Journal of Kerbala University , Vol. 6 No.2 Scientific. 2008

Speeding up Fractal Compression Using Different Values of Domain Steps Size

تسريع الضغط ألكسوري بواسطة استخدام قيم مختلفة لحجم خطوات المدى

SADIQ HESSEN LEFTA
College of women Science / Babylon University

Abstract:

This research is test a different values of domain block size which used in the domain pool (codebook) constructed from image partitioning, which reduces the computational complexity in the encoding step and which are led to decreasing the encoding time while the reconstruction image from the work as good as we wont .We applied this method on medical images also present a comparison of this method against the quadtree method. The application of fractal images compression presented in this research is based on minimize size of domain pool (codebook) by using different values of domain steps size this led to decreasing of the computational complexity of fractal encoding procedure which make high effective one encoding time that make another way to speeding up fractal medical images compression.

الخلاصة:

هذا البحث يختبر قيم مختلفة لحجم درجة المدى التي تستخدم في المدى المشترك المركب من تقسيم الصورة, هذا يقلل التعقيد في الحسابات في مرحلة الضغط والذي سوف يقود إلى تقليل وقت الضغط إثناء أعادة تركيب الصورة بالوضع الجيد والمطلوب. نحن نطبق هذه الطريقة على الصور الطبية أيضا يقدم مقارنة هذه الطريقة مع طريقة الشجرة المربعة .التطبيق لضغط الصور ألكسوري المقدم في هذا البحث يبنى على تقليل حجم المدى المشترك بواسطة استخدام قيم مختلفة لحجم خطوات المدى هذا يقود إلى تقليل التعقيد الحسابي لإجراء الضغط ألكسوري مما يؤدي إلى فعاليه عالية في وقت التنفيذ هذا يعطي طريقه أخرى لتسريع الضغط ألكسوري للصور الطبية.

1. Introduction

Fractal image compression was first invented by according to the contractive mapping fixed-point theorem. In his proposed iterated function system (IFS), the contacted transform consisting of a sequence of affine transformations is applied to the entire image. Later proposed a partitioned iterated function system (PIFS) associated with a block-based automatic encoding algorithm where those affine transformations are applied to partitioned blocks. Fractal image compression can be used in many applications such as image retrieval, watermark, multimedia encyclopedia, and hybrid coding methods [6][2].

The bottleneck in the PIFS fractal coding scheme is time-consuming in the encoding process. In order to alleviate this serious encoding time problem, several efficient fractal encoding algorithms have been developed. These developed encoding algorithms include partitioned-based approach, domain pool selection approach, and search strategy-based approach. Recently, Furao and Hasegawa presented a no search fractal encoding method and experimental results showed the speeding up of the encoding time when compared to the previous method by Tong and Wong, but having little image quality degradation. Note that Tong and Wong_s encoding method have better image quality and encoding time performance when compared with Saupe_s fractal coding method. Currently, Truong et al presented an efficient spatial-correlation-based algorithm for fractal encoding and their proposed algorithm had a significant improvement when compared with the baseline algorithm, i.e. the full search algorithm. In this research, a two-phase prediction- and sub block-based fractal encoding algorithm is presented. Initially the original gray image is partitioned into a set of variable-size blocks according to the S-tree- and interpolation-based decomposition

principle. In the first phase, each current block of variable-size range block tries to find the best matched domain block based on the proposed prediction-based search strategy which utilizes the relevant neighboring variable-size domain blocks. The first phase leads to a significant computation-saving effect. If the domain block found within the predicted search space is unacceptable, in the second phase, a sub block strategy is employed to partition the current variable-size range block into smaller blocks to improve the image quality. Experimental results show that our proposed prediction- and sub block-based fractal encoding algorithm outperforms the conventional full search algorithm and the recently published spatial-correlation-based algorithm by Truong et al. in terms of encoding time and image quality. In addition, the performance comparison among our proposed algorithm and the other two algorithms, the no search-based algorithm and the quadtree-based algorithm, are also investigated [3][6].

The remainder of research is organized as follows. Our proposed prediction- and sub block-based encoding algorithm is presented in Section 6, some experimental results are demonstrated in section 9. Conclusions are addressed in section 10.

2. Concepts of Fractal Image Compression

In the following section, the basic concepts of fractal image compression on the traditional square structure would be introduced. Before delving into details, there are some highlights of fractal image compression.

