MODIFIED FRACTAL IMAGE COMPRESSION (MFIC) AND DECOMPRESSION TECHNIQUE

by

Rahil Kazi, B.S.

A thesis submitted to the Graduate Council of
Texas State University in partial fulfillment
of the requirements for the degree of
Master of Science
with a Major in Electrical Engineering
May 2023

Committee Members:

Semih Aslan, Co-Chair

Damian Valles, Co-Chair

Harold Stern

COPYRIGHT

by

Rahil Kazi

2023

ACKNOWLEDGEMENTS

This research could not have been possible without the invaluable expertise and guidance of my thesis advisor, Dr. Semih Aslan. I am immensely grateful for his support and mentorship throughout this journey. I also want to thank the late Dr. William Stapleton, whose initial guidance laid a strong foundation for my research. His expertise and encouragement will always be remembered and appreciated. Furthermore, I sincerely thank Dr. Damian Valles and Dr. Harold Stern for joining my panel and reading and providing insightful feedback on my thesis. Their valuable input and expertise have significantly contributed to completing this work.

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF EQUATIONS

Page

LIST OF FIGURES

Page

LIST OF ABBREVIATIONS

Abbreviation	Description
DCT	Discrete Cosine Transform
DWT	Discrete Wavelet Transform
FIC	Fractal Image Compression
MFIC	Modified Fractal Image Compression
JPEG	Joint Photography Expert Group
HVS	Human Visual System
IFS	Iterated Function System
FIF	Fractal Image Format
RMS	Root Mean Square
PIFS	Partitioned Iterated Function Systems
QD	Quadtree Decomposition
QR	Quadtree Recomposition
HV	Horizontal-Vertical
SNR	Signal-to-noise ratio
PSNR	Peak signal-to-noise ratio
CR	Compression ratio

ABSTRACT

The field of image compression has been extensively researched for many years due to the increase in image resolution and quality. However, this improvement in image quality results in larger image sizes, making image transfer slower and storage more challenging. To overcome this issue, lossy image compression techniques are commonly used, but they often come with the trade-off of longer compression times. This study evaluates the performance of the Modified Fractal Image Compression (MFIC) method against traditional techniques such as JPEG and fractal image compression (FIC). Our results show that MFIC achieves faster decompression times and delivers higher PSNR values compared to both JPEG and traditional fractal compression. This highlights the potential of MFIC in optimizing the performance and user experience of image-based applications. The proposed MFIC approach in this paper offers fast image decoding using just one iteration. This approach allows for the precise calculation of the error contributed by each step of the partitioning optimization process.

Keywords— MFIC, FIC, JPEG, Image compression, HV Partitioning.

1. INTRODUCTION

With the increase in technological advancements, storing and using important data in memory storage have become common. However, reducing the size of this data remains a significant challenge. This is especially true for image data, which contains millions of pixels and can take up a lot of memory space. As a result, memory capacity needs to be high, which can be expensive. Various image compression methods have been developed to reduce the size of image files without compromising their quality, with the JPEG algorithm being the most used. Data compression has become crucial for data storage and transmission, particularly for databases that have high-definition images and videos. Fractal Image Compression is a promising technique that uses natural affine redundancy to achieve high compression ratios but has high computational demands.

The JPEG algorithm compresses images by filtering out high-frequency changes in color that the human eye cannot fully perceive. As a result, the compressed image may not have the same pixel values as the original image, but it will look the same to the human eye. This is because the compression algorithm is designed to be imperceptible to the Human Visual System (HVS), meaning that the compressed image is visually indistinguishable from the original image. This technique reduces the size of the image while maintaining its quality. JPEG2000, developed by the Joint Photographic Experts Group, is a more advanced image compression technique that can offer both lossy and lossless compression. It uses the Wavelet transform instead of DCT, which can result in better quality or better compression than the original JPEG algorithm, particularly when the quality rate is significantly reduced. Unlike the original JPEG, which is limited to handling a single RGB data channel, JPEG2000 can hold up to 256 channels of RGB data and display compressed images at different resolutions. However, the downside of the JPEG2000 technique is its high computational complexity, which limits its accessibility in applications like

web browsers. In contrast, JPEG is widely available in all browsers and devices. JPEG and JPEG2000 are compression methods based on transformations, which have limitations. However, another lossy compression technique, Fractal Image Compression (FIC), is based on fractal mathematics. This method is particularly suitable for natural and textured images as it uses an iterated function system (IFS) to generate patterns that resemble other portions of the image at varying scales. The algorithm generates the parameters of the IFS and converts them into "fractal codes" to recreate the encoded image [1]. FIC produces a resolution-independent image after compression, which can be scaled using a compressed system called "fractal scaling." At the same time, the encoding of the fractal IFS is computationally complex. FIC provides similar compression results as JPEG for compression ratios up to 50:1, but at higher compression ratios, FIC offers higher-quality compression. FIC is particularly efficient for images with higher complexity and color depth than grayscale images [1].

Fractal image compression (FIC) divides an image into smaller range and domain blocks. Range blocks are smaller than domain blocks, where domain blocks are twice the size of range blocks [1][2][3]. The mapping operation then performs intensity similarities on these blocks. While searching for the best domain block, the algorithm performs multiple search operations, which increases the encoding time and computational complexity. M. Barnsley and A. Jacquin discovered that raw images contain affine redundancy, meaning that large regions of an image can resemble finer parts of the same idea with a suitable Iterated Function System (IFS) [1][2][3][8]. FIC partitions the original image into sets of overlapping ranges and non-overlapping domains of any size or shape. The algorithm searches for the most suitable region for each domain and transforms it using an appropriate affine transformation. This generates a Fractal Image Format (FIF) file containing information on the domain and affine transformations. The pixel data of each

region is compressed into a set of transform matrices, taking up one byte corresponding to an integer between 0 and 255 levels of gray. FIC produces resolution-independent images that resemble the original at any resolution. However, this process is computationally expensive, particularly when searching for suitable domain ranges.

2. FRACTAL COMPRESSION AND ITERATED FUNCTION SYSTEMS

Fractal Image Compression (FIC) is a lossy compression technique that utilizes the mathematical concept of fractals to achieve high compression ratios for images [1]. The idea behind FIC is to identify and exploit the self-similar and affine redundancies present in natural and textured images. This is done by using an Iterated Function System (IFS) to generate patterns that resemble different parts of the image at varying scales [3][8]. The image is partitioned into range and domain blocks, where range blocks are smaller and domain blocks are twice the size of range blocks. The mapping operation then performs the intensity similarities between the blocks. The search operations are used to find the best domain block to match with each range block, which can be computationally expensive. The fractal algorithm generates the parameters of the IFS and converts them into data called “fractal codes,” which are used to recreate the encoded image [1][3]. The output image is resolution independent and can resemble the original at any resolution. Michael Barnsley and Alan Jacquin first developed the FIC method in their research, where they observed the rich affine redundancy present in raw images. By applying a suitable affine transformation, large regions of an image can be made to resemble finer parts of the same idea. The FIC process involves encoding the pixel data of each region into a small set of transform matrices, taking up a one-byte corresponding to an integer between 0 to 255 levels of gray. The final output is a Fractal Image Format (FIF) file containing information on the domain and affine transformations. FIC has been shown to provide similar compression results as JPEG at compression ratios up to 50:1, with high-quality compression at higher ratios. However, due to its high computational demands, FIC is not widely supported in applications like web browsers compared to JPEG, which is available on all devices [12].

The key concept in FIC is domain and range splitting, which involves dividing the image into non-overlapping domain blocks and larger range blocks. Each domain block is further divided into four smaller sub-blocks. The goal is to find the best matching domain block from the same image for each sub-block by comparing their pixel values using a similarity measure. The transformation required to map the domain block onto the sub-block is then recorded. Finally, the domain blocks and their corresponding transformations are used to reconstruct the compressed image.

Considering an example for a 128x128 image. The image is divided into non-overlapping 16x16 domain blocks, resulting in 64 domain blocks. Each domain block is further divided into four 8x8 sub-blocks. The best matching domain block from the same image is found for each sub-block. The similarity measure used for comparison could be Mean Squared Error (MSE) or Peak Signal-to-Noise Ratio (PSNR). Once the best matching domain block is found, the affine transformation needed to map the domain block onto the sub-block is recorded. The transformations can include scaling, rotation, translation, and reflection. After encoding the domain and range blocks, the compressed image can be generated by applying the affine transformations to the domain blocks and combining them to form the range blocks. The resulting image is resolution-independent, meaning it can be displayed at any resolution without losing quality.

In summary, FIC uses iterated function systems and domain and range splitting to compress images by finding self-similarities within the image. By dividing the image into smaller blocks and encoding the transformations needed to map these blocks onto each other, FIC can achieve high compression ratios with minimal loss of quality.

3. PARTITIONED ITERATED FUNCTION SYSTEMS (PIFS)

Jacquin is known for developing the first automated method of creating an IFS from a regular image [1]. In this technique, the image is divided into non-overlapping regions called ranges and overlapping regions called domains. The ranges and domains together cover the entire image space, with the domains allowed to overlap. Small blocks represent the ranges, and the domains are represented by large blocks, as depicted in Figure 1. Encoding involves finding the best affine transformations by searching partial or global domain block spaces. However, since affine transformations are used, the domain set and range set partitioning schemes must be the same geometric shapes, usually rectangles or squares. The domains can overlap to reduce artifacts between blocks during the decoding process. The mapping between the domain and range blocks is illustrated in Figure 2, where each range block needs a good domain block for mapping. PIFS is a more advanced version of IFS, allowing for more image partitioning flexibility. This method provides a higher level of compression and better-quality images at higher compression ratios. PIFS is used in many image compression applications, including satellite and medical imaging.

