CHAPTER VI

FINDING IFS USING GENETIC ALGORITHMS

6.1 Introduction

The Genetic Algorithm based optimization techniques [13] have been used widely in many applications. In recent times, many people attempted to implement image compression operation using GA method. One of the major disadvantages of fractal image compression is the time it taken to encode an image. The number of comparisons and calculations that are required in order to maintain image quality will increase the amount of time tremendously to compress the image.

As discussed in earlier chapters, the possible optimization strategies for enhancing the speed and performance of IFS algorithm for fractal image compression with a RBRS mechanism have been experimented. An enhanced strategy is proposed for further increase in the performance of IFS coding by implementing Genetic Algorithm based search while selecting suitable fractal seed blocks and transformations for IFS coding. For implementing this idea, a quadtree decomposition scheme is used for partitioning the image for block matching operation. The proposed changes increase the overall performance of fractal IFS coding significantly.

The encoding process consists of approximating the blocks from the input image called domain blocks through some operations. As a result of this encoding process, separate transformations for each range block are obtained. The set consisting of these transformations, when iterated upon any initial image, will produce a fixed point (attractor) which approximate the target image. Several algorithms with different motivation have been suggested to obtain IFS code or fractal code of a given image. A faster algorithm to obtain IFS coding has been developed which is faster than the conventional Normal IFS algorithm. Actually, the efficiency of the proposed algorithms
in terms of computational time depends on the selection of genetic parameters. The main purpose of using this GA is to find the appropriate transformation for a range block to generate the IFS coding of an image.

6.2 Genetic Algorithms

John Holland of the University of Michigan first introduced the original concept of Genetic Algorithms in the 1970s. As the name denotes, Genetic Algorithms try to imitate the concept of evolution observed in nature [46] artificially. In a standard approach of solving an unconstrained or constrained problem using Genetic Algorithm, the common way is to form that problem for obtaining a solution by imitating the natural evaluation of organisms. His first achievement was the publication of *Adaptation in Natural and Artificial System* in 1975. By then, Holland had two aims:

1. To improve the understanding of natural adaptation process, and
2. To design artificial systems having properties similar to natural systems.

In this research, the Genetic Algorithms based technique [97] has been explored to improve IFS coding for fractal image compression in terms of speed and quality. Various parameters of a typical Genetic Algorithm such as number of Population, Crossover Point, Crossover-level, Mutation Level, etc., are altered to arrive at a better result in a comparatively shorter time to achieve this.

6.2.1 GAs versus Traditional Methods

GA differs substantially from more traditional search and optimization methods. The four most significant differences are:

- GAs search a population of points in parallel, not a single point.
- GAs do not require derivative information or other auxiliary knowledge; only the objective function and corresponding fitness levels influence the directions of search.
GAs use probabilistic transition rules, not deterministic ones.

GAs work on encoding the parameter set rather than the parameter set itself (except in where real-valued individuals are used).

It is important to note that the GA provides a number of potential solutions [33] to a given problem and the choice of final solution is left to the user. In case, where a particular problem does not have one individual solution, for example, a family of Parento-optimal solutions, as is the case in multi objective optimization and scheduling problems [100], then the GA is potentially useful for identifying these alternative solutions [76] simultaneously.

6.3 The Proposed Algorithm

Iterated Function Systems are the foundation for fractal image compression. In this section, a standard IFS algorithm for fractal coding is experimented.

Applying a Genetic Algorithm in IFS involves the following steps:

- Choosing a space of "potential answers" for one's problem;
- Determining an appropriate measure of "fitness" on this space;
- Defining appropriate genetic operators on this space, for instance crossover and mutation operators.