- It is a promising technology, though still relatively immature.
- The fractals are represented by Iterated Function Systems (IFSs).
- It is a block-based lossy compression method.
- Compression has traditionally been slow but decompression is fast.

The fundamental principle of quadtree compression consists of the representation of an image by an iterated function system (IFS) of which the fixed point is close to that image. This fixed point is named as ‘fractal’ [2]. Each IFS is then coded as a contractive transformation with coefficients. Banach’s fixed point theorem guarantees that, within a complete metric space, the fixed point of such a transformation may be recovered by iterated implementation there of to an arbitrary initial element of that space. Therefore, the encoding process is to find an IFS whose fixed point is close to the given image. The usual approach is based on the collage theorem, which provides a bound on the distance between the image to be encoded and the fixed point of an IFS [5] .

A suitable transformation may therefore be constructed as a ‘collage’ from the image to itself with a sufficiently small ‘collage error’ (the distance between the collage and the image) guaranteeing that the fixed point of that transformation is close to the original image. In the original approach, devised by Barnsley, this transformation was composed of the union of a number of affine mappings on the entire image. While a few impressive examples of image modeling were generated by this method (Barnsley’s fern, for example [2]), no automated encoding algorithm was found. Fractal image compression became a practical reality with the introduction by Jacquin of the partitioned IFS (PIFS) [8], which differs from an IFS in that each of the individual transformation operates on a subset of the image, rather than the entire image. Since the image support is tiled by ‘range blocks’, each of which is mapped from one of the ‘domain blocks’ as depicted in (Figure 2), the combined mappings constitute a transformation on the image as a whole. The transformation minimizing the collage error within this framework is constructed by individually minimizing the collage error for each range block, which requires locating the domain block which may be made closest to it under an admissible block mapping. This transformation is then represented by specifying, for each range block, the identity of the matching domain block together with the block mapping parameters minimizing the collage error for that range block[6][7].

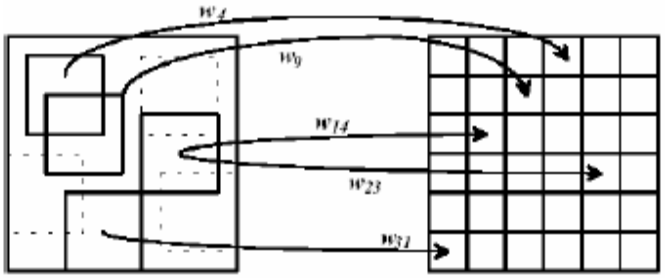


Figure (1): Each range block is constructed by a Transformed domain block

3. Scale less and Resolution Enhancement

When an image is captured by an acquisition device, such as a camera or scanner, it acquires a scale determined by the sampling resolution of that device. If software is used to zoom in on the image, beyond a certain point you don't see additional detail, just bigger pixels. A fractal image is different. Because the affine transformations are spatially contractive, detail is created at finer and finer resolutions with each iteration .In the limit; self similar detail is created at all levels of resolution, down the infinitesimal. Because there is no level that 'bottoms out' fractal images are considered to be scale less. What this means in practice is that as you zoom in on a fractal image, it will still look 'as it should' without the staircase effect of pixel replication. The significance of this is cause of some misconception, so here is the right spot for a public service announcement. Iterated Systems is fond of the following argument. Let us say, a grayscale image 250x250 pixels in size, 1 byte per pixel. You run it through their software and get a 2500 byte file (compression ratio = 25:1). Now zoom in on the person's hair at 4x magnification. What do you see? A texture that still looks like hair .Well then, it's as if you had an image 1000x1000 pixels in size. So your effective compression ratio is 25x16=400. But there is a catch. Detail has not been retained, but generated. With a little luck it will look as it should, but don't count on it. Zooming in on a person's face will not reveal the pores. Objectively, what fractal image compression offers is an advanced form of interpolation. This is a useful and attractive property. Useful to graphic artists ,for example, or for printing on a high resolution device. But it does not best fantastically high compression ratios. That said, what is resolution enhancement? It is the process of compressing an image, expanding it to a higher resolution, saving it, then discarding the iterated function system. In other words, the compressed fractal image is the means to an end, not the end itself [2][6].

