Available online at sjuoz.uoz.edu.krd

Science Journal of University of Zakho

Vol. 13, No.1 pp40–44 January-March,2025

p-ISSN: 2663-628X
e-ISSN: 2663-6298

ACCELERATED FRACTAL ENCODING VIA DOMAIN CLASSIFICATION

Ziyad T. Najim [1, *]

[1]College of Science, University of Garmian, Kalar, Kurdistan Region, Iraq.

*Corresponding author email: ziyad.tariq@garmian.edu.krd

Received: 27 Aug.2024 Accepted: 27 Nov.2024 Published:06 Jan 2025 https://doi.org/10.25271/sjuoz.2025.13.1.1371

ABSTRACT:

The story of fractal image compression is that the image to be compressed is partitioned into many blocks (range pool). Domain pool is created from range pool. Then, each range block is compared with all domain blocks, searching for the best match. This comparison process is highly time-consuming. Many ideas were proposed trying to reduce search time. Here in this paper, a fractal image encoding algorithm based on domain pool classification and domain pool reduction is proposed to speed up the searching process. It works on classifying domain blocks—according to some criteria—into many classes. In this way, range blocks are compared only with the domain blocks that belong to the same class as the range blocks. Experiments showed a considerable reduction in encoding time when compared with the standard Fisher's fractal image compression algorithm while maintaining image quality.

KEYWORDS: Fractal image compression, domain classification, quadtree, encoding time.

1. INTRODUCTION

With the giant development of computer technologies, the need for images has increased. At the same time, that development in technology with the state-of-the-art means of image acquisition devices (cameras, mobiles, etc.) has added more megabytes to represent image pixels. Of course, those extra megabytes will need extra storage space to be saved and extra time to be shared. Hereby, many methods have emerged that aim to reduce image data, and one of these methods is the fractal image compression (FIC). (Menassel *et al.*, 2020; AL-Bundi & Abd, 2020; Fathi & Abduljabbar, 2020; Ahmed *et al.*, 2020)
Fractal image compression has shown great potential in terms of achieving high compression rate and good rate-distortion performance. It can obtain extraordinarily high-rate compression performance superior to that of the DCT-based algorithms. However, due to its complicated algorithm and the time-consuming process, the algorithm has been less attractive and applied less to real-time encoders. However, with the advancement of technology and the development of faster processing units, real-time encoding using fractal encoding is becoming more feasible. The main fast algorithms capable of achieving these impressive compression rates are classification-based fast fractal image compression algorithms. In the fractal coding process, a small domain pool can bring about better encoding features and a smaller file size of the encoded domain. This reduction in domain pool size through classification can lead to more efficient encoding and improved compression ratios. The domain pool reduction through classification proposed in this project, provides a better complexity-quality trade-off in fast fractal image compression. Many papers have concluded different types of domain pool reduction, which efficiently speed up the encoding process through different measures and actions. (Dwivedi & Mishra, 2022; Ahadullah *et al.*, 2020; Singh & Bharany, 2020; Faragallah *et al.*, 2022)

Literature Review:

Fractal image compression is a computationally intensive process that has prompted significant research into methods for reducing encoding time while maintaining image quality. Various strategies have been proposed to address this challenge, focusing primarily on domain pool reduction and classification techniques.

Nithila and Kousalyadevi (2014) introduced a method to speed up fractal image compression by employing classification concept. Their approach involves classifying domain blocks into 72 classes based on pixel intensity orderings and variance, allowing for a more efficient search by comparing only those domains and ranges within the same class. This significantly reduces the search space and thus improves compression speed without sacrificing much accuracy.

Similarly, Revathy and Jayamohan (2012) proposed a dynamic domain classification method that adapts the domain pool for each range block based on its local fractal dimension. By calculating the fractal dimension of range and domain blocks, their method selectively searches for matches within domains that have similar fractal dimensions. This dynamic approach significantly reduces encoding time without compromising image quality.

