Journal of Data Acquisition and Processing
ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.7765895

A NOVEL DOLPHIN BASED FRACTAL IMAGE COMPRESSION TECHNIQUE.

*** S.Elakkiya, **K.S.Thivya***M. Janaki Rani**
* Research Scholar, **Professor, *** Professor
Department of Electronics and Communication Engineering, Dr. M.G.R Educational and Research Institute, Chennai
ece08elakkiya@gmail.com, thivya.ece@drmgrdu.ac.in, janakirani.ece@drmgrdu.ac.in

Abstract— Fractal Image Compression (FIC) is a highly familiarized and effective compression algorithm in the field of image compression. This approach is based on identifying fractals within an image and generating repeating blocks in response to numerical changes. The major drawback of FIC is the significant amount of time required for the encoding process, which can be addressed by using optimization algorithms that are known to converge efficiently, thus optimizing the encoding process for better resource utilization. In this study, a new image compression model inspired by dolphins is presented that utilizes a metaheuristic algorithm. The evaluation of the suggested algorithm is compared to other prevailing techniques which show improvements in aspects like compression ratio, compression time, and PSNR.

Keywords— Image compression, Fractal Image Compression (FIC), metaheuristic, fractal, Dolphin Echolocation Algorithm (DEA).

Introduction

With the advancement of digital technology, the preservation of image data has become crucial. Image compression is a significant field of research because of the rise in demand for data transfer and storage [1]. With image compression, fewer data are needed to represent a digital image thereby reducing the image size. As the size of the image is reduced more images can be saved in a given amount of memory thereby facilitating faster image transmission over the internet. Image compression is not only meant to turn down the image size but also includes the reconstruction of the original image from the compressed data, all while preserving the nature of the original image.[2].The key to successful image compression is removing only the redundant parts of an image[3].

Fractal Image Compression (FIC) is considered as best image compression method due to many factors [4]. FIC leverages self-similarity concepts and has several advantages, including long coding times, fast decoding, high compression ratios, and resolution independence [5].

Recently, researchers have been exploring the potential of combining FIC with various metaheuristic optimization algorithms, such as Genetic Algorithm (GA), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and Bat-Inspired Algorithm (BIA), to improve encoding time and achieve a better compression ratio with improved image quality. This study proposes a novel approach that integrates the Dolphin Echolocation Algorithm (DEA) with FIC to achieve faster encoding and superior results by retaining the image quality as such and reducing the file size to an enormous amount. The orientation of the work is structured as: Section II provides an overview of Fractal Image Compression. Section III presents the hybridized DEA-FIC algorithm. The experimental analysis and performance evaluation is discussed in Section IV. Section V incorporates the summary of the work.

Basics Of Fractal Image Compression

Fractal image compression is a type of lossy image compression that leverages the idea of self-similarity in images. The technique was developed by Michael Barnsley in 1988 and operates by identifying and decoding repeated patterns in the image [6]. The name "fractal" originates from the Latin word "fractus" which means "broken". Fractal compression is based on an iterated function system and has proven to produce high-quality reconstructed images. It has the benefit of using less memory compared to the original image data and also offers resolution independence, as the numerical data can be easily scaled up or down without affecting the quality [7].

Fractal compression works like a special type of photocopying machine, where the output image is obtained by reproducing a copy at each stage of the compression process resulting in size reduction. The transformations used to produce these results are called affine transformation denoted by w_i which allow the input image to be skewed, rotated, stretched, scaled, and translated.

$$w_i \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix} \qquad (1)$$

$$\text{for } i = 1, 2, \dots, N,$$

The coordinates of a two-dimensional image are represented by x and y. The parameters a,b,c and d controls the rotation, scaling, stretching and skewing process, whereas, e and f controls the translation process.

To create fractals using self-similarity, an Iterated Function System (IFS) mathematical model is employed, which is based on the photocopying machine concept. The IFS method uses a collection of contractive affine transformations, where the original image is subjected to each transformation, and the combination of all transformations yields the final result.

