m. Nappi, and d. Riccio, "R range/domain approximation error-based approach for fractal image compression,"IEEE trans. Image process., 15, 1, 89–97, 2006.

[18] T. K. Truong, j. H. Jeng, i. S. Reed, p. C. Lee, and a. Q. Li, "A Fast encoding algorithm for fractal image compression using the dct inner product," IEEE trans. Image process., 9, 4, 529–535, 2000.

[19] Yancong, Y. and Ruidong, P. "Fast Fractal Coding Based on Dividing of Image", 2010.

[20] J.E. Hutchinson, Fractals and self-similarity, Indiana University journal of mathematics, 30(5), 713-747, 1981.

[21] K. J. Falconar, "The Hausdorff dimension of self-affine fractals", Math. Proc. Comb Phil. Soc. 103, 339-350, 1988.

[22] K. J. Falconar, "Random fractals", Math. Proc. Comb Phil. Soc.100, pp:559-582, 1986.

[23] K. A. De Jong. An Analysis of the Behavior of a Class of Genetic Adaptive Systems. PhD thesis, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI, 1975.

[24] C. Chen, T. Liu, J. Chou, "A Novel Crowding Genetic Algorithm and Its Applications to Manufacturing Robots", IEEE Transaction on Industrial Informatics, 10, 3, 2014.

TABLE I. STANDARD FRACTAL IMAGE COMPRESSION BY JACQUINE APPROACH [7]

Images					
Coding Time	2.03	2.03	2.99	2.98	2.97
Decoding Time	0.20	0.20	0.20	0.20	0.20
PSNR	11.15	7.13	11.74	9.01	11.18
MSE	0.81	0.89	0.94	0.91	0.83
Compression ratio	11.6	9.21	10.82	9.41	12.1

TABLE II. FRACTAL IMAGE COMPRESSION BASED ON GENETIC ALGORITHM

Images					
Coding Time	3.08	1.94	2.01	2.09	1.92
Decoding Time	0.61	0.27	0.93	0.21	0.57
PSNR	12	8.76	11.97	9.93	12.01
MSE	0.129	0.138	0.109	0.262	0.396
Compression ratio	12.6	7.33	8.88	11.53	14.44

TABLE III. CODING, DECODING, TIME, AND COMPRESSION RATIO OF THE SELECTION IMAGES USING THE SUGGESTED TECHNIQUE

Images					
Coding Time	0.99	0.89	0.87	0.89	0.92
Decoding Time	0.26	0.27	0.26	0.26	0.26
PSNR	13.18	9.86	12.48	9.14	12
MSE	0.026	0.39	0.113	0.03	0.039
Compression ratio	7.04	4.33	5.72	6.03	7.16

TABLE V. COMPARISON BETWEEN STANDARD, GENETIC AND PROPOSED CROWDING FOR RANGE BLOCK OF SIZE 4

	Images					
Fractal image compression based on Jacquine method	Coding Time	2.03	2.03	2.99	2.98	2.97
	MSE	0.81	0.89	0.94	0.91	0.83
	Compression Ratio	11.6	9.21	10.82	9.41	12.1
Fractal image compression based on genetic algorithm	Coding Time	3.08	1.94	2.01	2.09	1.92
	MSE	0.129	0.138	0.109	0.262	0.396
	Compression Ratio	12.6	7.33	8.88	11.53	14.44
Fractal image compression based on crowding method	Coding Time	0.99	0.89	0.87	0.89	0.92
	MSE	0.026	0.39	0.113	0.03	0.039
	Compression Ratio	7.04	4.33	5.72	6.03	7.16