4. Error Metrics

Two types of the error metrics used to compare the various image compression techniques are the Mean Square Error (MSE) and the Peak Signal to Noise Ratio (PSNR). The MSE is the cumulative squared error between the compressed and the original image, whereas PSNR is a measure of the peak error. The mathematical formulae for the two types are[4]:

$$\text{MSE} = \frac{1}{N \times N} \sum_{X=0}^{N-1} \sum_{Y=0}^{N-1} \left[f'(x,y) - f(x,y) \right]^2 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

PSNR (dB) = 10 * $\log_{10}(255^2 / \text{MSE})$ …….. (2)

$$\text{SNR}=10\log_{10}\left[\frac{\max f(x,y)-\min f(x,y)}{MSE}\right] \text{……. (3)}$$

Where f (x, y) is the original image, f'(x, y) is the approximated version (which is actually the decompressed image) and M,N are the dimensions of the images. A lower value for MSE means lesser error, and as seen from the inverse relation between the MSE and PSNR, this translates to a high value of PSNR. Logically, a higher value of PSNR is good because it means that the ratio of signal to noise is higher. Here, the 'signal' is the original image, and the 'noise' is the error in reconstruction. So ,if you find a compression scheme having a lower MSE (and a high PSNR) ,it can be recognized that it is better one [1][8].

5. Essentials of fractal image coding

Fractal image coding is based on the original theory of Iterated Function Systems (IFS). In what follows, we provide some bare essentials of fractal image coding. For those readers interested in seeing the larger mathematical picture, we summarize the main ideas of contraction maps and associated inverse problems in Appendix A. Let Γ denote an image of interest as defined by an image function u(x, y) supported over a region X ϵ R^2 . Here x,y ϵ X denote spatial coordinates of a point or pixel of Γ . Now suppose that there exists a suitable partition R of X into range sub-blocks Ri so that $X=U_iR_i$. For simplicity, the R_i are assumed to be "non over lapping," intersecting only over their boundary curves. (In the discrete case, i.e., pixels, there is no overlapping.) Assume that associated with each sub-block R_i is a larger domain sub-block D_i so that $R_i = w_i (D_i)$, where w_i is a one-to-one contraction map in the continuous (non digitized) plane., typically, the R_i and D_i blocks are square pixel blocks and the w_i are affine contractions that may also rotate or invert the R_i . In this case there are eight such possible maps [4][8].
For each range (R_i) an optimal approximation must be obtained from the following equation:

$$R \approx sD \div o \text{……. (4)}$$

The computation; needs for each codebook (D_i) compute an optimal approximation (i.e. s and o values) is determined by calculate the values of scaling (s) and offset (o) coefficients by using the least square optimization, i.e. :

$$s=\frac{n\sum_{i=1}^{n} d_i r_i - \sum_{i=1}^{n} d_i \sum_{i=1}^{n} r_i}{n\sum_{i=1}^{n} d_i^2 - \left(\sum_{i=1}^{n} d_i\right)^2} \text{ …… (5)}$$

$$o=\frac{1}{n}\left(\sum_{i=1}^{n} r_i \quad - s\sum_{i=1}^{n} d_i\right) \text{ ……..... (6)}$$

And

$$d_{rms}(f,g)=\sqrt{\frac{1}{N\times N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}\left(f(x,y)-g(x,y)\right)^2} \quad \ldots\ldots\ldots\ldots (7)$$

Where, r is the range of block.
d is the domain of block**,**
n is the number of pixels.

6. Encoding Scheme

6.1 Quadtree Partitioning

This is perhaps the most common method of adaptive partitioning. Beginning with the original image, square pixel blocks are broken down into quadrants in a recursive tree structure. The partitioning, which will vary throughout the image,is terminated when a particular condition is satisfied. Typically, regions of higher image activity, for example edges, will produce partitions of finer resolution, i.e., small block sizes. As such, edge information is rather well represented in quadtree-based coding schemes, including fractal coding[10].

Typically more detailed areas of the original image need smaller size range blocks, in order to be matched accurately with a domain block. Because of this, quadtree partitioning uses square range blocks that can vary in size. If a large range block does not have a suitably accurate match with some domain block, using the standard block encoding, then it is divided into four quadrants and the process is repeated. Two integer parameters specify the maximum and minimum possible sizes for range blocks. A sample result of using quadtree partitioning is illustrated in Figure 2.Notice how the blocks are much smaller in the more detailed areas of the image [9][10].