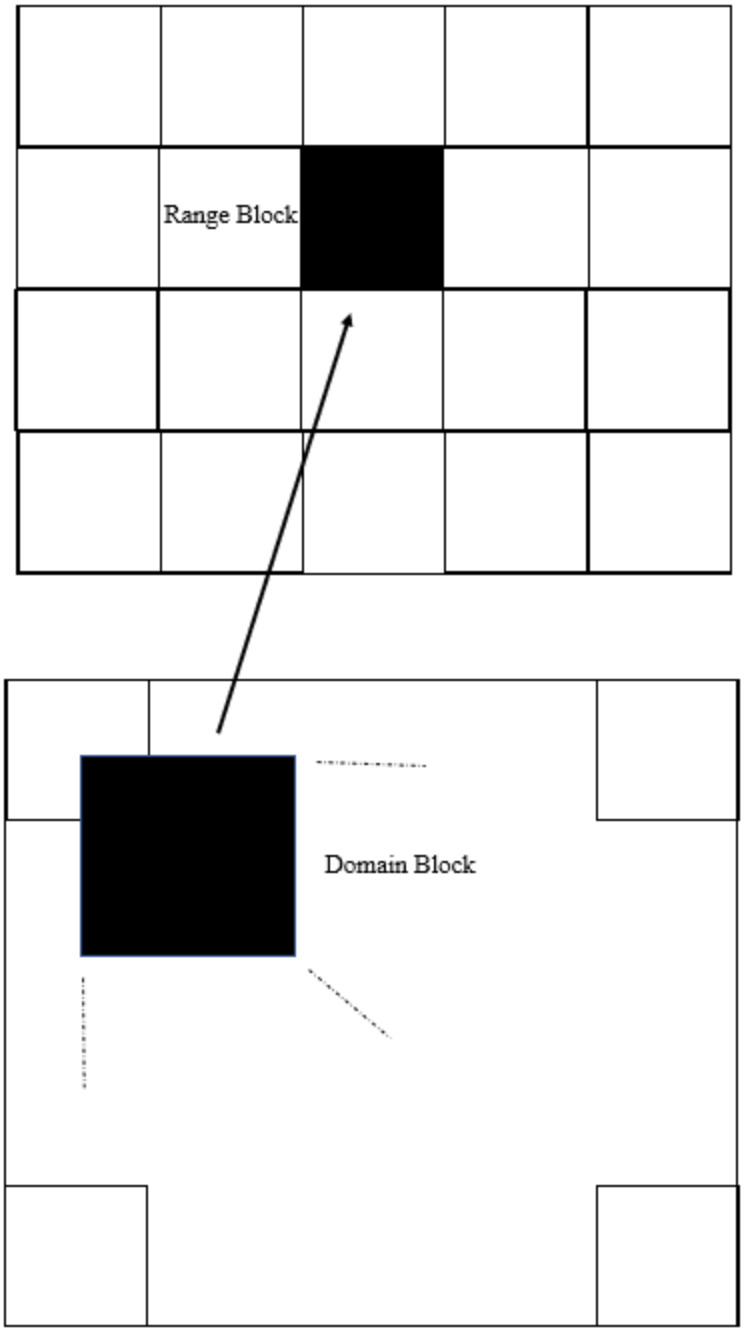

Figure 1. PIFS Representation of Range and Domain Blocks

A PIFS is a generalization of an IFS and attempts to simplify the IFS computations by partitioning the whole space into subspaces. PIFS is recognized as a significant improvement

over IFS. It reduces the amount of search time both theoretically and technically. Furthermore, it provides some possible aspects of improving fractal encodings, like using different partitioning schemes and mapping methods. A shortcoming of this approach is that the matching process is time-consuming and computationally intensive for large domain spaces.

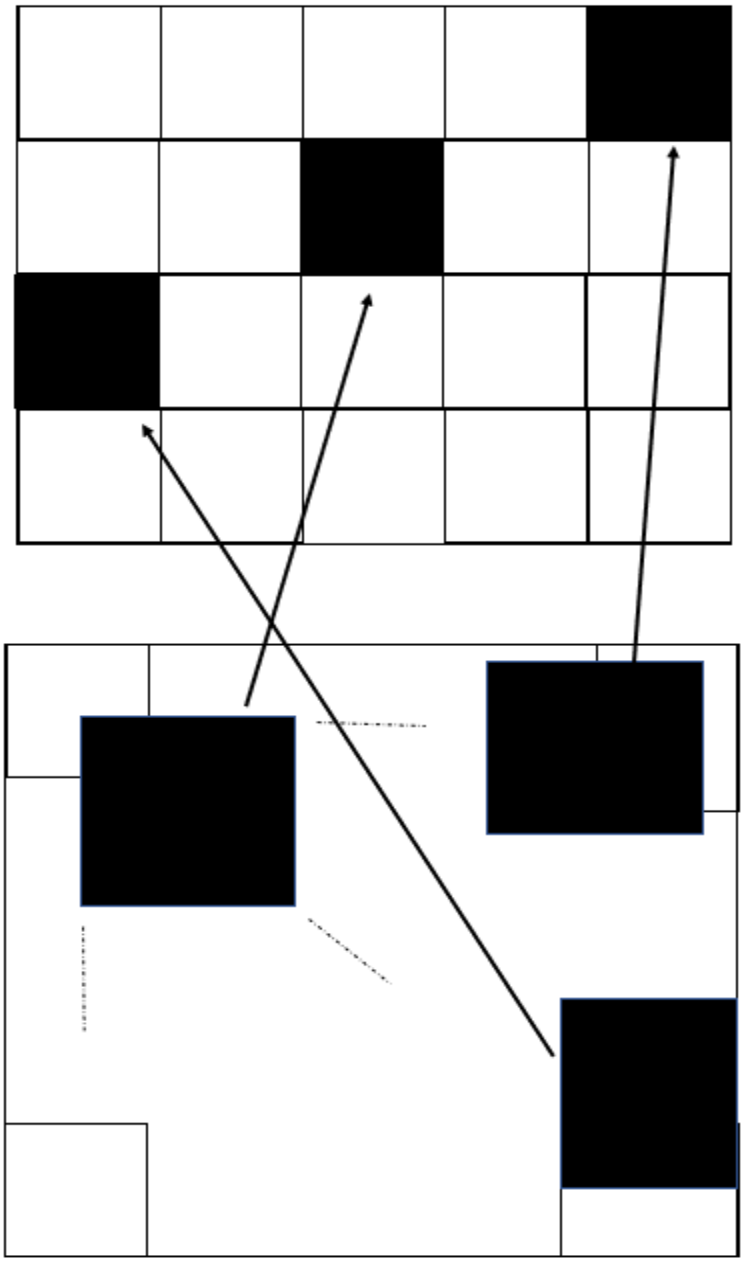

Figure 2. Domain to Range Mapping

4. QUADTREE PARTITIONING

Quadtree partitioning is used in Fractal Image Compression (FIC) to divide an image into smaller sub-images. It is a recursive algorithm that partitions an image into smaller rectangular blocks, known as quadtrees [1][4][5][7].

The algorithm starts by dividing the original image into four equal-sized blocks, forming the first level of the quadtree. Each block is then analyzed to determine if it can be compressed using a fractal transformation. If it cannot be compressed satisfactorily, the block is further subdivided into four smaller blocks at the next level of the quadtree. This process is repeated until all blocks can be compressed using a fractal transformation or a predetermined maximum subdivision level is reached. The resulting quadtree represents the decomposition of the image into smaller sub-images that can be compressed using the fractal transformation.

For example, consider a 128x128 image that needs to be compressed using the quadtree partitioning technique. The algorithm starts by dividing the image into four equal-sized blocks, each with dimensions of 64x64 pixels. Each block is then analyzed to determine if it can be compressed using a fractal transformation. If any block cannot be compressed satisfactorily, it is subdivided into four smaller blocks of 32x32 pixels. This process continues recursively until all blocks can be compressed using the fractal transformation or a predetermined maximum subdivision level is reached. The resulting quadtree would consist of a series of nodes, each representing a block in the image, with the root node representing the entire image. The leaf nodes of the quadtree represent the smallest blocks that can be compressed using fractal transformation. The compressed image can be reconstructed by applying the fractal transformation to each leaf node.

5. CONVENTIONAL HORIZONTAL-VERTICAL (HV) PARTITION SCHEME

The Horizontal-Vertical (HV) partition scheme is a type of partitioning used in fractal image compression [1]. In this scheme, the image is partitioned into two sets of blocks: horizontal and vertical blocks. The horizontal blocks are formed by dividing the image into rows of equal height, while the vertical blocks are formed by dividing the image into columns of equal width. The HV partitioning scheme is like the quadtree partitioning scheme, which recursively divides the image into quadrants. However, HV partitioning is a simpler and more computationally efficient method of partitioning [3][9][10][11][12]. Once the image is partitioned into horizontal and vertical blocks, fractal encoding involves finding the best domain block for each range block. This is done by matching the intensity values of the range block with those of the domain blocks. The matching process involves searching the domain blocks to find the one that best matches the intensity values of the range block. HV partitioning can result in good compression ratios and image quality, especially for images with prominent horizontal or vertical features. However, it may need to perform better for images with complex and irregular features. Other partitioning schemes, such as quadtree partitioning, may be more suitable for such images.

For example, consider a 128x128 grayscale image. Begin by dividing it into four 64x64 sub-blocks. The top-left sub-block can be further divided into four 32x32 sub-blocks. For each sub-block, compare them with all possible domain blocks within a search range to find the best match. Suppose the top-left sub-block matches best with a domain block that is split vertically. Split the domain block into two equal-sized sub-blocks and compare them with the corresponding range blocks. Continue this process recursively until we reach a stopping criterion, such as a minimum block size.

In the HV scheme, the resulting domain blocks are either rectangular or square, unlike in the PIFS scheme, where they can be of any shape. This restriction helps reduce the encoding process's computational complexity [12]. Once the encoding is complete, the resulting compressed file contains the parameters of the affine transformations for each domain block and the corresponding range block pixel values. The decoder uses these parameters to recreate the original image.

6. LITERATURE REVIEW

This section reviews the relevant study and prior work on implementing Fractal Image Compression. The studies show how to design image processing algorithms and their practical implementation.