Jacquin proposed a successful fractal block coding technique depending on the fact that images contain a high degree of self-similarity at the block-image level [8]. In this, the image is initially portioned into range blocks and domain blocks. This approach involves discovering a series of transformations, which are then applied to produce a fractal image that serves as a representation of the original image with reduced size or resolution while preserving its structural features. This technique also achieves remarkable compression results by saving only the essential transformation parameters and sorting the matching domain blocks during the encoding phase.

Consider a grayscale input image represented by I. The image is divided into N range blocks R_i, $R_i \subseteq I$ for $i = 1, 2, \dots, N$ and M domain blocks D_j, $D_j \subseteq I$ for $j = 1, 2, \dots, M$ where each domain block is twice as big as each range block. Using the minimal mean-squared error criteria, the best-matched D_j block is found initially during the encoding process from the domain pool using the local affine transformations denoted by w_i, as per equation (1) such that $w_i : R_i \rightarrow D_j$. The search continues until the correct combination of transformations is found. Considering the whole image, the affine transformation τ is determined as per the equation (2) and it is formed by compiling the union of the w_i computed for each R_i block

$$\tau = \cup_{i=1}^{N} w_i \qquad (2)$$

Apply this transformation to get the range block that closely resembles the image in domain block using the relation

$$R_i \approx w_i \, (D_j) \qquad (3)$$

The range blocks are then recorded along with the transformations used, and the same process is repeated for other domain blocks.

The final compressed image is created by producing fractal codes for all R_i individually depending solely on the perfectly matched D_j and the resulting output is saved in a codebook.

The decoded image can be obtained in the decoding phase by starting with a blank image, typically a black image with all pixel values set to zero, and iteratively applying the transformations from the fractal code until the original image is reconstructed. The reconstruction process is given by

$$R_i = \alpha\gamma(\varphi \, D_i) + \beta \ I_m \qquad (4)$$

where, $I_m$is an identity matrix with all elements as 1, γ and φ denote contracting and isometric operations, respectively, and α and β represent the scaling and offset parameters.

There are a variety of innovations to the existing fractal compression techniques.

The fractal block coding proposed by Reza Hashemian [9] divides range blocks into edge and shade types, resulting in improvements in both processing time and memory usage. To address the encoding time issue of the traditional fractal scheme, Yi Zhang et al. [10] discussed a wavelet transform-based fractal coding method that combines fractal coding with discrete wavelet transform. This method uses a two-level wavelet transform to divide the image into sub-images, on which basic fractal coding is applied, resulting in high-quality image compression.

T.M. Hasan et al. [11] presented an adaptive fractal image compression methodology that minimizes the sequence of steps pertaining to the operation relevant to match R_i and D_j blocks. Wang et al. introduced a novel fractal image compression (FIC) scheme that categorizes range and domain blocks using the Absolute Pearson's Correlation Coefficient (APCC) and groups domain blocks based on APCC to speed up the matching process. Many other studies have suggested modifications to fractal coding to improve time complexity, stability, and speed. [12] [13].

Recently, many researchers have shifted their focus to the development of metaheuristic optimization algorithms for the improvement of fractal compression techniques.
Metaheuristic algorithms for FIC
The word "metaheuristic" was coined by Fred Glover in 1989, where "meta" refers to beyond or at a higher level, and "heuristic" refers to finding or discovering by trial and error. It is defined as a higher-level search method applied to address complex optimization problems in the presence of incomplete or insufficient data [14].

Nature-inspired metaheuristic algorithms inspiration [15]. The cognitive behaviour of these species can be translated into an algorithm that aids in the solution of challenging issues. Numerous optimization algorithms have been developed in the area of nature-inspired computing to solve complex problems. These nature-inspired metaheuristic algorithms can be used to improve the performance of fractal coding.

Jinjiang Li et al. [16] proposed an ant colony optimization (ACO) algorithm for fractal compression. This algorithm follows the behavior of ants searching for food using pheromone trails left by other ants. The pheromones are placed in R_i and D_j forming a matrix. The ants then search through the matrix to discover the matching D_j for each R_i.