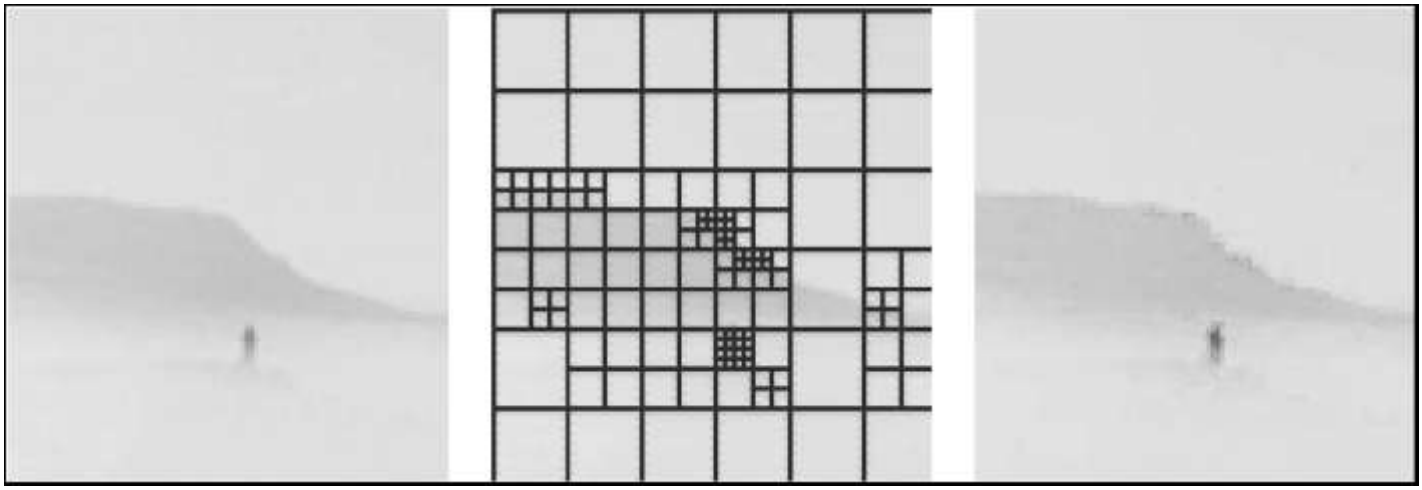
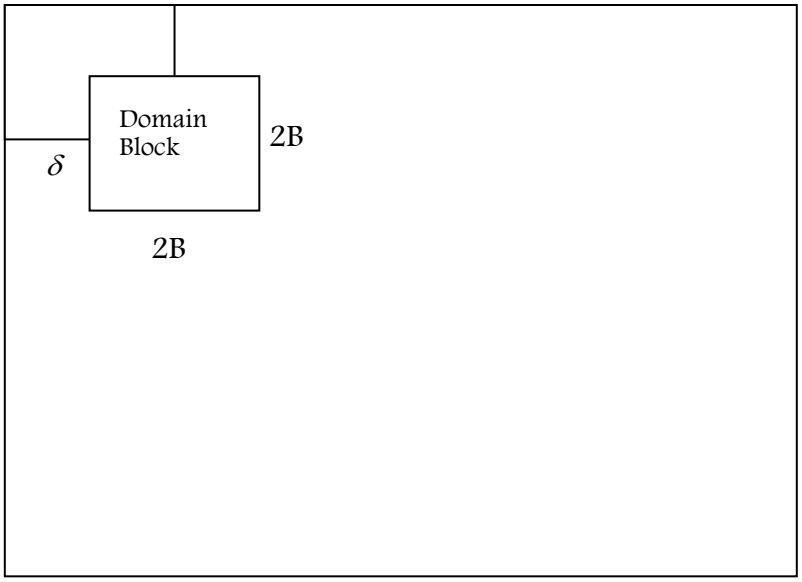


Figure (2): Quadtree partitioning: original (left), partitions (centre), decoded image (right)

6.2 Domain block pool

As described above, for each range block, a domain block and map need to be found so that range block and domain block become the best pair. Then where to hunt the best-pair domain block?---- Among the domain block pool.

The domain block pool consists of 2Bx2B squares from the original image and is a set of domain blocks. It can be generated by sliding 2Bx2B window within the original image by skipping δ pixels from left to right, top to bottom [8].