6.1 Fractal Image Compression - Theory and Application

In [1], Fisher discusses the theory and practical implementation of Fractal image compression. It covers various topics related to fractal image compression, including the basics of fractal geometry, FIC algorithms, image quality metrics, and applications of fractal image compression. The author investigates the use of the Horizontal-Vertical (HV) partitioning scheme for fractal image compression. The HV partitioning scheme involves dividing an image into horizontal and vertical blocks of equal size. The book discusses how this scheme can be used to improve the efficiency of the fractal compression process by reducing the number of candidate domain blocks that need to be searched during the encoding process. It also presents experimental results showing this approach's effectiveness on various test images. It also includes case studies and examples to illustrate the concepts such as iterated function systems, quadtree partitioning, domain, and range block matching, and PIFS. The author also discusses various applications of fractal image compression, such as image and video compression, texture synthesis, and image retrieval.

6.2 Methods and Apparatus for Image Compression by Iterated Function Systems

Barnsley and Sloan in [2] present an image compression method and apparatus using Iterated Function Systems (IFS). The invention involves partitioning an image into non-overlapping ranges and overlapping domains. The compression is achieved by approximating the domains using IFS transformations applied to the ranges. The patent discusses the use of affine transformations to achieve this compression and the use of domain blocks of different sizes and shapes to improve compression performance. The paper also describes the generation of fractal codes for each domain and the decoding process to reconstruct the original image from the compressed data. Overall, the paper describes an innovative approach to image compression that uses fractal geometry to achieve high compression ratios.

6.3 Fractal Encoding with HV Partitions

Fisher and Menlove [3] investigate using the HV partitioning scheme in fractal image compression. It explains the concept of HV partitioning, a technique that partitions the image into horizontal and vertical stripes. Then it applies the fractal compression algorithm to each stripe separately. The paper describes how the HV partitioning scheme can improve the compression ratio and reduce the encoding time compared to other partitioning schemes. The paper explains the HV partitioning scheme and how it works with the fractal encoding algorithm. It presents experimental results that demonstrate the effectiveness of HV partitioning in reducing the encoding time and improving the compression ratio. Overall, the paper is a useful contribution to fractal image compression, clearly explaining the HV partitioning scheme and its benefits. It presents experimental results that support the technique’s effectiveness, making it a valuable resource for researchers and practitioners in image compression.

6.4 Fractal Image Compression using Quadtree Recomposition.

Jackson et al., in [4], investigated a fractal image compression algorithm employing a new quadtree compression technique, the Quadtree Recomposition (QR) approach. In this approach, quadtree subblocks of an image are iteratively recombined into larger blocks for fractal coding. This approach exhibits superior runtime performance for complex ideas compared to a classical quadtree decomposition (QD) approach while maintaining high fidelity for reconstructed images. In the QR algorithm, processing begins from the lowest level in the decomposition. That is, the image is initially partitioned into range blocks that are assumed to be uncovered. Quantitative results include an evaluation of attained compression ratios, runtime performance, and signal-to-noise ratios (SNR) for reconstructed images. The range-domain comparisons proceed in the same manner as the quadtree decomposition algorithm. The authors suggested that the QR algorithm proves highly efficient and provides consistently superior compression time performance compared to the QD algorithm. Image block classification schemes should be employed to reduce the compression time further. The author also indicated that additional research is required to incorporate the QR technique into content-sensitive image partitioning techniques such as HV partitioning.

6.5 Fractal Image Compression with Quadtree Partitioning and a New Fast Classification Strategy

Nandi, Santra, Mandal, and Nandi [5] proposed a new fast classification strategy for Fractal image compression with a quadtree partitioning scheme. The proposed method claims to reduce the mean square error (MSE) computations while encoding images. The method involves dividing the image into blocks using quadtree partitioning and then using a new classification strategy to determine the best-matching domain blocks for each range block. The authors claim that their method achieves faster encoding times compared to existing methods while maintaining good compression performance. The paper is well-structured and clearly explains the proposed method. The experimental results presented in the paper demonstrate that the proposed method achieves good compression performance and significantly reduces the encoding time compared to existing methods. However, the paper needs a detailed comparison of the proposed method with state-of-the-art methods in terms of compression performance, which would have strengthened the claims made by the authors. Overall, the paper presents a promising new method for fractal image compression that could potentially have practical applications. The proposed technique significantly reduced the image encoding time by using a proposed fast classification strategy without degrading the compression ratio and PSNR of the decoded image.

6.6 Non-search Fractal Image Compression Algorithm Research

Li and Xia [6] have proposed a non-search Fractal image compression technology to solve the long search operation in Fractal image compression due to domain search. As fractal image compression techniques use the self-similarity of an image, it requires domain search, a time-absorbing and computer-intensive operation that results in low coding speed. To overcome this drawback, the authors proposed a new search-less algorithm; the results show that it is better than the previous one. The non-search algorithm is the most distinct feature of this paper; the results prove that assigning a single domain block to each range for data matching reduces the operation time and improves data compression and the quality of the compressed image. The non-search algorithm does not need each range to domain matching for optimal compression; One domain block is assigned as the best match of ranges. This results in better optimization and reduces the operation time heavily. Since the number of fractal codes reduces, the compression ratio also improves.

6.7 A three-component hybrid image compression method

Stapleton and McNees [7] present a hybrid image compression algorithm combining the JPEG, JPEG 2000, and search-less FIC techniques. The results show the comparison between each method regarding PSNR and compression rate. The three-component algorithm can provide a high compression rate and high-quality reconstructed images. The fractal image compression implemented for hybrid algorithms is a searchless algorithm proven to be faster and provides high compression. Also capable of performing deep Quadtree spanning the most significant square range blocks of 32x32 pixels and smallest of 1x1 is considered. Both JPEG and JPEG2000 algorithm partitions an image into 8x8 blocks before quantization. After compressing images for

each method separately, the quality of each is compared to the best performing for each block. After operating on several test images, the results prove that the hybrid compression method surpasses the capabilities of individual compression methods alone with lesser PSNR and a high compression rate. Also, using the searchless fractal algorithm significantly reduces the execution time; hence, speed is not a limiting factor.

6.8 Fractal image coding based on a theory of iterated contractive image transformations.

Jacquin in [8] presents the concept of fractal image coding based on the theory of iterated contractive image transformations. The author describes the theoretical framework and implementation details of the proposed method. The paper provides a clear explanation of the mathematical background and practical considerations involved in fractal image coding.

One of the paper's strengths is its detailed discussion of the properties of fractal transforms and how they can be used to compress images efficiently. The author presents a step-by-step algorithm for fractal image coding that includes the selection of appropriate domain blocks and the computation of contractive transforms for each block. The paper also includes several experimental results that demonstrate the effectiveness of the proposed method. Overall, this paper is a seminal contribution to fractal image coding and remains an important reference for researchers in this area. The clear presentation and comprehensive discussion make it accessible even to readers with limited background in the subject.

6.9 Modified Horizontal Vertical partition scheme for fractal image compression.

Nikolay et al., in [9], present a modification to the HV partitioning scheme for fractal image compression. The authors propose to use a modified HV partitioning scheme where the image is first divided into vertical strips. Then each strip is subdivided horizontally using the standard HV

partitioning scheme. The resulting encoding process is more efficient than the standard HV scheme, and it achieves better compression ratios for a given level of distortion. The authors also propose a modification to the range block matching algorithm that improves the speed of the encoding process. The paper presents a well-written and clearly explained modification to the standard HV partitioning scheme for fractal image compression. The authors provide a thorough analysis of the proposed modification and demonstrate its effectiveness through experiments on a variety of test images. However, it should be noted that the paper was published in 2002 and may not reflect the state-of-the-art of FIC.

6.10 Fractal Image Compression Using Fast Context Independent HV Partitioning Scheme

Nandi and Mandal in [10] propose a fast context-independent HV (Horizontal-Vertical) partitioning scheme for fractal image compression. The authors claim that their scheme improves the compression ratio and computational time compared to existing methods. The proposed method involves dividing the image into HV partitions and determining the best range blocks for each domain block using an efficient matching algorithm. The paper presents experimental results that demonstrate the effectiveness of the proposed scheme on standard test images. Overall, the paper clearly and concisely explains the proposed HV partitioning scheme and the matching algorithm. The experimental results are also presented in a clear and organized manner. However, the paper needs a detailed comparison with other state-of-the-art methods, which could have provided a better understanding of the effectiveness of the proposed scheme. Additionally, the paper could have benefited from a more in-depth discussion of the proposed method’s limitations and potential future directions.

6.11 Fractal image compression using a fast affine transform and hierarchical classification scheme.

Nandi in [11] proposes a fast affine transform and hierarchical classification scheme for fractal image compression. The author addresses the high computational complexity restricting the practical use of fractal image compression. The paper presents a novel method of encoding fractal image compression using a fast affine transform and hierarchical classification scheme. The fast affinity of image blocks uses relationships among neighboring pixels of the transformed image block, which reduces the number of multiplication and addition operations. This sorting is applied in fractal coding with quadtree and horizontal and vertical partitioning schemes to reduce the compression time. The study tested the proposed method on two partitioning schemes, quadtree (FICQP-FAT) and HV (FICHV-FAT), and found that the FICQP-FAT technique significantly reduced compression time while the FICHV-FAT offered better image quality than most other techniques, except for FICHV. However, there is a need for further investigation to find a technique that balances both schemes for an acceptable compression time and well-reconstructed image quality.

6.12 Fractal Image Compression and Its Techniques: A Review

Joshi, Agarwal, and Gupta in [12] address an investigation of the fractal coding design with iterated function system (IFS) utilizing the Fast encoding, ICA (Independent Component Analysis), and DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithms. The investigation begins with mentioning the fundamental problems and then moves to study various methods of fractal designs. The findings suggest that DBSCAN can rapidly pack and interpret the shading pictures by considering their RGB values. The ICA algorithm decreases

the MSE computation complexity; the fast-encoding method compares the HV scheme with the Quadtree scheme, where HV can be less complex and reduces the computation complexity with acceptable quality. The compression ratio achieved by fractal image compression using DBSCAN is comparable to that of other fractal compression techniques, and in some cases, it can be even higher. Additionally, using the DBSCAN clustering algorithm allows for a more efficient compression process, eliminating the need for an exhaustive search for the best affine transformations for each block. However, fractal image compression using DBSCAN requires much computation power, as the DBSCAN algorithm has a high computational complexity. Also, the quality of the compressed image is highly dependent on the choice of epsilon and the size of the blocks used for partitioning. The ICA algorithm has improved the compression ratio and image quality compared to traditional FIC techniques. However, there are still challenges to be addressed with the use of ICA in FIC, such as the choice of appropriate parameters for the algorithm and the potential loss of information during the compression process. Nonetheless, the use of ICA in FIC shows promise for improving the efficiency and effectiveness of the technique in compressing digital images. Overall, this paper is well-structured and well-written and thoroughly introduces Fractal Image Compression. The authors have done an excellent job summarizing FIC's key concepts, techniques, challenges, and applications and provided a good foundation for further study in this area.