Building on these foundational works, the current research introduces an approach to accelerate fractal encoding through domain classification based on standard deviation. Standard deviation is selected as a metric for classifying domain blocks because it effectively captures the variability and texture of pixel values within a block. Additionally, this approach helps preserve important image features by ensuring that blocks with similar textural characteristics are matched, thereby maintaining image quality. This static classification approach ensures that only domain blocks with similar standard deviations are considered for matching, thereby decreasing encoding time. Unlike the dynamic adaptation of Revathy and Jayamohan or the extensive class division of Nithila and Kousalyadevi, the current method

* Corresponding author

offers a streamlined process that balances simplicity and efficiency.

Theoretical Background

Fractal image compression is a method that uses the self-similarity properties of images to achieve efficient compression. It is based on the concept of iterated function systems (IFS), which represent images as fixed points of contractive transformations. This technique was initially developed by Michael Barnsley and later refined by Arnaud Jacquin, who introduced a practical algorithm for its implementation (Jacquin, 1992; Barnsley, 1993).

The process involves dividing an image into smaller blocks and finding self-similarities within these blocks. These similarities are encoded using affine transformations, which include scaling, rotation, and translation. The method exploits the redundancy and self-similarity in natural images, allowing for high compression ratios (Fisher, 1995; Wohlberg & de Jager, 1999).

One of the main challenges of fractal image compression is the computational complexity of the encoding process, which requires extensive searching for matching blocks. To address this, various strategies have been developed, such as quadtree partitioning and fast search algorithms. Recent advancements have also explored combining fractal compression with other techniques like wavelet transforms and neural networks to improve performance (Saupe & Hamzaoui, 1994; Wohlberg & de Jager, 1999).

2. METHODOLOGY

The method was performed on four 256x256 images, Cameraman, Pepper, Tiger, and Flower (figure 1), with 8 bits per pixel, and the software simulation was done using Visual C++ on Windows 10, Intel Core i7-2670QM CPU 2.20 GHz platform.

The current study's method focuses on decreasing the time required for encoding through domain pool reduction. Reduction is done as follows:

The maximum standard deviation of range blocks for an image is found, which represents the measure of the distribution of the block's pixel values. Then, that value is divided by the required number of classes in order to create matching groups.

Before searching for the best match in the encoding step, the standard deviation for each range block was compared with that of the domain block to determine whether they both lay within the same group. If this was the case, a search for the best match was executed; otherwise, this domain block would be excluded from the matching process, and the next domain block would be evaluated in the same previous process. This reduction in domain pool size should result in a decrease in encoding time. The algorithm is as follows:

Algorithm:

1. Load Image
- Load a 256x256 grayscale image.

2. Quadtree Partitioning
- Partition the image using the quadtree method. This involves recursively dividing the image into four quadrants until a certain homogeneity criterion is met.

3. Domain Pool Creation
- Create a domain pool by downsampling the original 256x256 image to a 128x128 image using the averaging method.

4. Calculate Maximum Standard Deviation
- Compute the standard deviation for each range block.

$$STD = \sqrt{\frac{\sum_{i=1}^{n}(R_i - R_{mean})^2}{n}} \qquad (1)$$

- Identify the maximum standard deviation among all range blocks.

5. Classify Range Blocks
- Determine the number of classes (*ClassNo*) desired.
- Calculate the group threshold using the formula:

$$G = \frac{Max\_Std}{ClassNo} \qquad (2)$$

- Classify each range block into a group based on its standard deviation and the calculated threshold *G*.

6. Find Best Match for Each Range Block
- For each range block:

a. Determine its group based on its standard deviation.

b. Search for the best matching domain block within the same group:

- Iterate over domain blocks in the same group.
- Compare each domain block with the range block.
- Keep track of the best match based on root mean square (rms) similarity metric.

$$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(R_i - D_i)^2} \qquad (3)$$

c. Select the domain block that best matches the range block.

The rationale for the choice of parameters in the classification and quadtree partitioning processes is detailed below.

The quadtree partitioning method was chosen because it is a well-established technique for dividing an image into smaller blocks based on homogeneity criteria. It ensures that the image is partitioned into blocks of varying sizes, with smaller blocks used for areas of high detail and larger blocks for homogeneous regions.