Figure(3):Domain step size

6.3 Encoding algorithm

1. The description of the fractal encoding process can be given below : Enter a minimum error between mach block, d_{min} .
2. Choose the original image, call this image the Range-image (R-image).
3. Resample the original image to create another image of the same size, call this image the Domain-image (D-image).
4. Select the domain step size.
5. Partition the R-image by some collection of ranges R of size $B \times B$.
6. Partition the D-image by some collection of domains of size $2B \times 2B$.
7. For each $R_i \in R$, do the following steps;
8. Loop on all of the D-regions.
9. Down sampling of D-image by using average.
10. Compute contrast (s) and brightness (o) by using Eq.(5) and Eq.(6) respectively and then root mean square error (d_{rms}) between two blocks R and D by using Eq.(7).
11. Choose the minimum $d_{rms} < d_{min}$ between mapping blocks.
12. Store the fractal codes parameters (s , o , D_x ,D_y) out to a file.

7. Encoding Time

The objective of this project is to reduce the time complexity in the encoding step. where the time complexity arises in the encoding step the time complexity arises, and how we can reduce it. Consider an image that is 256x256 pixels in size. We partition it into 8x8 non-overlapping ranges R.We also partition it into 16x16 overlapping domains D = D ,58;081. Now, for each Ri, we search through all D to find a Di which minimizes some threshold. In other words, we try to find a part of the image that looks similar to Ri. There are eight ways to map one square to another. So for each range Ri, 464,648 domains have to be compared. Various techniques have been proposed to overcome the time complexity. One such technique is the classification of domains based on some feature, such as the edges or bright spots. A domain once discarded removes from the pool all other similar domains for a particular range. Another technique is to classify the domains as multidimensional keys. This reduces the complexity from O(N) to O(log N) . The weakness of this method is that the size of the range Ri is fixed. This is overcome in the quadtree partition method. Here, a square in the image is split up into four equal-sized sub-squares when it is not covered sufficiently by the domain. This process is repeated recursively, starting from the complete image and continued until the squares are small enough to be covered within some specified threshold [7].

8. The Speed Problem

The essence of the compression process is the pairing of each range block to a domain block such that the difference between the two, under an affine transformation, is minimal. This involves a lot of searching.

In fact, there is nothing that says the blocks have to be squares or even rectangles. That is just an imposition made to keep the problem tractable.

More generally, the method of finding a good PIFS for any given image involves five main issues:

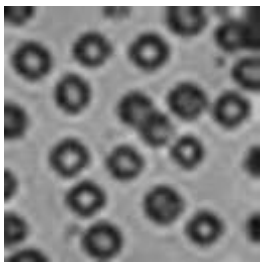
1. Partitioning the image into range blocks.
2. Forming the set of domain blocks.
3. Choosing type of transformations that will be considered.
4. Selecting a distance metric between blocks.
5. Specifying a method for pairing range blocks to domain blocks.Many possibilities exist for each of these. The choices that Jacquin offered in his paper are:

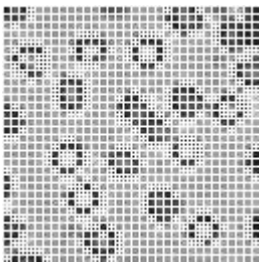
1. A two-level regular square grid with 8x8 pixels for the large range blocks and 4x4 for the small ones.
2. Domain blocks are 16x16 and 8x8 pixels in size with a sub sampling step size of four. The 8 isometric symmetries (four rotations, four mirror flips) expand the domain pool to a virtual domain pool eight times larger.
3. The choices in the last point imply a shrinkage by two in each direction, with a possible rotation or flip, and then a translation in the image plane.
4. Mean squared error is used.
5. The blocks are categorized as of type smooth, midrange, simple edge, and complex edge. For a given range block the respective category is searched for the best match [8].

9. The Results

In this research used the two types images to explain the domain step size effectives on the quality of the reconstruction image and effect it on compression values and speeding up of the encoding operation which are refer by the encoding time also the experiment results show the relation ship between the domain step size values with compression ratio and domain step size

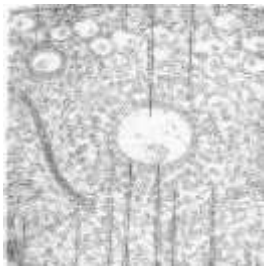
values with PSNR values which represented the quality measurement and explain the relationship between the domain step size values and encoding time. According to this work we need to improve the values of encoding time to make the encoding time as acceptable as quality of reconstructed image. in this work used three medical Gray scale images with size 256× 256 and used the quad tree portioning techniques with minimum block size 4 and maximum block size 8. In the following experiment results we used different values of standard deviation (std) and Mean (M) to find the best values of each one which led to improve the values of encoding time to make the encoding time as acceptable as quality of reconstructed image. The best value of Mean(M) is 0.1 and the best value of standard deviation is 0.7 which are using in this work to make the encoding time as acceptable and high quality of reconstructed image. The minimum value an maximum value of the scaling which are used in this results (from -1.5 to 1.5) and the minimum value and maximum value of the offset (from -200 to 500) and
the Qntscl =100.00 and Qntofs=1.000 and the number of iterations are used in decompression process for extracting the reconstruction image are 8 iterations.