Chun-Chieh Tseng et al. [17] proposed the use of Particle Swarm Optimization (PSO) in fractal compression. Their approach utilizes the edge property of the image to reduce the search process and find the matching domain and range blocks.

Meanwhile, Gohar Vahdati et al. [18] proposed the use of Chaotic Particle Swarm Optimization (CPSO), which employs a Partitioned Iterated Function System (PIFS) to speed up the encoding process while preserving a tolerable level of image quality after compression.

Farao et al. [19] suggested the use of genetic algorithms to optimize fractal compression. The method involves selecting the best offspring from a population based on its fitness value. The algorithm is tested for different crossover rates, and mutation rates, by varying the population count.

Shaimaa S. Al-Bundi et al.[20] introduced crowding optimization as an advancement of the genetic algorithm for fractal compression. This technique incorporates a crowding factor to get close to the optimal solution in one run for the entire search area of the input image.

Menassel et al. introduced the Wolf Pack Algorithm (WPA) [21] and Bat Inspired Algorithm (BIA) [22] for improved fractal compression. The WPA method divides the input image into smaller blocks and uses wolves to search for matching domain blocks, leading to a faster compression time and a better compression ratio. The BIA algorithm uses bats to scan the entire image and compares blocks based on loudness and frequency parameters to find the best-matching domain blocks. This procedure is continued till the whole image is searched and results in improved encoding and decoding time, as well as improved image quality and size.

The Proposed Algorithm

Behaviour of Dolphins

Echolocation: The dolphin is regarded as one of the smartest creatures and has a variety of fascinating biological traits. Though dolphins have good eyesight, it is difficult for them to hunt for food in the sea due to poor lighting conditions. To overcome this problem, they use a technique called "echolocation" to locate their food. A dolphin has the ability to make click-like noises that are unique and have a higher frequency than the sounds used for communication. When these sounds hit an object, they are reflected as an echo. The dolphin produces a new click when it hears an echo which enables them to determine the object's location, size, and shape by measuring the time between the sound emission and echo receipt.

Cooperation and division of labor: Dolphins can showcase their predatory behaviour by working in a group and sharing tasks. When a dolphin encounters an enormous prey, it divides the work in such a way that the dolphins near the prey are in charge of monitoring the prey's movements, and the dolphins far from the prey gather in a circle to encircle the prey.

Information exchanges: Dolphins can communicate with one another using their linguistic system and can express various ideas through the use of noises at various frequencies. The information is usually about the prey, and the entire predation process is divided into three stages. In the first stage, each dolphin works individually with the help of noises to find nearby prey. In the second stage, dolphins trade information with other dolphins when they encounter a big fish. After receiving information, the dolphins move toward the prey and encircle it with other dolphins. In the last stage, dolphins take turns eating the prey, and the predation process is complete.

Dolphin Echolocation Algorithm (DEA)

The DEA was developed by Kaveh and Farhoudi in 2013 [23]. The algorithm applies the principle of dolphin echolocation to find the optimal solution in the search space. Similar to how dolphins search for prey, the algorithm starts by exploring the entire area and gradually narrows down the search as it gets closer to the target. In DEA the optimization is executed in two different phases. The first phase is a global search across the entire search space to identify unexplored areas, while the second phase focuses on refining the results obtained in the first stage using the convergence curve.

Before starting the optimization process, certain input parameters must be set.

- o Loops number (LN)-This refers to the number of times the algorithm is repeated and is proportional to computation cost. Unlike other algorithms, DEA achieves good results with fewer loops as the convergence criteria are reached earlier. The loop number is chosen between 10 to 100.
- o Convergence curve-The convergence curve is a crucial parameter in the DEA algorithm. The goal is to choose a curve that allows the algorithm to reach the final

solution smoothly. It is ideal to choose a linear curve with power=1, but experimenting with a power<1 may also yield good results.