6.13 Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations

Jacquin [13] proposes a method for compressing digital images using fractal theory and iterated contractive image transformations. The approach involves dividing an image into small blocks and encoding each block using mathematical formulas that capture its self-similarity. Experimental results demonstrate that the proposed method achieves high compression ratios without significant loss of image quality, with compression ratios of up to 1:10. Additionally, Jacquin compares his method with other compression techniques, such as DCT and wavelet transform-based methods, showing that his approach outperforms them in terms of both compression ratio and image quality. Overall, the paper provides a new perspective on image compression and has influenced subsequent research in the field. It has also found practical use in various commercial applications.

6.14 A Fast-Partitioning Strategy: Its Application to Fractal Image Coding

Nandi et al. in [14] propose a fast and efficient partitioning strategy for fractal image coding. They argue that conventional partitioning methods for fractal image coding are time-consuming and computationally intensive. They suggest a more efficient strategy to maintain high compression performance while reducing computational complexity. Their partitioning method involves hierarchical quadtree decomposition to break the image into smaller blocks. They also introduce a new similarity measure between image blocks, called the Discrete Cosine Transform (DCT) residual similarity measure. Using this measure to group blocks together, they can form larger partitions to generate fractal codes with high compression ratios. The authors present experimental results that demonstrate the effectiveness of their proposed partitioning strategy.

They show that their approach achieves similar compression ratios to traditional methods while being much faster and more computationally efficient.

7. INTRODUCTION TO MODIFIED FRACTAL IMAGE COMPRESSION (MFIC)

FIC has been an area of in-depth research during recent decades. It is confirmed that several severe shortcomings are inherent to FIC. One of the strictest drawbacks of FIC is that significant computations are required while encoding the image. In this sense, the time needed to encode an image for FIC is usually larger than for other compression techniques. Another drawback is the relatively substantial computational expenses spent on fractal image decompression and the large number of iterations required for decompression. Besides, with decompression at the period of partition scheme optimization at the encoding stage, it is feasible to know the error contributed due to the non-accurate mapping of the range blocks to the domain blocks into the collage error of image decompression. The presence of cycles in mapping determines the latter two drawbacks. For range block decoding, finding the range block values placed at the region occupied by the corresponding domain is vital. In turn, for their calculation, it is necessary to calculate the values for range blocks located in places occupied by the corresponding domain block, etc. Figure 3 shows an example of the cycle where the range blocks R1, R2, and R3 correspond to the domain blocks D1, D2, and D3, respectively. Then for getting R1, R2 is needed; for obtaining R2, R3 is required; and, for getting R3, it is crucial to know the values of R1. And this is a relatively simple case.

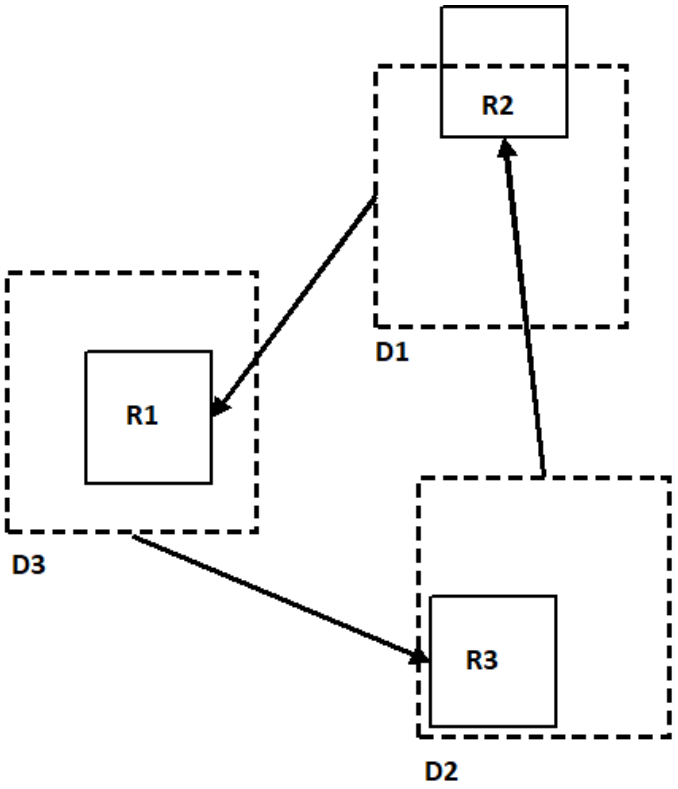

Figure 3. An example of the cycle in block mappings

The actual image compression situations in practice are much more complex than the example in Figure 3. Usually, for calculating the values of one range block, it is vital to use the chain of the pixels of many image range blocks, including the range block itself. Therefore, several iterations are required in image decompression to get reasonable accuracy. This procedure could be more computationally efficient, making calculating the change of collage error for each block variant dividing for each step of partition scheme optimization impractical.

The proposed FIC modifications consist of organizing mapping of the range blocks to the domain blocks that do not assume the existence of any cycles in mapping and chains inside the regions. This allows, at the phase of partition scheme optimization, to accurately know the error *(E)* contributed by each range block into decoding error, where.

$$E = \sum_{i=1}^{N} (Bi - DiK - C\text{^}2),$$

Equation 1. Error Contributed by Range Blocks.

Bi is the *i*-th range block, *Di* represents the best matching domain block, N is the number of ranges, K represents the contrast coefficient, and C is the intensity. This also allows the decoding of the compressed image in one iteration without worsening the decoding accuracy.

8. PRINCIPLES OF MODIFIED FRACTAL IMAGE COMPRESSION (MFIC)

This paper presents a modified fractal image compression (MFIC) approach. To circumvent the rotation in range blocks to domain block mapping, an image is divided into ranges like a "waterfall." In coding every waterfall region, as the domain blocks, only regions already coded to the given moment are used. Initially, a 4x4 pixel placed in one of four workable image corners is selected. Figure 4 provides an example of coding starting from the image's upper left corner.

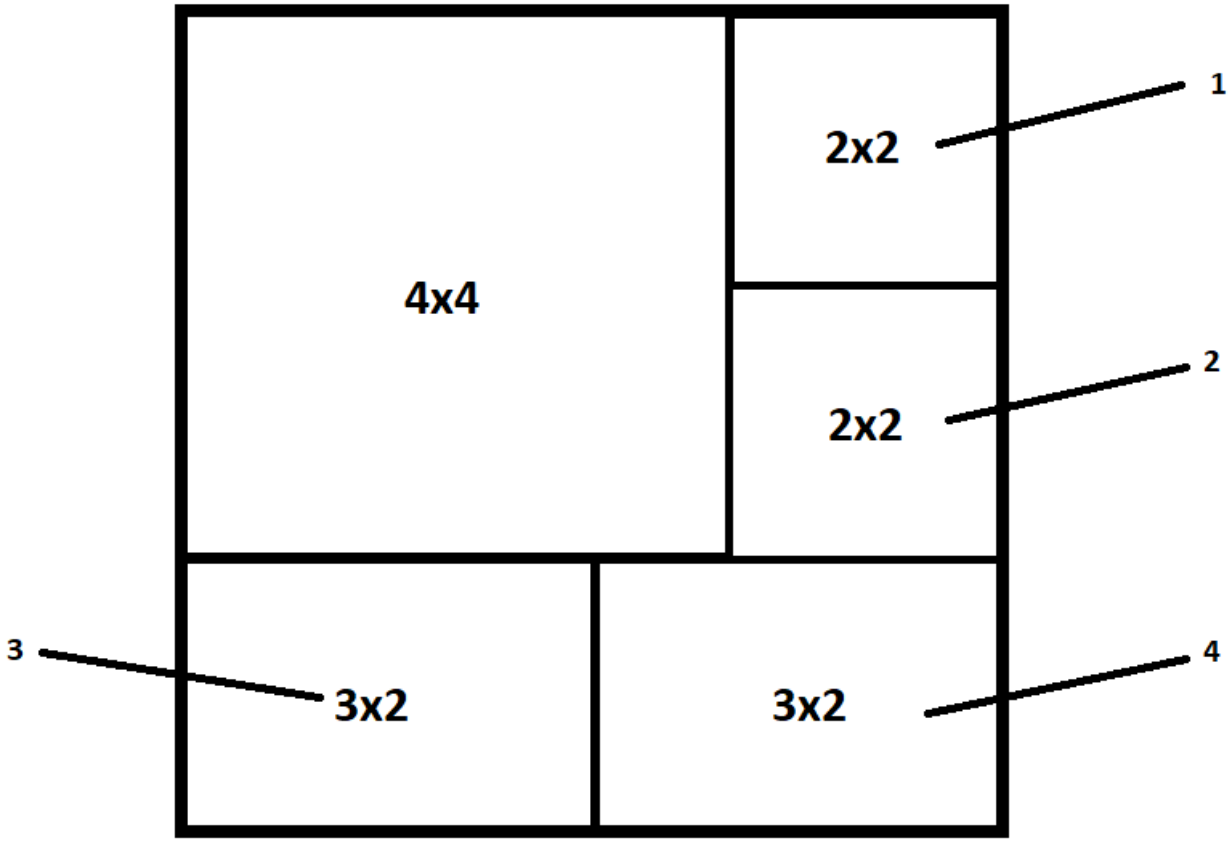

Figure 4. Block of size 4x4 pixels and the method of forming the first and the following cascade regions.