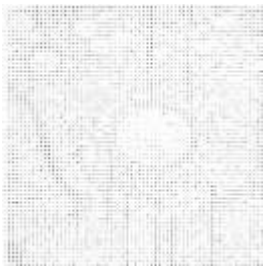




Figure(4): A1 image with domain step size= 2

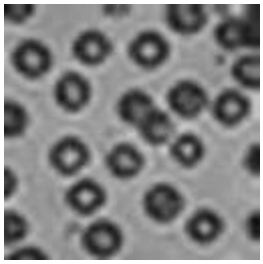
- a-

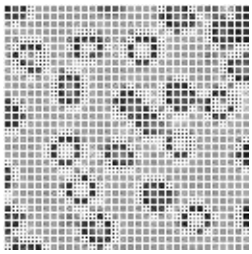




figure(5): A2 image with domain step size= 2

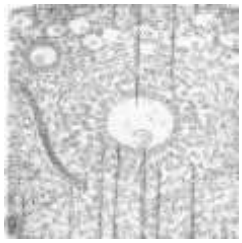
-a-

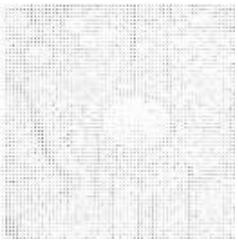




Figure(4):A1 image with domain step size= 4

-b-





Figure(5): A2 image with domain step size= 4

-b-

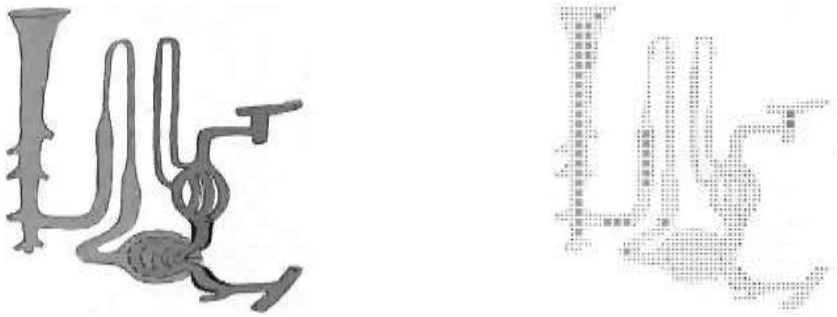


Figure (6): A3 image Domain step size = 4
-a-

Figure (6): A3 image with Domain step size = 2
Number of block = 1965
-b-

Figure (6): A3 image with Domain step size = 2
Number of block =2225

-c-

In the result image (A3) with domain step size 2, we notice that image have blockness and for obtaining better result image (without blockness) and have high quality, we modify the value of standard deviation (Std) from 0.7 to 0.2 that will be make high quality reconstructed image. In above image we notice that the value of standard deviation (std) effect on the number of blocks in the domain pool of using image in the encoding stage also effect on the value of PSNR.

Table 1: deterministic the output value of compression process of A3image

Domain step size	Compression Ratio	SNR	PSNR	Time(second)
2	6.89	28.239	29.158	12.218
4	8.22	26.804	27.723	6.329
6	8.20	26.057	26.976	5.248

Table 2: deterministic the output value of compression process of A2 image

Domain step size	Compression Ratio	SNR	PSNR	Time(second)
2	3.86	19.487	21.124	20.554
4	4.10	18.944	20.582	11.827
6	9.70	16.670	23.799	4.853

Table 3: deterministic the output value of compression process of A1 image

Domain step size	Compression Ratio	SNR	PSNR	Time(second)
2	9.25	32.933	38.596	44.272
4	9.84	32.451	38.115	22.69
6	9.87	32.154	37.817	13.103

In the above tables we notice the values of PSNR of each image have great different from another image that's because the different between the type of details of original image this details shown clearly in quadtree result images

In the following shapes we explain the comparison between the values of domain step sie with values of
PSNR , compression ratio and the value of encoding time of image.