- Effective radius (R_e)- Its value is usually set to a quarter of the size of the entire search area.
- Number of locations (NV)- It is similar to the population size in other optimization algorithms

The general DEA algorithm includes the following steps:

(1) ***Initiate:*** The optimization process starts by generating the Alternatives matrix AL_{NL*NV} through random selection for each $i = 1\ to\ NL$ and $j = 1\ to\ NV$, here NL denotes the total count of locations to be searched and NV denotes the count of variables used. The predefined probability is then calculated using a specific equation.

$$PP(L_i) = PP_1 + (1 - PP_1) * \frac{L_i^{power} - 1}{(LN)^{power} - 1} \qquad (5)$$

where PP denotes the predefined probability, PP_1 denotes the convergence factor of the first cycle and $power$ refers to the degree of the convergence curve.

The symbol L_i denotes the current loop number, while LN represent the total count of loops needed by the algorithm to attain the convergence point.

***(2) Find the fitness of every location**:* In DEA, it is crucial to determine the fitness of every location, as maximizing fitness is always a priority in optimization. The fitness at each location is given by

$$F = 1/(h + 1) \qquad (6)$$

where h denotes the objective function chosen for this optimization.

(3) Determine the accumulative fitness: The accumulative fitness can be calculated by locating the position of $AL(i, j)$ in the jth column of the Alternatives matrix and referring to it as A, for $i = 1\ to\ NL$ and $j = 1\ to\ NV$

$$AF_{(A+k)j} = \frac{1}{Re} * (R_e - |k| F(i) + AF_{(A+k)j} \qquad (7)$$

for $k = -R_e\ to\ +R_e$

where, A refers to accumulative matrix, $AF_{(A+K)j}$ is the value accumulative fitness for $(A + K)th$ alternative that is chosen for the jth variable, R_e points to the effective radius, which should not exceed ¼ of the search space, and $F(i)$ represents the fitness of the i^{th} location.

(4) Best Location: The next step is to determine the best location for the present ongoing loop and identify the alternatives associated with the variable of the best location.

Calculate the probability for the variable $j = 1 \ to \ NV$ by varying i from $1 \ to \ NA$ (number of alternatives) in accordance with the below equation

$$P_{ij} = \frac{AF_{ij}}{\sum_{i=1}^{LAj} AF_{ij}} \qquad (8)$$

In the next step, a probability of $\mathrm{P_{ij}} = \mathrm{PP}$ is assigned to the alternatives of all the variables involved in the selection of the best location. Meanwhile, a probability of $\mathrm{P_{ij}} = (1 - \mathrm{PP})\mathrm{P_{ij}}$ is assigned to other alternatives that are not related to the best solution.

(5) Termination: Based on the probability assigned to the other alternative, find the next step location. Repeat steps 2-9 by varying the loop number LN till all the locations are searched, and then the loop is terminated.

Proposed DEA-FIC algorithm.

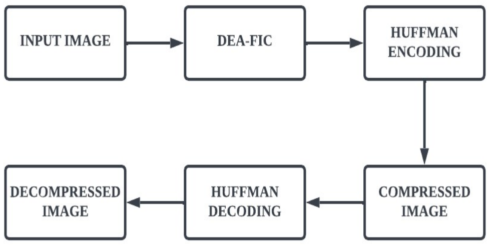

Fig. 1. The general block diagram of the proposed system

The DEA –FIC scheme follows the below steps:

Step 1- The input image is portioned into overlapping blocks of size $n \times n$, referred to as range blocks R_i and domain blocks D_i

Step 2- The dolphins explore each R_i to find its corresponding matching D_i of size $2n \times 2n$.

Step 3- Determine the accumulative fitness of each domain block from the entire pool of domain blocks using equation (7).

Step 4 – The entire search space is viewed as the sea, and for each R_i block the D_i block with the highest fitness value is chosen and considered the prey.

Step 5 - The optimal solution is determined by choosing the location with the highest fitness value and discarding the smaller blocks.

Step 6 – Steps 2 to 5 are repeated for a number of times that was predetermined at the start of the process based on the loop number value.

Step 7- The search continues until a matching domain blocks is found for all range blocks.

Step 8- The block size, location, and fitness value are stored using Huffman encoding

Step 9- The original image is reconstructed using Huffman decoding.