This 4x4 block is used as a domain block for comparing the ranges 1 and 2 that have the size 2x2 pixels attached to the initial block from the right side. After coding ranges 1 and 2, it is

possible to use three ranges as the domain block set – the initial one and the ranges 1 and 2 (the total size is then 6x4 pixels). That is why the following ranges, 3 and 4 are to be coded (the following regions are permanently attached to the longer side of the established rectangular of the domain block field) and have the size of 3x2 pixels. The process is continued until the entire image is divided into ranges. The areas of the waterfall range quickly grow. Figure 5 shows the order of compression and the speed of range area growing for an image with an entire dimension of 512x512 pixels. The numbers inside the cells show the order of their encoding. This region partition scheme is applied to optimize each waterfall region compression. An essential advantage of this approach is an elevated speed of image decompression. That is why to avoid decreasing decompression speed; it seems logical to use the partitioning schemes with rectangular shapes of blocks (quadtree, horizontal-vertical and similar ones) [2]. The computational cost for each pixel search inside the range block at the decoding stage becomes minimal.

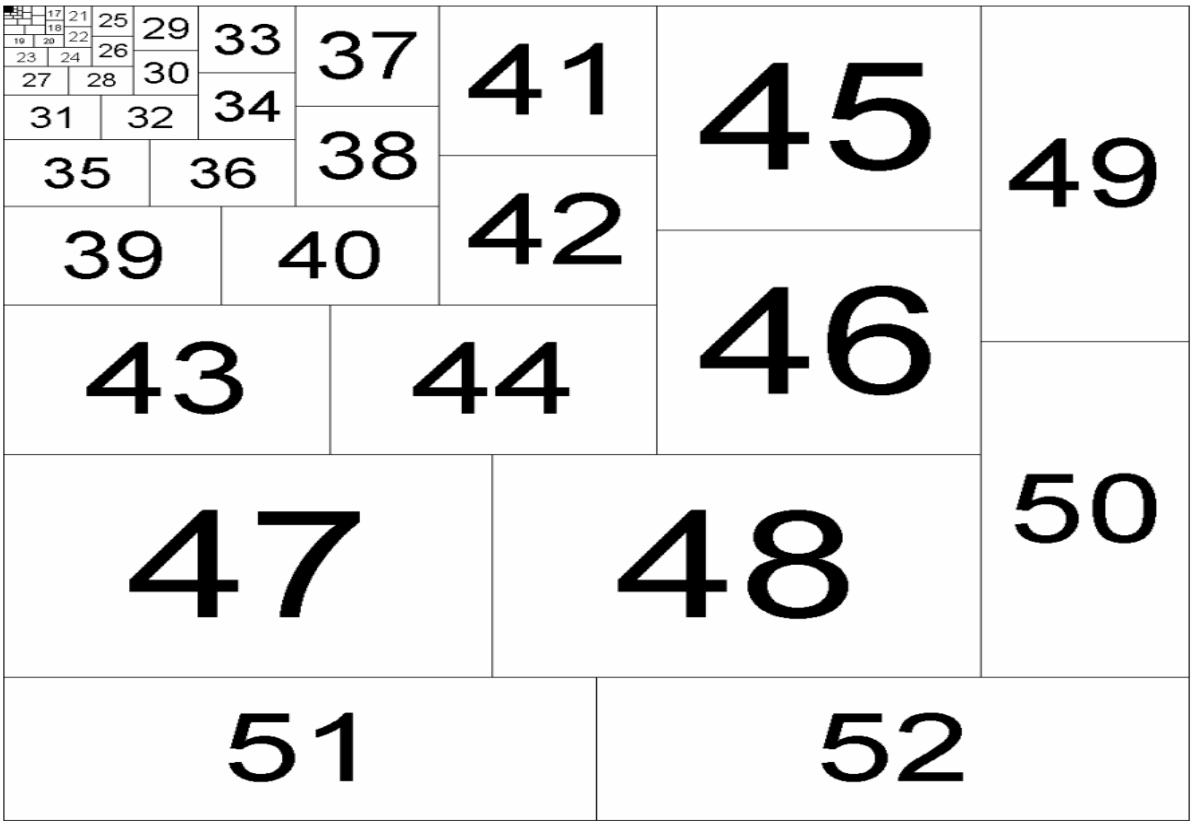

Figure 5. An order of compression of cascade region for an image of 512x512 pixels.

Different approaches to memory resource distribution between waterfall ranges are possible. The most straightforward implementation is an indirect distribution considering the limit of losses in image compression. In this case, the standard acceptable level of losses for all regions is preset. The partitioning scheme is optimized sequentially according to the order of ranges number in the waterfall. The partitioning scheme is then divided into details until the losses for a given range become smaller than the present values. Thus, the partition scheme for the more complicated areas must contain more blocks than those for simpler regions.

At the stage of image decoding, the ranges are decoded in the same order as executed at the coding stage. That is why the values for regions used as domain blocks are now known for the given region decoding.

9. MODIFICATIONS WITH EQUAL SIZES OF DOMAIN AND RANGE BLOCKS

As mentioned, the MFIC approach to image coding avoids the necessity of cycle presence in the domain block-to-range block mapping scheme. Therefore, the domain block scaling required for the traditional method to provide the convergence of image iterative decoding is now unnecessary. It becomes possible to use range and domain blocks having equal size (see Figures 6 and 7). In the case of waterfall compression, this opportunity allows having a smaller number of waterfall regions and, therefore, improves the quality of each image region coding and the whole image coding. This also increases the speed of image decompression due to the absence of the domain block scaling operation and decreases the computational time. For FIC and its modification with similar domain and range block sizes, the field of domain block search for a range block is made smaller. Naturally, this will lead to decreasing coding efficiency. However, such a damaging effect is partly compensated by reducing the bits required for coding the found domain block coordinates. Besides, simultaneously some increase in coding efficiency is observed due to more accurate calculation.

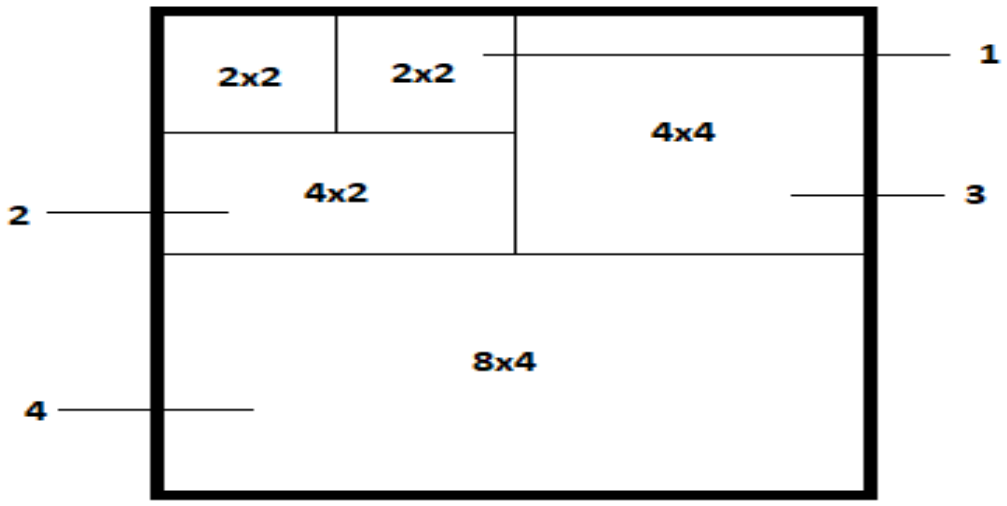

Figure 6. Method of forming the first and the next region of Cascade.

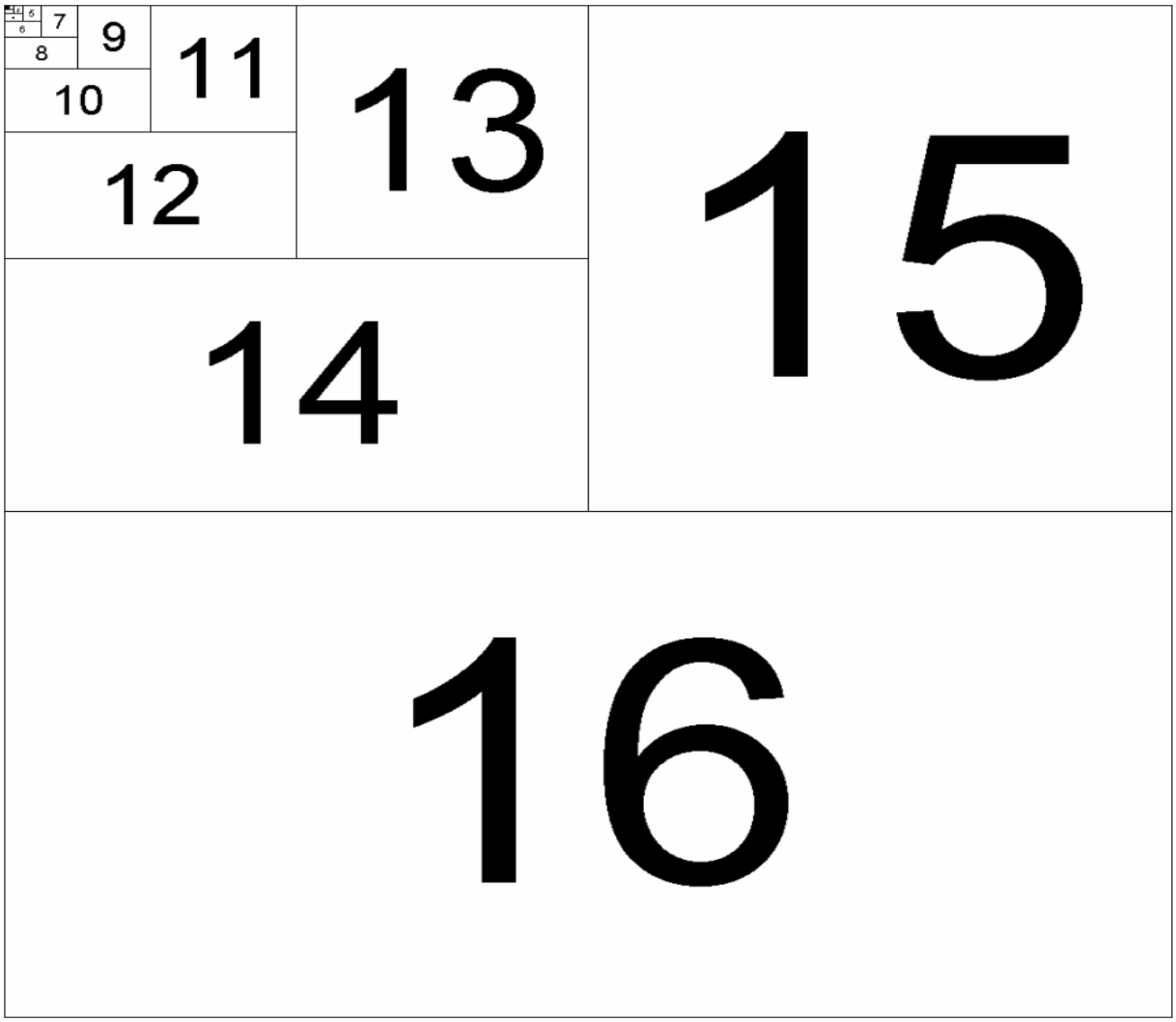

Figure 7. An order of region compression for equal sizes of range and domain blocks in a 512x512 pixels image.