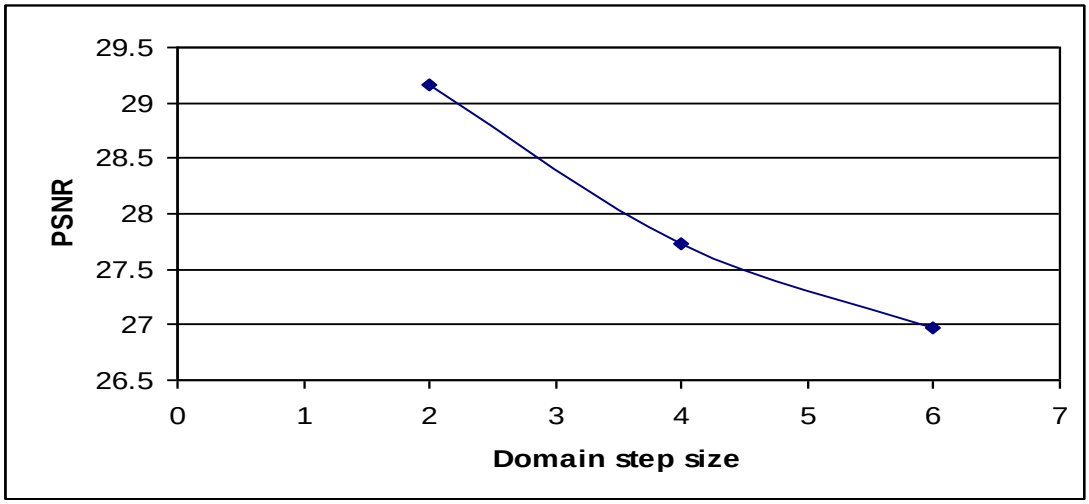


Figure (7): The relationship between domain step size and PSNR

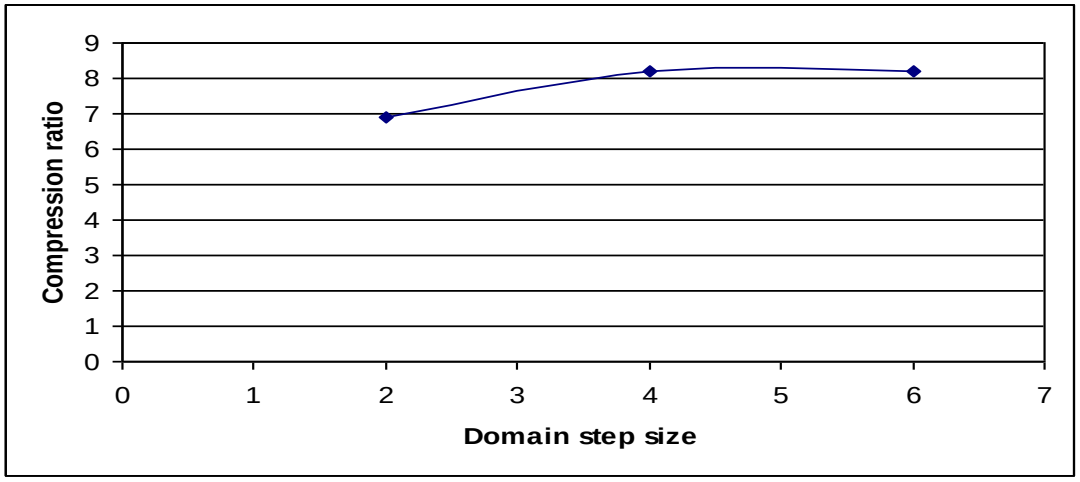


Figure (8): The relationship between domain step size and compression ratio

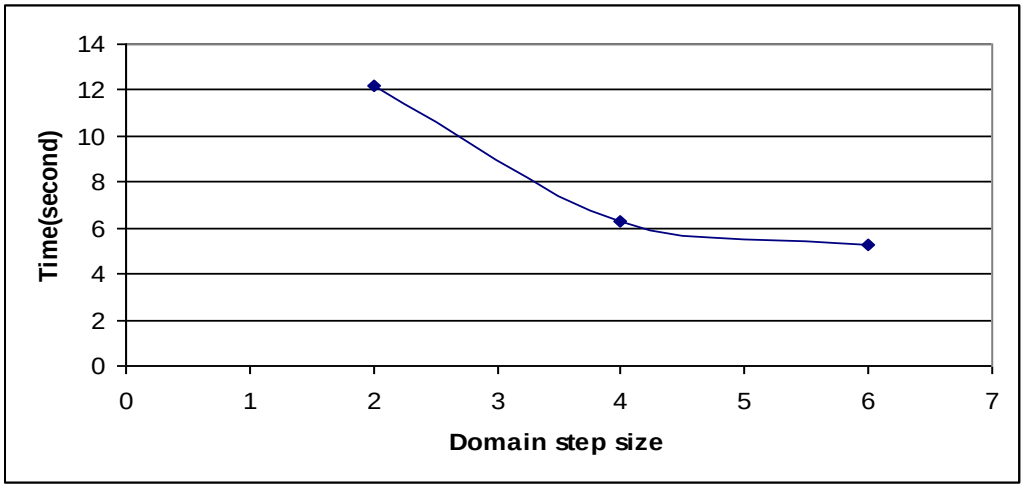


Figure (9): The relationship between domain step size and time

10. Conclusions and discussion:

In this work, the fractal image compression method was implemented . As mentioned in the previous work, this method has the disadvantages of a very long encoding time. For this reason, a new approach are suggested for speeding up the operations in the encoding process, which leads to reduce that time. And we used this approach.

Many tests were preformed on different images to study the behaviour of normal and speeded FIC method. From the results of these tests , the following conclusions could be summarized:

- The encoding time is inversely proportional with. Domain step size.
- PSNR is inversely proportional with Domain step size.
- Compression ratio is proportional with Domain step size .
- The encoding time is inversely proportional with each of :Mean (M) and Standard deviation (Std).
- PSNR is inversely proportional with each of (M)And (Std).
- Compression ratio is proportional with each of And (Std) .

Number of blocks of quadtree results image are inversely proportional with each of (M) and (Std).

2. In order to keep the quality of the reconstructed image, it is not recommended to increase the block size more than 16×16 although this operation will increase the compression ratio.
3. The encoding time is inversely proportional with number of blocks, also the PSNR is inversely proportional with number of blocks , while the compression ratio C.R is proportional with number of blocks.
4. The encoding time is inversely proportional with block size, also the PSNR is inversely proportional with block size ,while the compression ratio C.R is proportional with block size.