Results and Discussion

The proposed algorithm was executed on five images, namely Lena, Mandrill, Peppers, Fruits, and Butterfly. The performance was evaluated using parameters like compression ratio, compression time, PSNR, and MSE.

The tests were conducted on a personal computer equipped with a 10th Generation Intel® Core™ i3-1115G4 processor, 8 GB DDR4 memory, and the code was executed in Matlab 2021a.

The DEA algorithm uses a set of parameters that has to be predetermined before beginning the optimization phase. The parameter includes selecting the convergence curve which is a smooth linear curve with power=1. The selection of the loop number is also an important aspect. Table I displays the results for different loop numbers using the Lena test image.

TABLE I. TEST RESULT WITH DIFFERENT LOOP NUMBERS

Image	Loop Number	Compression time (s)	Compression ratio	PSNR (db)	MSE
Lena (256* 256)	10	6.657	10.253	34.46	6.89
	20	6.062	10.946	35.21	6.53
	30	5.478	10.987	36.49	6.99
	40	5.002	11.004	36.1	5.21
	50	4.892	11.157	36.89	4.8
	60	4.488	11.204	37.2	4.74
	70	4.452	11.216	37	4.87
	80	4.426	11.185	36.15	4.96
	90	4.415	11.199	35.84	4.96
	100	4.487	11.197	36.23	4.87

From Table I it is ideal to choose the loop number value of 60, as it points to a better result. The number of locations is fixed at 30.

Table II shows the analysis of the proposed technique for different resolutions of the sample images and the quality measurement for these resolutions.

Fig. 2(a) - 6(a) shows the original images, and Fig. 2(b) - 6(b) shows the equivalent decompressed images after the application of the proposed DEA-FIC compression technique.