10. PARTITIONING SCHEME

For the MFIC partition scheme, a modified horizontal-vertical scheme is used [9]. According to this partition scheme, each range block could be divided into two equal blocks horizontally or vertically. For each range block, the following information will be stored: the coordinates of the domain block (up to 15 bits) depending upon the size of the codebook, the contrast coefficient (5 bits), and the intensity (7 bits). Partition scheme data is compressed by arithmetical coding (2-2.5 bits/block) and stored in the encoded image.

11. EXPERIMENTAL RESULTS

Figures. 8 (a) and (b) show two images used to test the MFIC algorithm. These will be referred to as Kashmir and Colorado. These images were chosen for their combination of complexities and textures containing smooth and contoured areas. Two measurements were used to compare the MFIC, FIC, and JPEG results. These measurements are the Compression Ratio (CR) and Peak Signal-To-Noise Ratio (PSNR). CR refers to the ratio of the size of the original uncompressed image to the size of the compressed image. It is used to measure how much the size of an image has been reduced through compression.

$$Compression\ Ratio\ =\ Original\ Image\ Size\ /\ Compressed\ Image\ Size$$

Equation 2. Compression Ratio

PSNR is a commonly used metric to evaluate the quality of compressed images. It measures the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. In image compression, PSNR compares the original uncompressed image with the compressed image and computes a numerical value that indicates how closely the compressed image matches the original.

$$PSNR\ =\ 20\ *\ log10(MAXp)\ -\ 10\ *\ log10(MSE)$$

Equation 3. PSNR

Where *MAXp* is the maximum possible pixel value of the image. For an 8-bit image, the *MAXp* value is 255, and *MSE* (Mean Squared Error) is the average squared difference between the original uncompressed and compressed images. The PSNR value is expressed in decibels (dB) and

ranges from 0 to 60 dB. Higher PSNR values indicate better image quality, while lower values indicate lower quality.

Figure 8. The Test Images (a) Kashmir (b) Colorado

Table 1. PSNR Comparison

Image	Resolution	MFIC PSNR	FIC PSNR	Qs	JPEG PSNR
Kashmir	128x128	32.36	31.14	90	20.67
	256x256	35.23	34.65	90	27.8
	512x512	36.35	35.45	90	30.02
Colorado	128x128	35.89	34.46	85	26.27
	256x256	38.35	37.89	85	26.49
	512x512	32.64	31.32	85	26.93

Table 1 shows the resolutions of Kashmir and Colorado, which are 128x128, 256x256, and 512x512. MFIC was able to maintain the PSNR better than FIC and JPEG. Hence, the quality of decoded images using MFIC will be better than for FIC and JPEG. MFIC performs better than the FIC and JPEG for both images at all resolutions tested.

11.1 Kashmir 512x512 Results

Figure 9. Kashmir Compressed Images (a) OG (b) MFIC (c) FIC (d) JPEG

Figure. 9 shows examples of the Kashmir image of 512x512 resolution compressed by each method. Figure. 9 (a) of Figure. 9 is the original image. Figure. 9 (b) is the image compressed by MFIC with MSE = 15.04, CR= 5.17, and PSNR = 36.35. Figure. 9 (c) is the image compressed by FIC with CR = 3.64 and PSNR = 35.45. Figure. 9 (d) is the image compressed by JPEG with

Quality = 85, CR = 3.18, and PSNR = 30.02. The results of MFIC are better than the other methods, and the compressed image quality looks visibly better than the other methods.

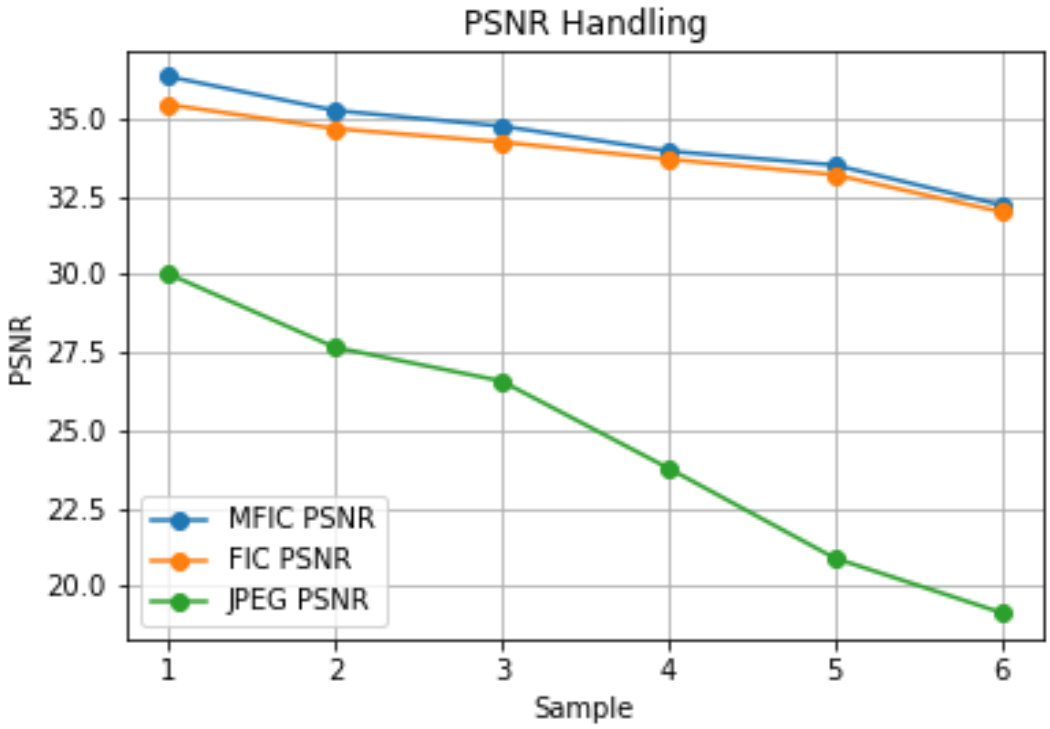

Figure 10. PSNR Handling for Kashmir 512x512.

Figure. 10 shows the PSNR comparison for the Kashmir 512x512 image for three image compression methods. These methods include each of the MFIC, FIC, and JPEG methods individually. Figure. 10 shows that the MFIC method maintains the PSNR better than FIC and JPEG methods. However, with the increase in CR, the PSNR levels for JPEG fall much more rapidly than the other methods. Figure. 10 indicates that PSNR is close for MFIC and FIC; both methods compress different image blocks well.

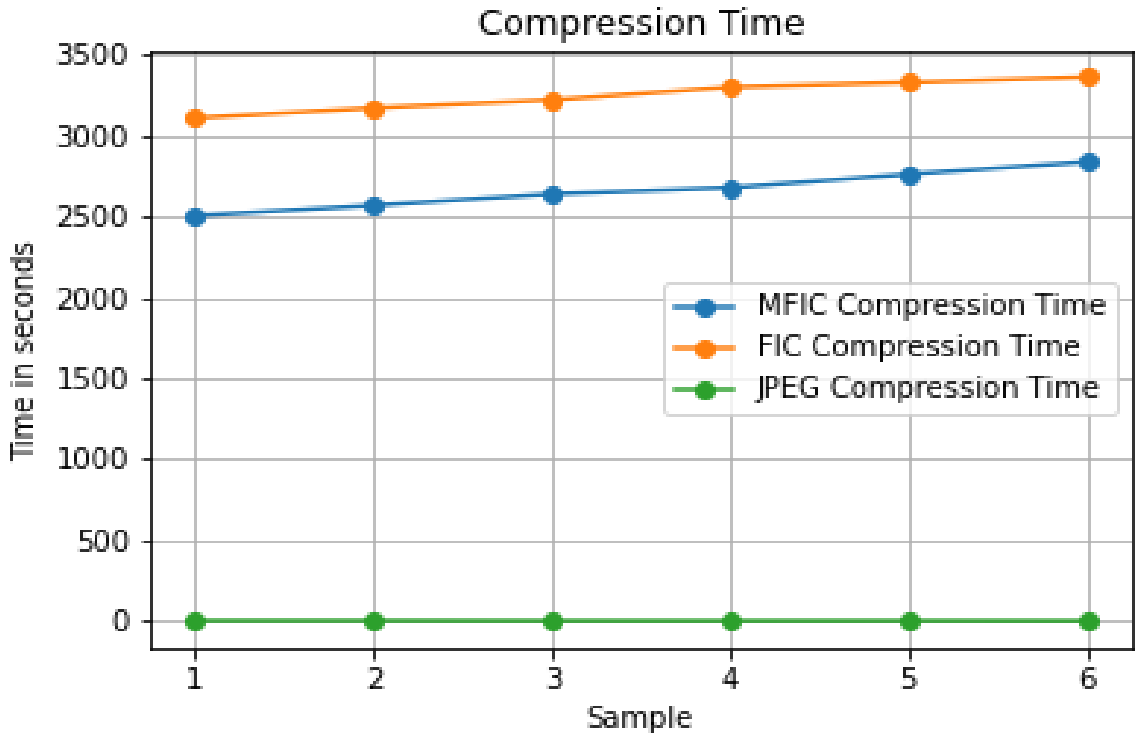

Figure 11. Compression Time for Kashmir 512x512

Figure. 11 shows the Compression Time in seconds comparison of the Kashmir image of 512x512 resolution compressed by each method. JPEG is the fastest image compression which took 4.91-3.76 seconds. MFIC took 2503-2835 seconds, which is faster than FIC, which took 3106-3360 seconds, which makes MFIC 15% faster than FIC to perform compression. While both MFIC and FIC maintained the PSNR well, it can be said that MFIC compresses different image blocks faster than the FIC method.