The algorithm then involves iteration generation in a population of finding potential answers. At each stage, selecting certain population elements using some parameters to generate new elements are based on genetic operators. The idea in this selection process is to provide an intuitive simulation of the biological process of evolution by natural selection. The measures must be taken to ensure that, on the whole, the "fittest" elements are chosen to reproduce population. The simplest way to do this is to select each element with probability proportional to its fitness.
A digital gray scale image is decomposed using quadtree decomposition as discussed in earlier chapters. After decomposition [85], \( I_{nm} \) number of blocks is available. The blocks which are exactly like another, are removed from the search space of \( I_{nm} \) using the RBRS mechanism. Then there are only \( D_n \) numbers of domain blocks to be used as seed blocks for the fractal IFS coding. The removed blocks \( I_{nm} - D_n \) yields \( R_m \) range blocks. That is,

\[
I_{nm} = D_n + R_m
\]

Binary strings are introduced to represent the elements of \( D_n \). The set of binary strings, each of length \( L \), are constructed in such a way that the set exhausts the whole parametric domain and range space. The value of \( L \) depends on the number of domain blocks and the number of transformations used. The fitness value of a string is taken to be the RMSE between the given domain block and the estimated range block.

If \( P \) be the population size, \( G \) be the maximum number of generations for the GA, then the total number of strings searched up to \( G \) iterations is \( G \times P \). That means \( G \times P \) matching operations are performed for finding one suitable domain block for a range block. Since there are \( D_n \) domain blocks, \( G \times P \times D_n \) matching operations are performed for finding suitable domain block for corresponding range blocks.

### 6.3.1 Quadtree decomposition procedure

Quadtree decomposition is an image analysis technique that partitions an image into homogeneous blocks.

- This decomposition works by dividing a square image into four equal-sized square blocks, and then testing each block to see if it meets some criterion of homogeneity.
- Homogeneity of a block can be determined by using maximum \((V_{Max})\) and minimum \((V_{Min})\) values of that particular block.
If a block meets the criterion \((V_{\text{Max}} - V_{\text{Min}} < \text{threshold value})\), it is not divided any further.

If it does not meet the criterion, it is subdivided again into four blocks, and the test criterion is applied to those blocks.

This process is repeated iteratively until each block meets the criterion.

The result may have blocks of several different sizes.

### 6.3.2 Procedure for RBRS mechanism

- Remove all higher size blocks as the seed blocks of sizes smaller than a particular size is to be used for coding the image.

- Remove all the blocks having variance smaller than a particular threshold value. This will remove all the redundant blocks in the image.

- The remaining blocks will have distinct information which form the main features of the image. These feature blocks are used as seed blocks (Domain blocks) to create the remaining blocks (Range blocks).

### 6.3.3 Procedure for GA based IFS

**GA based IFS**

- Let \(D_n\) be the Domain Blocks, which can be addressed from the index 1 to \(n\).

- Let \(R_m\) be the Range Blocks, which can be addressed from the index 1 to \(m\).

- Let \(P_{\text{tot}}\) be the Total Population;

- Let \(G_{\text{tot}}\) be the Total Number of Generations;

- Let \(M_{\text{level}}\) be the Mutation Level;

- Let \(X_{\text{rate}}\) be the cross-over Rate;
The set of binary strings with the length of $L$ have been constructed in such a way that the set exhausts the whole parametric domain and range space. The value for $L$ depends on the number of domain blocks and the number of transformations used.

For example,

- Two bits are used to denote the intensity transformation of $2^2$ possibilities, (here only two intensity transformations are considered, but in Random IFS four intensity values are considered)
- Three bits are used for representing the $2^3$ geometric transformations and a few more bits are used to represent the index of the domain blocks.

Let $B_{tot}$ be the Total Bits of the binary string.

For $i = 1$ to $m$ # Repeat it for all Range Blocks

Let $P$ be a bit array of size $P_{tot} \times B_{tot}$.

Initialize $P$ with random values.

# Main loop of GA

For $j = 1$ to $G_{tot}$ # Repeat it for all Generations

# Evaluate objective function for each individual

~ Evaluate the Population $P$ to find the best domain block using the Fitness function.

~ Take the Root Mean Square Error (RMSE) between the given range block and the estimated range block for the fitness value of a string.

# Generate next population via selection, crossover and mutation

~ Select two parents based on their scaled fitness values.
Perform crossover operation with respect to the crossover rate $X_{rate}$.

- Crossover is nothing but exchanging bits at randomly selected positions.
- Mutate all except the parents with respect to Mutation Level $M_{level}$.
- Mutation is nothing but flipping binary values at randomly selected positions.
- After Crossover and Mutation, there will be a new population $P$.

End,

Formulate the IFS code [28] from the bit string and record it.

End.