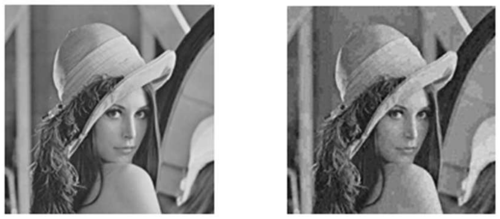

Fig. 2. Lena (a) original image (b) decompressed image

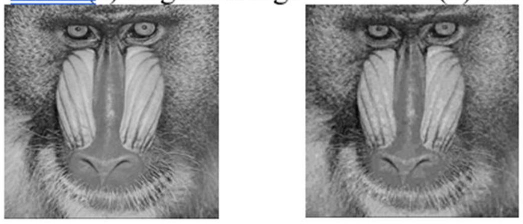

Fig.3. Mandrill (a) original image (b) decompressed image

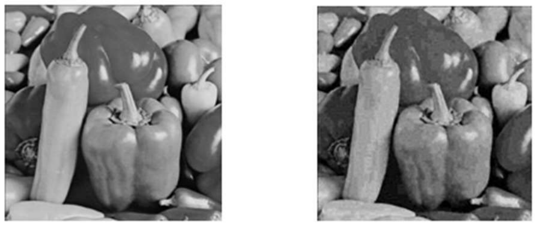

Fig.4. Peppers (a) original image (b) decompressed image

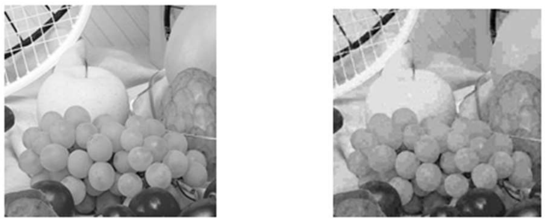

Fig.5. Fruits (a) original image (b) decompressed image

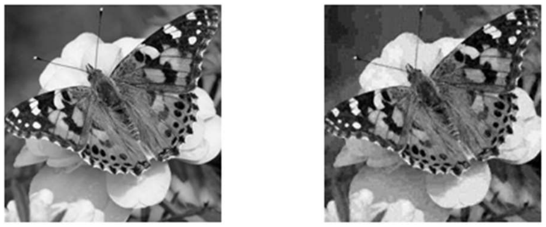

Fig.6. Butterfly (a) original image (b) decompressed image

TABLE II. TEST RESULT OF PROPOSED SYSTEM WITH DIFFERENT RESOLUTIONS

Image	Resolution	Compression time(s)	Compression ratio	PSNR (db)	MSE
Lena	32*32	0.022	12.285	32.946	5.427
	64*64	0.094	11.903	34.907	5.246
	128*128	0.532	11.525	36.551	4.926
	256*256	4.488	11.204	37.202	4.740
Mandrill	32*32	0.029	9.953	30.861	6.973
	64*64	0.092	9.473	32.749	6.229
	128*128	0.293	8.865	33.403	5.614
	256*256	3.421	8.772	34.822	5.008
Peppers	32*32	0.022	11.956	32.831	6.527
	64*64	0.079	11.439	34.154	6.075
	128*128	0.383	11.041	36.403	5.646
	256*256	3.640	10.823	37.443	4.935
Fruits	32*32	0.034	13.789	33.770	5.852
	64*64	0.118	13.327	35.188	5.257
	128*128	0.488	12.900	36.502	4.824
	256*256	4.635	12.431	36.905	4.414
Butterfly	32*32	0.028	10.796	31.871	7.321
	64*64	0.081	10.684	33.658	6.657
	128*128	0.392	9.924	35.751	6.127
	256*256	3.907	9.210	36.128	5.029

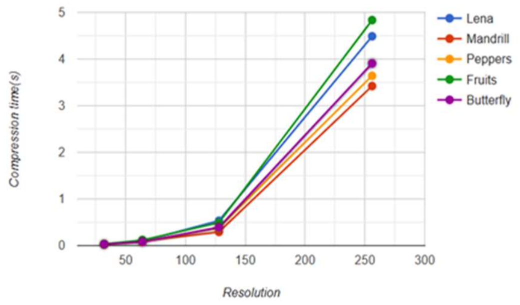

Fig.7. Resolution vs Compression time(s)

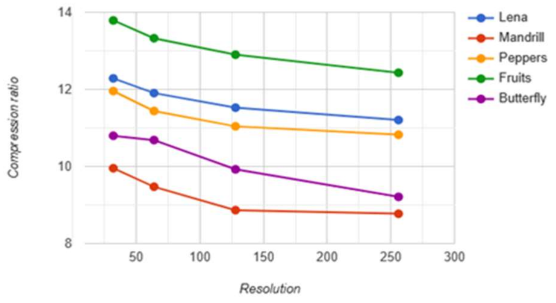

Fig.8. Resolution vs Compression ratio

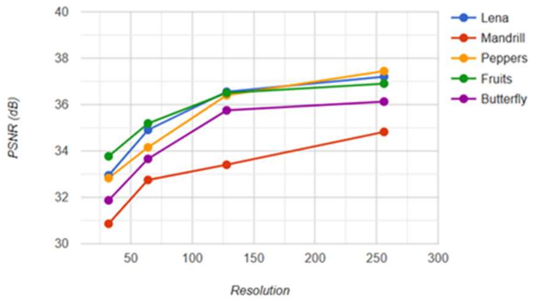

Fig 9. Resolution vs PSNR

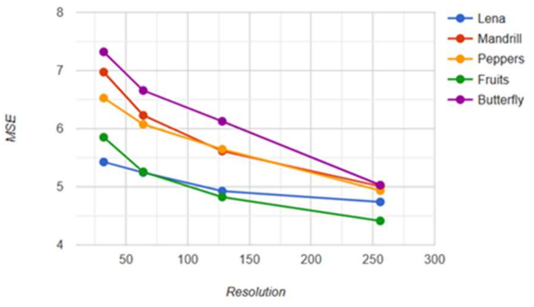

Fig 10. Resolution vs MSE

Fig.9. and Fig.10. shows the plot of resolution vs PSNR and MSE. It is evident that MSE decreases as the resolution increases and PSNR increases as the resolution improves. This indicates that the standard of the compressed image is good, as indicated by the high PSNR and low MSE values. The highest PSNR value of 37.443 was obtained for the 256x256 Peppers image, and the lowest MSE value of 4.414 was achieved for the 256x256 Fruits image.