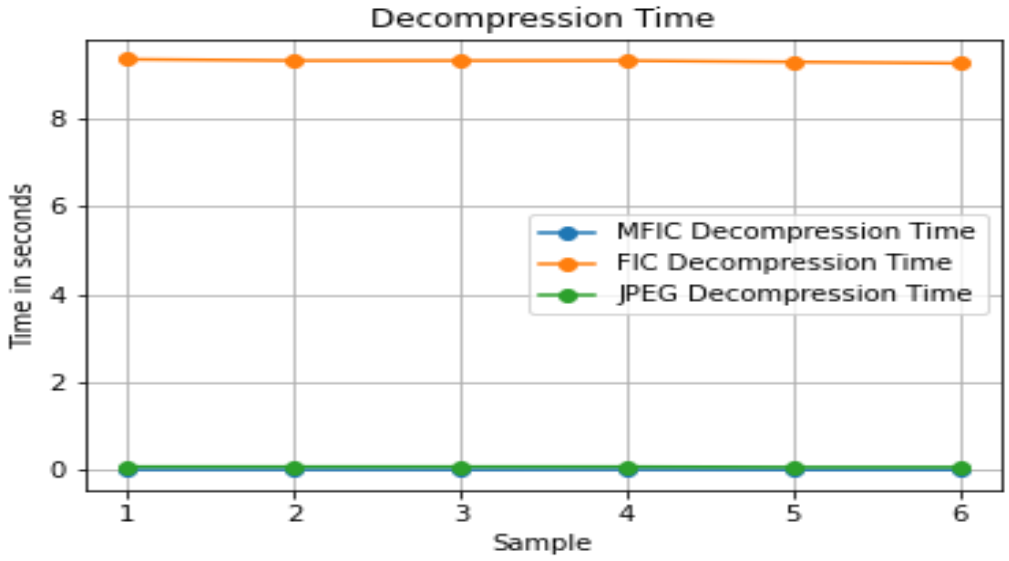

Figure 12. Decompression Time for Kashmir 512x512

Figure. 12 shows the Decompression Time in seconds comparison of the Kashmir image of 512x512 resolution compressed by each method. FIC is seen to be taking the most time, with 9.34-9.25 seconds.

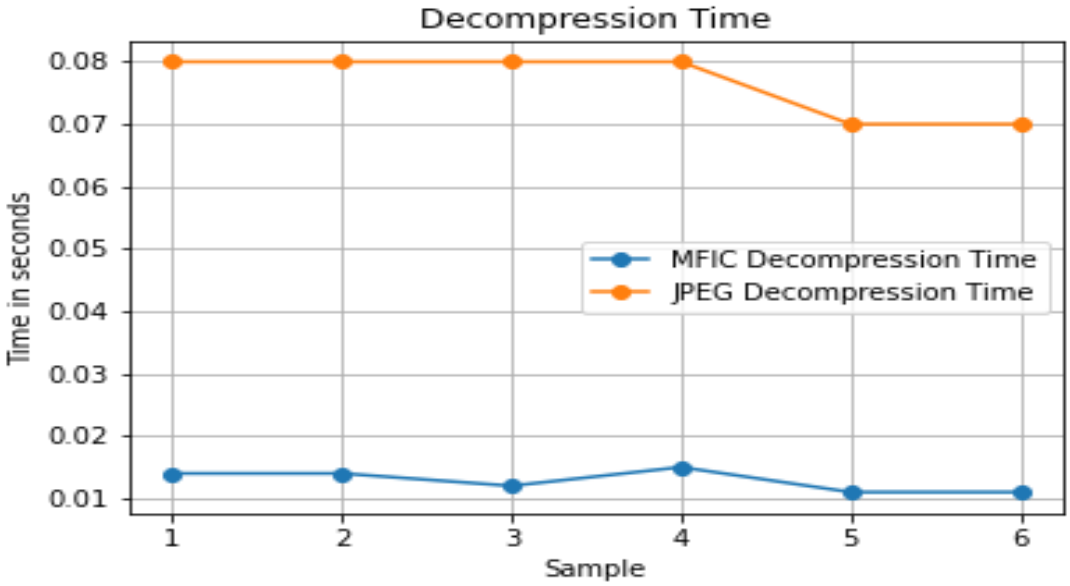

Figure 13. Decompression Time for Kashmir 512x512

Whereas MFIC is the fastest to perform decompression taking 0.014-0.011 seconds compared to JPEG, which took 0.08 seconds. Hence, the Decompression time in MFIC is 86 times faster than in JPEG.

11.2 Colorado 5125x512 Results

Figure 14. Colorado Compressed Images (a) OG (b) MFIC (c) FIC (d) JPEG

Figure. 14 shows examples of the Colorado image of 512x512 resolution compressed by each method. Figure. 14 (a) of Figure. 9 is the original image. Figure. 14 (b) is the image compressed by

MFIC with MSE = 35.33, CR= 4.74, and PSNR = 32.64. Figure. 14 (c) is the image compressed by FIC with CR = 4.68 and PSNR = 31.32. Figure. 14 (d) is the image compressed by JPEG with Quality = 85, CR = 2.42, and PSNR = 26.93. The results of MFIC are better than the other methods, and the quality of the compressed image looks way better than the Fractal image, which shows much blockiness, and the JPEG image, which loses much contrast.

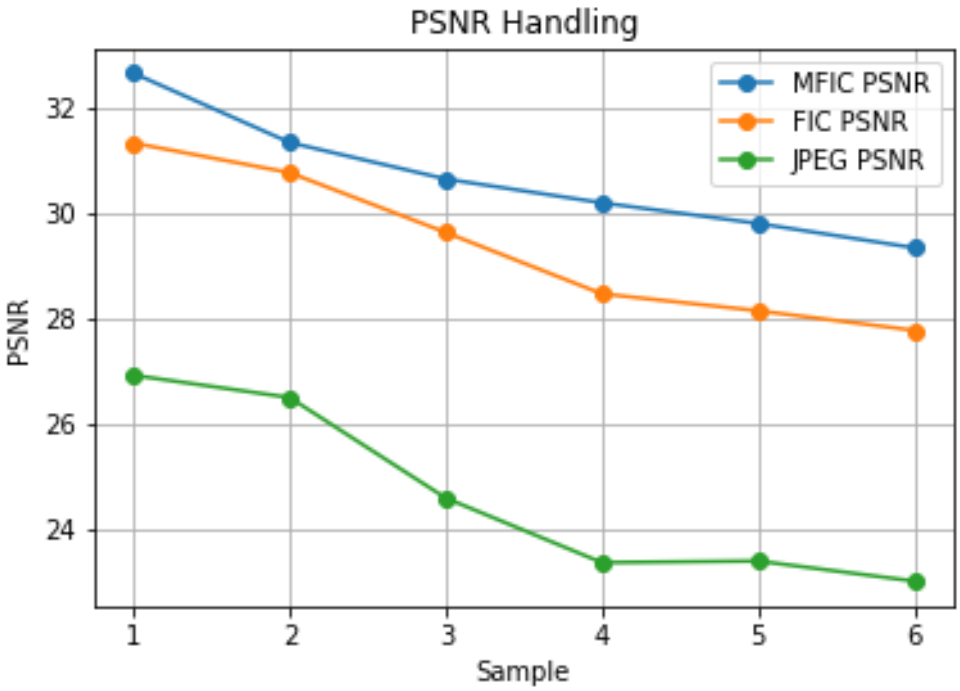

Figure 15. PSNR Handling for Colorado 512x512.

Figure. 15 shows the PSNR comparison for the Colorado 512x512 image for three image compression methods. These methods include each of the MFIC, FIC, and JPEG methods individually. Figure. 15 shows that the MFIC method maintains the PSNR better than the FIC and JPEG methods. However, with the increase in CR, the PSNR levels for JPEG fall much more

rapidly than the other methods. With increased CR, FIC maintains the PSNR levels better than JPEG. However, MFIC maintains the PSNR way better than both the other methods.