6.3.4 The stepwise operation

The GA based method has been implemented on 128 X 128 and 256 X 256 size images of various types. To make the encoding process faster, the GA based approach is used to perform the transformation selection operation. In this present work the number of iterations used is fixed for the termination of GA based IFS coding process. The implementation procedure is explained in various steps as given below.

[1] Take an input image 'I' of a particular size crop it as a square image which is suitable for quadtree decomposition.

[2] Decompose the input image into a number of non-overlapping blocks of various sizes based on its features and other details using quadtree decomposition.

[3] Remove redundant blocks from all size groups by leaving only one seed block as the seed block (domain block) of sizes smaller than a particular threshold value are used for coding the image.

[4] Consider the image as ‘Dn’ domain blocks (seeds) of various sizes and ‘Rm’ range blocks of various sizes.

[5] Derive a population of binary strings is derived by crossover and mutation operation
on randomly selected binary string which denotes the geometric transformation, intensity transformation, and the domain block address.


[7] Test the fitness of the entire population. In this case, the fitness function is a simple RMSE, which is calculated for a selected range block and the individual blocks of the new population of strings.

[8] Select only two best candidate binary strings among the whole population and allow them to generate new population.

[9] Repeat the steps from 5 to 6 for few generations.

[10] Find the binary string that has the best fitness value after ‘n’ generations.


[12] Repeat all the steps from step-5 for each and every range blocks.

6.4 Experimental Results

The implementation of both Normal IFS and GA based IFS method with eight and ten transformations are considered for IFS fractal coding [89]. For finding a match of a domain block to a range block, the range blocks are searched to select the appropriate one using Genetic Algorithm approach. To find the best-matched domain block as well as the best-matched map, one has to search all possible domain blocks. The first step is transformation of rows and columns from domain blocks to range blocks. This can be achieved by using any one of the eight transformations on the domain blocks. These eight transformations are all isometric transformations since they preserve distances. Once the first part is obtained, second part is an estimation of pixel values of a range block from...
the pixel values of the transformed domain block. These estimates can be obtained by fitting a straight line using the method of least squares.

During the experiment it is observed that the number of transformations used in fractal IFS coding [17] has significant effect on the overall performance of a Normal IFS search method. But it has only negligible effect on Genetic Algorithm based approach. The experimental results of GA based approach for 128 x 128 size image is given in Table 6.1. In this approach, eight geometric transformations and two intensity transformations have been considered. The comparative study of Normal IFS and GA based approach in terms of compression time and quality measured in PSNR (db) is presented in the Table 6.1.

<table>
<thead>
<tr>
<th>Image</th>
<th>Total Pixels</th>
<th>Pixels used for fractal Coding</th>
<th>Normal IFS</th>
<th>GA Based IFS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Transform s</td>
<td>Time in Min.</td>
</tr>
<tr>
<td>Face1</td>
<td>16384</td>
<td>7936</td>
<td>8</td>
<td>2.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>4.9</td>
</tr>
<tr>
<td>Face2</td>
<td>16384</td>
<td>8896</td>
<td>8</td>
<td>2.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>4.9</td>
</tr>
<tr>
<td>Face3</td>
<td>16384</td>
<td>6784</td>
<td>8</td>
<td>2.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>5.0</td>
</tr>
<tr>
<td>Lena</td>
<td>16384</td>
<td>11776</td>
<td>8</td>
<td>2.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>5.2</td>
</tr>
<tr>
<td>Nature</td>
<td>16384</td>
<td>6592</td>
<td>8</td>
<td>1.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>4.8</td>
</tr>
</tbody>
</table>

To analyze more about this GA based approach, another 256 x 256 size images are also taken and experiments have been carried out. In this case also, eight geometric transformations and two intensity transformations are considered for the execution. The obtained numeric values in terms of compression time and quality in PSNR (db) are presented in the Table 6.2. For both 128 x 128 and 256 x 256 size images eight geometric
and two intensity transformations are considered and experiments have been conducted. Since this algorithm for eight geometric transformations with four intensity transformations takes enormous time, the compression time taken is ignored in the Tables 6.1 and 6.2.