The quadtree partitioning process in this study was guided by four key control parameters: Maximum Block Size, Minimum Block Size, Ratio, and Inclusion Factor.

The Maximum and Minimum Block Sizes, which represent the largest and smallest allowable sizes, respectively, for a block in the quadtree partitioning process, were chosen to ensure that the partitioning process adapts to the content of the image, capturing fine details in complex regions while reducing the number of range blocks in homogeneous regions.

The Ratio parameter controls how sensitive the partitioning process is to local variations. Lower values allow for more splits, capturing finer details in regions with significant variations, whereas higher values reduce the number of splits, which can lead to faster processing in more homogeneous areas. The Inclusion Factor determines the threshold for splitting a block based on its homogeneity. Lower values allow for more splitting, ensuring that even slightly heterogeneous blocks are divided, which can enhance detail capture in complex areas, whereas higher values result in fewer splits, which can improve processing speed in more uniform regions.

These parameters were predetermined to ensure a balance between computational efficiency and image quality.

The classification of domain blocks was based on their standard deviation, which was chosen as the primary metric because it effectively captures the variability and texture of pixel values within a block. Standard deviation is computationally simple to calculate and provides a reliable measure of the distribution of pixel intensities. This makes it a suitable choice for grouping blocks with similar characteristics, ensuring that range blocks are only compared with domain blocks that are likely to produce a good match.

The number of classes (ClassNo) was set to 32 in this study. The choice of 32 classes was determined experimentally, as it provided a significant reduction in encoding time while maintaining high PSNR values across all test images. A higher number of classes would result in smaller groups of domain blocks, further reducing the search space and encoding time.

However, excessively increasing the number of classes could lead to insufficient domain blocks within each class, potentially degrading the quality of the reconstructed image. Conversely, a lower number of classes would increase the search space, reducing the time-saving benefits of classification.

The group threshold (G) for classification was calculated using the formula (G = MaxStd / ClassNo), where MaxStd is the maximum standard deviation among all range blocks, and ClassNo is the number of classes. This formula ensures that the range of standard deviation values is evenly divided among the classes, creating groups of domain blocks with similar variability.

3. RESULTS AND ANALYSIS

The different tests are performed on four 256x256 images, Camerman, Pepper, Tiger, and Flower, with 8 bpp, and the software simulation is done using VC++ on Windows 10, Intel Core i7-2670QM CPU 2.20 GHz platform. The quadtree partition (Y. Fisher, 1995) is adopted. The image quality is measured by the peak signal-to-noise ratio (PSNR).

The PSNR is defined as follows:

$$PSNR = 10log_{10}\frac{256^2}{MSE} \qquad (4)$$

where MSE (mean-square error)

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2 \qquad (5)$$

where *xi* and *yi* are the pixels in the original and in the reconstructed image at the same position, respectively.

Figure 1: Test images: Cameraman(a), Pepper(b), Tiger(c), Butterfly(d)

Out of the classification methodology applied for the current article, it has been shown, as compared with Fisher's method (Y. Fisher, 1995), that a very good reduction in encoding time has been achieved while keeping good quality as shown in table (1), table (2), and figure (2).

Table 1: The comparison of the proposed algorithm on Cameraman & Pepper pictures

Method	No. of Classes	Cameraman			Pepper		
		No. of Blocks	Enc. Time (sec.)	PSNR	No. of Blocks	Enc. Time (sec.)	PSNR
Fisher's	-	10045	22	26.4541	13060	25	27.5008
Proposed	16		11	26.0983		12	27.4918
	32		9	25.9834		9	27.478
Fisher's	-	8572	18	26.4155	12865	24	27.4989
Proposed	16		10	26.0775		12	27.4967
	32		8	25.956		9	27.4728
Fisher's	-	6283	16	26.3041	7483	19	27.3818
Proposed	16		6	25.9388		7	27.234
	32		5	25.808		5	27.1321