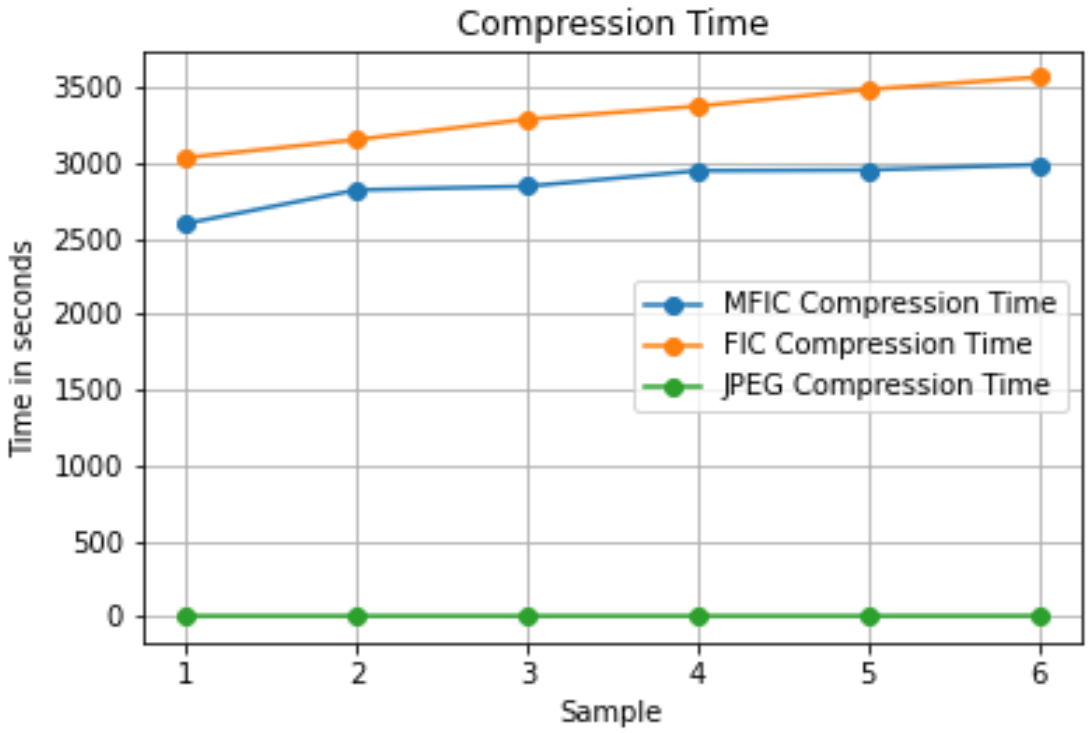

Figure 16. Compression Time for Colorado 512x512

Figure. 16 shows the Compression Time in seconds comparison of the Colorado image of 512x512 resolution compressed by each method. JPEG is the fastest image compression which takes 4-4.62 seconds. MFIC took 2599.64-2989.32 seconds, which is faster than FIC, which took 3035.45-3567.64 seconds, which makes MFIC 14.36% faster than FIC. With MFIC maintaining the PSNR better FIC, it can be said that MFIC compresses different image blocks faster than the FIC method.

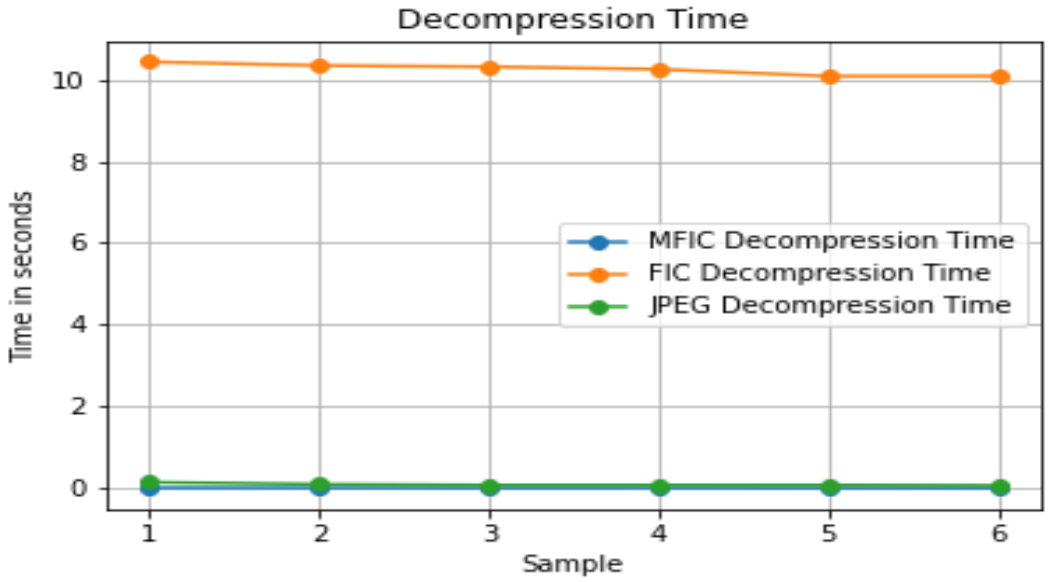

Figure 17. Decompression Time for Colorado 512x512

Figure. 17 shows the Decompression Time in seconds comparison of the Colorado image of 512x512 resolution compressed by each method. FIC is seen to be taking the most time, with 10.43-10.08 seconds.

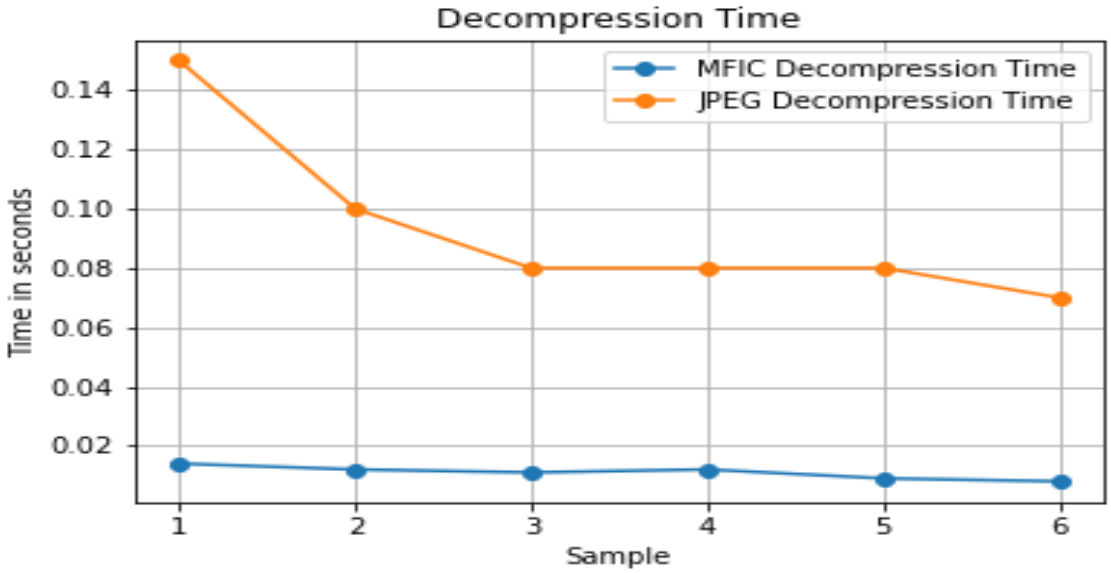

Figure 18. Decompression Time for Colorado 512x512

Whereas MFIC is the fastest to perform decompression taking 0.014-0.008 seconds, compared to JPEG, which took 0.15-0.07 seconds. Hence, the Decompression time in MFIC is 91 times faster than in JPEG.

12. CONCLUSION

The investigations and the simulation results can conclude two essential advantages of the proposed MFIC approach. High decoding speed is provided simultaneously with an acceptable quality of image compression, providing as good or better image quality at a given compression rate as JPEG or FIC. This clearly shows the fundamental perspective and application of the proposed image compression and decoding method. This shows the vital view and applications of the proposed image compression and decoding; this investigation can also be effectively used in video compression, where high-speed decoding of high-definition videos is essential. A drawback of the proposed method is the large computation time required for image coding. However, this drawback is typical for many fractal compression methods.

The extensive search procedure for range block to domain block mapping demands a high computational load. However, this procedure can be executed in parallel with appropriate hardware support. An excellent possible solution is using FPGA methods to speed up the compression.

REFERENCES

[1] Y. Fisher, “Fractal Image Compression - Theory and Application,” NewYork: Springer-Verlag, 1995

[2] M. Barnsley and A. Sloan, “METHODS AND APPARATUS FOR IMAGE COMPRESSION BY ITERATED FUNCTION SYSTEM” U. S. Patent 4,941,193, 10 Jul. 1990.

[3] Fisher, Y., Menlove, S. (1995). Fractal Encoding with HV Partitions. In: Fisher, Y. (eds) Fractal Image Compression. Springer, New York, NY. https://doi.org/10.1007/978-1-4612-2472-3_6

[4] David J Jackson, Wagdy Mahmoud, William A. Stapleton, Patrick T Gaughan, Faster fractal image compression using quadtree recomposition, Image, and Vision Computing, Volume 15, Issue 10,1997, Pages 759-767, ISSN 0262-8856, https://doi.org/10.1016/S0262-8856(97)00020-6.

[5] U. Nandi, S. Santra, J. K. Mandal and S. Nandi, “Fractal image compression with quadtree partitioning and a new fast classification strategy,” Proceedings of the 2015 Third International Conference on Computer, Communication, Control and Information Technology (C3IT), 2015, pp. 1-4, doi: 10.1109/C3IT.2015.7060153.

[6] S. Li and L. Xia, “Non-search fractal image compression algorithm research,” The 26th Chinese Control and Decision Conference (2014 CCDC), Changsha, 2014, pp. 4387-4391, doi: 10.1109/CCDC.2014.6852952.

[7] Stapleton, William & McNees, Michael. (2006). A Three-Component Hybrid Image Compression Method. IPCV 2006: 424-429

[8] Jacquin A.E., “Fractal image coding based on a theory of iterated contractive image transformations,” Proc. SPIE: Vis. Commun. Image Process., M.Kunt, Ed., Lausanne, Switz., vol.1360, pp.227-239, Oct. 1990.

[9] Ponomarenko, Nikolay & Lukin, Vladimir & Egiazarian, Karen & Astola, J.. (2002). Modified horizontal vertical partition scheme for fractal image compression.

[10] Nandi, Utpal & Mandal, Jyotsna. (2012). Fractal Image Compression Using Fast Context Independent HV Partitioning Scheme. Proceedings - 2012 International Symposium on Electronic System Design, ISED 2012. 306-308. 10.1109/ISED.2012.13.

[11] Nandi, U. Fractal image compression using a fast affine transform and hierarchical classification scheme. *Vis* Comput 38,3867–3880(2022), https://doi.org/10.1007/s00371-021-02226-y

[12] Joshi, M., Agarwal, A.K., Gupta, B. (2019). Fractal Image Compression and Its Techniques: A Review. Ray, K., Sharma, T., Rawat, S., Saini, R., Bandyopadhyay, A. (eds) Soft Computing: Theories and Applications. Advances in Intelligent Systems and Computing, vol 742. Springer, Singapore. https://doi.org/10.1007/978-981-13-0589-4_22

[13] 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

[14] Nandi, Utpal & Ghorai, Anudyuti & Laya, Biswajit & Singh, Moirangthem. (2021). A Fast Partitioning Strategy: Its Application to Fractal Image Coding. 10.1007/978-981-15-7511-2_21.