Quad-tree Fractal Image Compression

Dr. Fakhiraldeen H. Ali
lecturer
Dep. of Computer Eng.
University of Mosul

Azzam E. Mahmood
Msc. Student
Dep. of Computer Eng.
University of Mosul

Abstract

The demand for images, video sequences and computer animation has increased drastically over the years. This has resulted in image and video compression becoming an important issue in reducing the cost of data storage and transmission time. JPEG is currently the accepted industry standard for still image compression, but alternative methods are also being explored. Fractal Image Compression is one of them. In this paper implementation of fractal compression with quadtree partitioning have been done, this paper also includes a study of the parameters that affect fractal image compression.

Keywords: fractal, compression, quadtree .

hvĭβNjƨlllε hvƨσqdžNJllε !ε σQHƨzFG' YβNjTllε Ƨ G'l UDŽ hvĭYFG'βllε YβШllε !ε σQHƨzε

σqNjDŽΘDŽ !FTY !ε₹Y
βĭH ƨШFDŽ f llFU'
l FG'ƨzFΘllε hvƨσqdžlj !ƨƽ
üTNjDŽllε hvyDŽfШ

Ĭ ǂY σqDŽFΘ Džĭσqllε βQlƺ .σq
Ƨ βqDŽ
l FG'ƨzFΘllε hvƨσqdžlj !ƨƽ
üTNjDŽllε hvyDŽfШ

:hvTúQllε
Σhvù ǂ yôù u'fÅdž KơhvTD ÜŁŏ!G'ơ!f KfÅLjl!f ýŭP Åŏβŏƽ!fÅ ƧÅTl!f Ĭ!f lβŏfƨkdž!f ŁηG'ơ!! fƧULj Ôù5 ýIlj!f KpÅ ýŏ!IkÅ KGljG'ôhv!f LJƨP Ł₹ǂ ýŏ!Ik ýηD LJdž Łdžnjdž!f yôu'fÅdž!f LJdž Åŏβŏƽ!fÅ ƧÅTl!f yùdž ÜŁŏlG'ôI!f ŁŏZG'LjTl!f ƽFfƧŪ!f DžNjD ăβσϱ (JPEG)ù!f Ił ŏƧŪ Ƨhvkzk ł ŏơ KŬG'TkŬf ŁdžULjD Σhv ǂ Łŏljlk ƽƧŪ!f ljRNj LJdž ƧŏÅŪƽk!fÅ ł ơhv!f βŏp Åŏβŏƽ!f Å ƧÅTl!f Σhv ǂ! ăƧPD ƽFfƧŪ ll !GLjNj ll !R Σhvù ǂ yđž ŁŏZG'hvƧ!f lƧηT!f DžfβPklf Ĭ!f ł ơhv!f fRNj Ƨβnjŏ .Łŏlβljnj!f ƧÅl ǂ!f DžfβPklG'hv lƧÅTl!f lƧλdùdž!f ýdžfÅƶ!f Ił fƧβ Ĭ!ϱ G'u'ŏD ł ơhv!f fRNj Ƨβnjŏ G'dž ǂ ÜŁŏƧÅl ǂ!f Ił βljnj!f DžfβPklG'hv ƧÅTl!f .ŁŏƧÅl ǂ!f Ił βljnj!f DžfβPklG'hv ƧÅTl!f Σhv ǂ ĬllZ

Received 16 August 2005 **Accepted 5 Feb. 2006**

1-Introduction

Fractal is first introduced in geometry field. The birth of fractal geometry is usually traced back to the IBM mathematician Benoit B. Mandelbrot and the 1977 publication of his book "The Fractal Geometry of Nature". Later, Michael Barnsley, a leading researcher from Georgia Tech, found a way of applying this idea to data representation and compression with the mathematics of Iterated Functions Systems (IFS). Regarding the computational complexity, fractal compression algorithm based on IFS was still not practical to use at that time. And it is Arnode Jacquin, who finally settled down this problem with Partitioned Iterated Function Systems (PIFS), which is modified from IFS by partitioning the domain space into subspaces.

Wohlberg and de Jager conclude that the fundamental principle of fractal coding consists of the representation of an image by a contractive transform of which the fixed point is close to that image at their review. By the contractive mapping theorem and collage theorem, the contractive transform is always possible within certain threshold. Original approach taken IFS tries to find a number of affine mappings on the entire image, which is rather slow in terms of searching the contractive map function. Jacquin's PFIS takes different approach to find the individual mappings for subsets of the images. The partition of the image has been introduced to get those subsets. The main concept is to partition the image space twice as "range blocks" and "domain blocks". Both partitions follow the same scheme in geometric sense covering the whole image, but the latter is allowed to overlap with each other. Then each range block is mapped onto one domain block by taking certain transformations such as rotation, flip, etc. One mapping is eventually identified by two variables, named scaling and offset. Most cases are measured by distance between points under MSE (Mean-Squared Error) [1][2][3][4][5].

2-Fractal Image Compression:

Imagine a special type of photocopying machine that reduces the image to be copied by a half and reproduces it three times on the copy. Figure 1 shows this. Now, feed the output of this machine back as input. Figure 2 shows several iterations of this process on several input images. We observe that all the copies seem to be converging to the same final image, the one in figure 2(c). We call this image the attractor for this copying machine. Because the copying machine reduces the input image, any initial image will be reduced to a point as we repeatedly run the machine. Thus, the initial image placed on the copying machine doesn't effect the final attractor; in fact, it is only the position and the orientation of the copies that determines what the final image will look like.