Table 2: The comparison of the proposed algorithm on Tiger & Butterfly pictures

Method	No. of Classes	Tiger			Butterfly		
		No. of Blocks	Encoding Time (sec.)	PSNR	No. of Blocks	Encoding Time (sec.)	PSNR
Fisher's	-	12868	27	26.9809	11809	31	25.74
Proposed	16		10	26.7876		10	25.5603
	32		8	26.4536		8	25.5228
Fisher's	-	8539	20	26.8626	8137	19	25.6345
Proposed	16		6	26.5491		6	25.4298
	32		5	26.2081		5	25.4179

Figure 2: Reconstructed images

As illustrated in the tables, when no classification (Fisher's method) was used and the number of blocks was 10045, the encoding time was 22 seconds, and the PSNR was 27.5008. However, using the classification method with 32 classes for the same number of blocks significantly reduced the encoding time to only 9 seconds with a slight quality reduction of 27.478, which did not represent a significant difference.

The quality remains nearly intact despite the time savings due to the strategic classification of image blocks. By grouping blocks based on their standard deviation, the method ensures that only the most relevant domain blocks are considered during encoding, reducing unnecessary computations without compromising the accuracy of block matching.

The tables show that even with a significant reduction in encoding time, the PSNR values remain close to those achieved with Fisher's method. This indicates that the classification effectively narrows down the search space to blocks that are likely to match well, preserving image quality.

4. DISCUSSION

Fractal image compression techniques face challenges with high computational costs during encoding. The results of this study show that the proposed fractal image compression method, which incorporates domain pool classification based on standard deviation, achieves a significant reduction in encoding time while maintaining good image quality. For example, as shown in Table 1, the encoding time for the Cameraman image was reduced from 22 seconds (using Fisher's method) to 9 seconds with the proposed method, while the PSNR value decreased only slightly from 27.5008 to 27.478. Similar trends were observed across all tested images, including Pepper, Tiger, and Butterfly, confirming the effectiveness of the proposed classification approach in reducing computational complexity without compromising image quality.

The classification of domain blocks based on standard deviation ensures that range blocks are only compared with domain blocks belonging to the same class. By using standard deviation as the classification criterion, the method effectively captures the variability and texture of pixel values within blocks, ensuring that only relevant domain blocks are considered during the encoding process. This not only reduces computational complexity but also maintains high PSNR values, as shown in Tables 1 and 2. For instance, in the case of the Pepper image, the encoding time dropped from 25 seconds to 9 seconds when using 32 classes, with the PSNR value remaining nearly identical (27.5008 vs. 27.478). The slight reduction in PSNR values compared to Fisher's method is negligible and does not significantly impact the visual quality of the reconstructed images. These findings highlight the efficiency of the proposed method in narrowing the search space while preserving the accuracy of block matching.

The results of this study align with and build upon previous research in the field of fractal image compression. For instance, Nithila and Kousalyadevi (2014) demonstrated that classifying domain blocks based on pixel intensity and variance could significantly reduce encoding time while maintaining image quality. Similarly, Revathy and Jayamohan (2012) proposed a dynamic domain classification method based on fractal dimensions, which also achieved faster encoding with minimal quality loss. The current study complements these findings by introducing a simpler, static classification approach based on standard deviation, which balances efficiency and quality without requiring complex calculations or dynamic adjustments.

Conclusion and Future Work:

The current study demonstrates that employing a domain classification method significantly reduces encoding time and enhances compression performance. By categorizing domain blocks based on their standard deviation, the method effectively narrows down the search space during the encoding process. This targeted approach minimizes unnecessary computations by excluding domain blocks that do not match the range block's classification, thereby accelerating the encoding process. Furthermore, this reduction in domain pool size not only speeds up the encoding but also maintains high image quality, as the classification ensures that only the most relevant domain blocks are considered for matching. The results indicate that this method provides a practical and efficient solution for improving fractal image compression, making it a valuable contribution to the field.

Future research could explore alternative criteria for classifying domain blocks to enhance the efficiency of the fractal

encoding process. One approach is to use mean pixel intensity for classification, grouping blocks with similar brightness levels to improve the matching process. This could reduce the search space and lead to faster encoding and better compression performance. Additionally, classifying blocks based on entropy, which measures information content, could help distinguish between detailed and smooth areas, potentially accelerating encoding by excluding less informative blocks. Another promising direction is the use of machine learning-based features, where models could be trained to identify and classify domain blocks based on learned patterns from a diverse set of images, which could optimize the classification process. Exploring these criteria could further refine and optimize the fractal encoding process, contributing to more effective image compression techniques.