**TABLE 6.2**

EXPERIMENTAL RESULTS IN TERMS OF TIME AND QUALITY IN NORMAL IFS AND GA BASED ALGORITHM FOR 256 X 256 SIZE IMAGES.

<table>
<thead>
<tr>
<th>Image</th>
<th>Total Pixels</th>
<th>Pixels used for fractal Coding</th>
<th>Transform</th>
<th>Normal IFS</th>
<th>GA Based IFS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Time in Min</td>
<td>PSNR in db</td>
</tr>
<tr>
<td>Face1</td>
<td>65536</td>
<td>10048</td>
<td>8</td>
<td>18.9</td>
<td>34.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>38.9</td>
<td>34.7</td>
</tr>
<tr>
<td>Face2</td>
<td>65536</td>
<td>15488</td>
<td>8</td>
<td>17.2</td>
<td>33.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>35.5</td>
<td>33.5</td>
</tr>
<tr>
<td>Face3</td>
<td>65536</td>
<td>10800</td>
<td>8</td>
<td>19.1</td>
<td>34.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>36.7</td>
<td>33.9</td>
</tr>
<tr>
<td>Lena</td>
<td>65536</td>
<td>15632</td>
<td>8</td>
<td>17.8</td>
<td>33.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>35.7</td>
<td>33.4</td>
</tr>
<tr>
<td>Nature</td>
<td>65536</td>
<td>11600</td>
<td>8</td>
<td>18.0</td>
<td>34.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8+2</td>
<td>37.2</td>
<td>34.7</td>
</tr>
</tbody>
</table>

In GA based method, the selection of the parameters such as the crossover ratio and mutation rate, population size, and the number of generations have a significant effect on the results in terms of image quality. Among these parameters, change in population size and the number of generations [97] have a significant impact on both image quality and overall encoding time. The following parameter values have been used during the implementation of GA based algorithms. The values of these parameters were adjusted and different kinds of experiments have been performed.

The number of Transformations used = 10,
The Population Size = 10,
The Number of Generations = 10,
The Crossover Rate = 0.1,
The Mutation Level = 0.5
To analyze the performance of the GA based approach, different kinds of graphs are drawn and presented. The graphs 6.1 and 6.2 indicate the comparative study in terms of compression time for 128 x 128 size and 256 x 256 size images for GA based approach and Normal IFS algorithm. Similarly, the graphs 6.3 and 6.4 indicate the comparative study in terms of quality in PSNR (db) for 128 x 128 and 256 x 256 size images of GA based approach and Normal IFS algorithm. For example, the Normal IFS algorithm with eight transformations for Lena image of size 256 X 256 consumes 17.8 Min. and the quality of a decoded image in PSNR is 33.2 db. For the similar image in GA based algorithm the encoding time is 11.5 min. and the decoded image quality in PSNR is 31.3 db. The time reduction is about 40 % in GA based approach is achieved and the PSNR value is decreased slightly by 2 db.

GRAPH 6.1
Analysis in terms of Time after applying Normal IFS and GA based Algorithm for 128 x 128 size image.
Analysis of Time for Normal IFS and GA based IFS for 256 x 256 size images

GRAPH 6.2
Analysis in terms of Time after applying Normal IFS and GA based Algorithm for 256 x 256 size images.

Performance Analysis of Normal IFS and GA based Algorithm for 128 x 128 size images

GRAPH 6.3
Performance analysis in terms of Quality in PSNR after applying Normal IFS and GA based Algorithm for 128 x 128 size images.
Performance Analysis of Normal IFS and GA based Algorithm for 256 x 256 size images

GRAPH 6.4
Performance analysis in terms of Quality in PSNR after applying Normal IFS and GA based Algorithm for 256 x 256 size images

Even though the GA based method consumes a lot of time, it performs better compared to Normal IFS method in certain aspects. For example, In Normal IFS algorithm for Nature image with eight transformations it takes 1.9 minutes, for the same Nature image in 256 x 256 size, GA based approach consumes 4 minutes. At the same time, for 8+2 transformations Normal IFS algorithm consumes 4.8 minutes, but GA based approach took only 4.1 minutes. The study result shows that, the number of transformations used in the GA based approach has very negligible effect in the calculation of compression time. The results in terms of quality in PSNR of the decoded images are comparable. In fact, the results are better than the results of existing Normal IFS algorithm in many cases.