TABLE III. ANALYSIS OF THE PROPOSED METHODOLOGY FOR COMPRESSION TIME WITH RESPECT TO OTHER EXISTING METHODS

Test image	Methods	Compression time(s)
Lena	DEA-FIC	4.488
	ACO[16]	184
	PSO [17]	115.52
	GA[19]	44
	BIA[22]	33.376
Mandrill	DEA-FIC	3.421
	CPSO[18]	64
	BIA[22]	22.190
Peppers	DEA-FIC	3.640
	CPSO[18]	64
	GA[19]	45

TABLE IV. COMPARISON OF PROPOSED ALGORITHM WITH OTHER EXISTING METHODS FOR COMPRESSION RATIO AND PSNR

Test image	Methods	Compression ratio	PSNR (dB)
Lena	DEA-FIC	11.204	37.20
	ACO[16]	1.89	24.27
	PSO [17]	6.83	28.08
	GA[19]	9.83	32.23
	BIA[22]	1.604	33.11
Mandrill	DEA-FIC	8.772	37.822
	CPSO[18]	-	19.77
	BIA[22]	1.368	30.737
Peppers	DEA	10.823	37.446

	CPSO[18]	9.9	30.24
	GA[19]	-	28.35

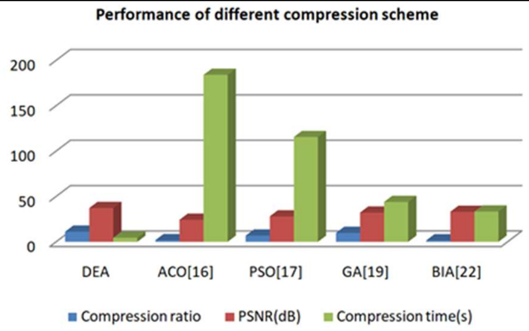

Fig.10. Performance of different compression methods based on their compression rate, PSNR, and processing time.

The comparison of the proposed algorithm with other prevailing techniques for compression time can be seen in Table III. From the results it is clear that the DEA-FIC algorithm is more effective than traditional optimization algorithms in achieving faster convergence rates, resulting in improved compression time.

Table IV presents a comparison of the DEA- FIC algorithm with other metaheuristic algorithms with respect to compression ratio and PSNR. The results reveal that the DEA-FIC algorithm outperforms other existing systems in terms of PSNR and MSE.

The graph in Fig. 10 demonstrates that the DEA-FIC technique has an improved efficiency when compared to other compression techniques. This is due to the fast rate at which the compression process is completed, while still maintaining good values for PSNR and compression ratio.

Conclusion

In this paper, a novel DEA- FIC scheme to compress images is presented and tested with various test images of different resolutions. The algorithm uses a search mechanism based on the dolphin behaviour to find the matching range blocks and domain blocks. The result indicates that the proposed DEA-FIC image compression technique outperforms other standard metaheuristic algorithms in terms of efficiency, with improved results in compression time, compression ratio, and PSNR values which directly indicates that the proposed algorithm results in larger size reduction and works better in preserving the quality of the compressed image.

References

[1] I. M. Pu, “Image compression,” Fundam. Data Compression, pp. 189–210, 2006, doi: 10.1016/B978-075066310-6/50013-1.

[2] A. J. Hussain, A. Al-Fayadh, and N. Radi, “Image compression techniques: A survey in lossless and lossy algorithms,” Neurocomputing, 2018, doi: 10.1016/j.neucom.2018.02.094.

[3] Diya V. Chudasama, Khushboo P. Parmar, Dipali Patel, and Kruti Dangarwala, Shaishav Shah, “Survey of Image Compression Method Lossless Approach,” Int. J. Eng. Res., 2015, doi: 10.17577/ijertv4is030953.

[4] X. Li, S. Zhang, and H. Zhao, “A fast image compression algorithm based on wavelet transform,” Int. J. Circuits, Syst. Signal Process., 2021, doi: 10.46300/9106.2021.15.89.