Acknowledgments

The author thanks the staff of the College of Science for their support in finalizing this research.

Ethical Statement

Ethical approval was not needed for this reseach.

Author Contribution

Z.T.N., contributed in writing, conceptualization, data collection, and reviewing the manuscript.

Funding

Not included

.

REFERENCES

Ahadullah, M., Sapar, S. H., Al-Saidi, N. M., & Said, M. R. M. (2020). Competitive improvement of the time complexity to encode fractal image: By applying symmetric central pixel of the block. *IEEE Access*, 9, 5028-5045. https://doi.org/10.1109/access.2020.3044290

Ahmed, Z. J., George, L. E., & Abduljabbar, Z. S., (2020). Fractal image compression using block indexing technique: A review. *Iraqi Journal of Science*. https://doi.org/10.24996/ijs.2020.61.7.29

AL-Bundi, S.S. and Abd, M.S., (2020). A review on fractal image compression using optimization techniques. *Journal of Al-Qadisiyah for computer science and mathematics,* 12(1),38.https://doi.org/10.29304/jqcm.2020.12.1.674

Barnsley, M. F. (1993). Fractals Everywhere. Academic Press. https://doi.org/10.1016/C2013-0-10335-2

Dwivedi, P., & Mishra, A. (2022, December). Comparison of Fractal Image Compression Techniques. In 2022 IEEE International Conference on Current Development in Engineering and Technology (CCET) 1-6. IEEE. https://doi.org/10.1109/CCET56606.2022.10080013

Faragallah, O. S., Naeem, E. A., El-sayed, H. S., & Abd El-Samie, F. E. (2022). Efficient compression processing of optically DCT-based DRPE encrypted images. *Optical and Quantum Electronics*, 54(5), 273. https://doi.org/10.1007/s11082-022-03668-x

Fathi, A. and Abduljabbar, N.S., (2020). Survey on Fractal image compression. *Journal of Research in Science, Engineering and Technology*, 8(3), pp.5-8. https://doi.org/10.24200/jrset.vol8iss3pp5-8

Jacquin, A. E. (1992). Image coding based on a fractal theory of iterated contractive image transformations. *IEEE Transactions on Image Processing*, 1(1), 18-30. https://doi.org/10.1109/83.128028

Menassel, R., Gaba, I. and Titi, K., (2020). Introducing BAT inspired algorithm to improve fractal image compression. *International Journal of Computers and Applications*,42(7),pp.697704.https://doi.org/10.1080/1206212X.2019.1638631

Nithila, B. A., & Kousalyadevi, R. (2014). Fast fractal image compression based on Fisher's classification scheme. *International Conference on Electronics and Communication Systems (ICECS)* 1-4. https://doi.org/10.1109/ECS.2014.6892671

Revathy, K., & Jayamohan, M. (2012). Dynamic domain classification for fractal image compression. *International Journal of Computer Science & Information Technology*,4(2),95.http://dx.doi.org/10.5121/ijcsit.2012.4208

Saupe, D., & Hamzaoui, R. (1994). A review of the fractal image compression literature. ACM SIGGRAPH Computer Graphics,28(4),268276.https://doi.org/10.1145/193234.193246

Singh, I., & Bharany, S. (2020). Comparative studies of various techniques for image compression algorithm. *Acta Technica Corviniensis-Bulletin of Engineering,* 13(2), 127-133. https://acta.fih.upt.ro/pdf/2020-2/ACTA-2020-2-20.pdf

Wohlberg, B., & de Jager, G. (1999). A review of the fractal image coding literature. *IEEE Transactions on Image Processing*,8(12),17161729.https://doi.org/10.1109/83.806618

Y. Fisher. (1995). Fractal Image Compression: Theory and Application. *Springer-verlag,Berlin*,.https://doi.org/10.1007/978-1-4612-2472-3