5. In case of the suggested approach for the encoding time reduction, the testing results showed that this approach achieve a relatively very short encoding time compared with the classical implementation of FIC method.
6. In the test results we observed that the Mean (std) values changed from one application to another, for example in our work the value of std=0.7 is used but in some cases this value failed so we try to use high and low value nearer from this one at (M=0.2) gives the best results (no blockness effects) . So are M values.

Reference:

[1] Viswanath Sankaranarayanan, 4th December, 1998 ' **Fractal Image Compression',** Project Report.

[2] A. E. Jacquin, Oct. 1993, ' **Fractal Image Coding** ': A Review, Proceedings of the IEEE, Vol.81, No.10.

[3] Mohsen Ghazel, George H. Freeman, and Edward R. Vrscay, SEPTEMBER 2006'**Fractal-Wavelet Image Denoising Revisited**', IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 15, NO. 9

[4] Jamila Harbi Saoud Al-A'mri, January 2001,'**Fractal Image Compression**'

[5] La Torre, F. Mendivil and E.R. Vrscay, (Springer-Verlag, Heidelberg,2006),' **Iterated function systems on multifunctions**', in Math Everywhere.

.

[6] Kuo-Liang Chung *, Chung-Hsiang Hsu, Accepted 16 August 2005,' **Novel prediction- and subblock-based algorithm for fractal image compression"** Department of Computer Science and Information Engineering',National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taiwan ROC.
[7] K. U. Barthel* , H. L. Cycon**, DECEMBER 2003,' **Image denoising using fractal and wavelet-based methods',**Fachhochschule für Technik und Wirtschaft Berlin ,University of Applied Sciences.
[8] Hanaa mehsen ali,May 2005,**'speeding up fractal image coding by using indexing block technique'.**
[9] S. A. Curtis and C. E. Martin, 2003,**' Functional Fractal Image Compression'** Department of Computing, Oxford Brookes University, UK.
[10] Mohsen Ghazel, George H. Freeman, and Edward R. Vrsca, DECEMBER 2003,**'Fractal Image Denoising'**