[5] V. Drakopoulos, “Fractal-based image encoding and compression techniques,” Commun. - Sci. Lett. Univ. Žilina, 2013, doi: 10.26552/com.c.2013.3.48-55.

[6] M. F. Barnsley and S. Demko, “ITERATED FUNCTION SYSTEMS AND THE GLOBAL CONSTRUCTION OF FRACTALS.,” Proc. R. Soc. London, Ser. A Math. Phys. Sci., 1985, doi: 10.1098/rspa.1985.0057.

[7] S. Khatun and A. Iqbal, “A Review of Image Compression Using Fractal Image Compression with Neural Network,” Int. J. Innov. Res. Comput. Sci. Technol., 2018, doi: 10.21276/ijircst.2018.6.2.1.

[8] A. E. Jacquin, “Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations,” IEEE Trans. Image Process., 1992, doi: 10.1109/83.128028.

[9] R. Hashemian and S. Marivada, “Improved Image Compression Using Fractal Block Coding,” 2006, doi: 10.1109/mwscas.2003.1562344.

[10] Y. Zhang and X. Wang, “Fractal compression coding based on wavelet transform with diamond search,” Nonlinear Anal. Real World Appl., 2012, doi: 10.1016/j.nonrwa.2011.07.017.

[11] T. M. Hasan and X. Wu, “An Adaptive Algorithm for Improving the Fractal Image Compression (FIC),” J. Multimed., 2011, doi: 10.4304/jmm.6.6.477-485.

[12] J. Kominek, “Advances in fractal compression for multimedia applications,” Multimed. Syst., 1997, doi: 10.1007/s005300050059.

[13] R. Gupta, D. Mehrotra, and R. K. Tyagi, “Adaptive searchless fractal image compression in DCT domain,” Imaging Sci. J., 2016, doi: 10.1080/13682199.2016.1219100.

[14] Z. Beheshti and S. M. H. Shamsuddin, “A review of population-based meta-heuristic algorithm,” International Journal of Advances in Soft Computing and its Applications. 2013.

[15] A. Singh and A. Kumar, “Applications of nature-inspired meta-heuristic algorithms: A survey,” Int. J. Adv. Intell. Paradig., 2021, doi: 10.1504/IJAIP.2021.119026.

[16] J. Li, D. Yuan, Q. Xie, and C. Zhang, “Fractal image compression by ant colony algorithm,” 2008, doi: 10.1109/ICYCS.2008.222.

[17] C. C. Tseng, J. G. Hsieh, and J. H. Jeng, “Fractal image compression using visual-based particle swarm optimization,” Image Vis. Comput., 2008, doi: 10.1016/j.imavis.2008.01.003.

[18] G. Vahdati, M. Yaghoobi, and M. R. T. Akbarzadeh, “Fractal image compression based on particle swarm optimization and chaos searching,” 2010, doi: 10.1109/CICN.2010.23.

[19] S. Furao et al., “Genetic algorithm applied to fractal image compression,” Electron. Lett., 2014.

[20] S. S., N. M., and N. J., “Crowding Optimization Method to Improve Fractal Image Compressions Based Iterated Function,” Int. J. Adv. Comput. Sci. Appl., 2016, doi: 10.14569/ijacsa.2016.070755.

[21] R. Menassel, B. Nini, and T. Mekhaznia, “An improved fractal image compression using wolf pack algorithm,” J. Exp. Theor. Artif. Intell., 2018, doi: 10.1080/0952813X.2017.1409281.

[22] R. Menassel, I. Gaba, and K. Titi, “Introducing BAT inspired algorithm to improve fractal image compression,” Int. J. Comput. Appl., 2020, doi: 10.1080/1206212X.2019.1638631.
[23] A. Kaveh and N. Farhoudi, “A new optimization method: Dolphin echolocation,” Adv. Eng. Softw., 2013, doi: 10.1016/j.advengsoft.2013.03.004.
[24] Y. Chakrapani and K. Soundararajan, “Implementation of fractal image compression employing particle swarm optimization,” World J. Model. Simul., 2010.