Another significant difference in calculating compression time is the transformation used in both these methods. When eight transformations are used, Normal IFS algorithm consumes very less time, but for the ten transformations, GA based IFS method takes less time. The 12 transformations also have been taken and experimented
and since it takes more computation time, the values are omitted in the table 6.1 and 6.2. The main advantage of using this GA based method is the same amount of time it consumes, when all types of transformations are applied on images of varying amount of details. It is not an important drawback since the data types used in MATLAB for the encoding consumes lot of time for the implementation.

6.5 Randomness in Genetic Algorithm

The Genetic Algorithm has been implemented as a search technique and a study of the characteristics randomness has been analyzed, while performing block-matching operation to find an appropriate range block for each domain block. The test results for Random IFS algorithm and Normal IFS algorithm in a fractal image compression scheme [95] in terms of speed and quality of decoded image is presented in Table 6.3. The study shows that GA based IFS is not purely random, but it has little randomness while selecting appropriate range block for the domain block and it has been presented in Table 6.3.

![Table 6.3](image.png)

**TABLE 6.3**

EXPERIMENTAL RESULTS IN TERMS OF TIME AND QUALITY OF NORMAL IFS AND GA BASED ALGORITHM WITH RANDOM IFS FOR 128 x 128 SIZE IMAGES
In Random IFS approach, four intensity transformations and eight Geometric transformations [36] have been used. Totally, there were 12 transformations used for IFS fractal coding. The Random IFS is choosing the transformation randomly and also performing the matching of domain blocks to the range blocks randomly while finding a match for range block for a domain block. The search is repeated for several iterations up to the number of iterations specified which is approximately equal to the time taken for the 'n' generations for the GA based algorithm. It makes the comparison meaningful and the whole process is repeated for the number of range block times. The comparative study of these three algorithms in terms of time and quality in PSNR (db) is presented in graphs 6.5 and 6.6.

![Analysis of Time for Normal IFS, GA IFS and Random IFS for 128 x 128 size images](image)

**GRAPH 6.5**
Analysis in terms of Time for Normal IFS and GA based Algorithm With Random IFS for 128 x 128 size images.
Performance Analysis of Normal IFS, GA IFS and Random IFS for 128 x 128 size Images

In the Figure 6.2, experimentation of the GA based algorithm on various images of size 128 x 128 is shown. The results of compression in each stage of the image is presented for knowing the algorithm better. In each column of that figure, the Row one is the original image, Row two is the image after quadtree decomposition, Row three is the image after the removal of redundant blocks, Row four is the reconstructed image from Normal IFS coding and the Row five is the reconstructed image from GA based IFS coding. Eventhough, this method consumes lot of time for the compression, the quality of a decoded image obtained is comparatively better. Further analysis is still possible for improving the performance of the algorithm in terms of time.
6.6 Results of High Resolution Images

The performance of the GA based technique in the context of image compression is found to be quite satisfactory. Even though the performance of this GA based strategy is better when compared to Normal IFS algorithm, still it can be improved further. This paved the way for developing new algorithms in the name of Bit-weight sorting and Transformation selection. The experimentation on higher resolution images has been performed for Bit-weight based implementations, since the encoding time in GA based
approach is more. For example, Lena image consumes 1950 seconds for 1024 X 1024 size. The quality of the decoded image is 32 db. Hence, the study on higher resolution images using GA based implementation is not presented here. Further exploration can be done on improving the speed of compression with specified framework for the implementation.

6.7 Summary

A Genetic Algorithm based IFS algorithm with RBRS mechanism has been explored to reduce the overall search time in fractal image compression. The central contribution is a new optimized IFS algorithm that extends ideas from conventional block matching operation. This idea has been implemented and tested using MATLAB successfully. While comparing this with Normal IFS algorithm, the GA based algorithm achieved better performance slightly in terms of speed, but with poorly decoded image quality.

Since the total number of transformations used in the IFS coding has a significant influence on overall speed of a Normal IFS algorithm, the performance of the system is acceptable if a minimum number of transformation or considerably smaller size images are used. In this approach, after selecting a suitable population size and the number of generations, the performance of the system has improved considerably in terms of speed.