Since it is the way the input image is transformed that determines the final result

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

of running the copy machine in a feedback loop, we only describe these transformations. Different transformations lead to different attractors, with the

technical limitation that the transformations must be contractive – that is, a given transformation applied to any two points in the input image must bring them closer together in the copy. In practice, choosing transformations of the form is sufficient to yield rich and interesting set of attractors. Such transformations are called affine transformations of the plane, and each can skew, stretch, rotate, scale and translate an input image[6][7].

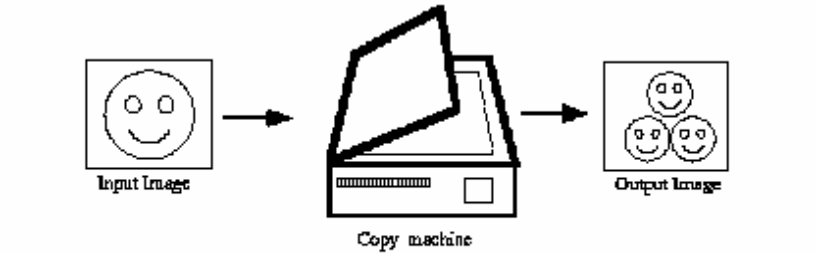


Figure 1. A copy machine that makes three reduced copies of an input image

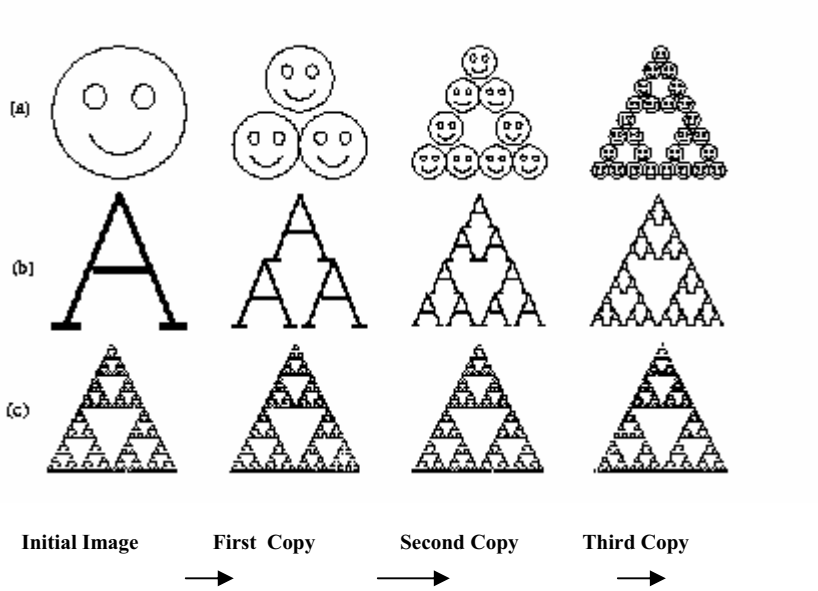


Figure 2. The first three copies generated by the copying machine of figure 1.

2-1 Advantages and disadvantages of fractal image compression:

When the fractal image compression is compared to other methods used to compress different images, some of the main advantages and disadvantages can be summarized.

2-1-1 Method Advantages:

- Good mathematical encoding frame.
- Resolution-free decoding.
- High compression ratio
- Quick decompression.

2-1-2 Method disadvantages:

- Slow encoding**.**

2-2 Contractive transformations

A transformation *w* is said to be contractive if for any two points P1,P2, the distance:

$$d(w(P1), w(P2)) < s\,d(P1,P2)$$

for some $s < 1$. This formula says that application of a contractive map always brings points close together (by some factor less than 1).

Contractive transformations have the nice property that when they are repeatedly applied, they converge to a point which remains fixed upon further iteration[6][7][8][9].

2-4 Self Similarity in Image :

A typical image of a face , for example figure 3-A does not contain the type of self-similarity that can be found in the fractals of figure 3-B. The image does not appear to contain affine transformations of itself. But , in infact, this image does contain a different sort of self-similarity. Figure 4 shows sample regions of Lena which are similar at different scales : a portion of her sholder overlaps a region that is almost identical, and a portion of the reflection of the hat in the mirror is similar(after transformation) to a part of her hat. The distinction from the kind of self-similarity we saw in figure 3-A is that rather than having the image be formed of copies of its whole self(under appropriate affine transformation), here the image will be formed of copies of properly transformed parts of itself. This means that the image we encode as a set of transformations will not be an identical copy of the original image but rather an approximation of it.

Experimental results suggest that most images that one would expect to "see" can be compressed by taking advantage of this type of self-similarity; for example, images of trees, faces, houses, mountains, clouds, etc[7][8].

(B) **(A)**

Figure 3. The original Lena Image and its self similar portions

2-5 Partitioned Copying Machines

In this section we describe an extension of the copying machine metaphor that can be used to encode and decode grey-scale images. The partitioned copy machine we will use has four variable components.

- The number of copies of the original pasted together to form the output.
- A setting of position, scaling, stretching, skewing and rotation factors for each copy.

These features are a part of the copying machine definition that can be used to generate the images in figure 3. We add the following two capabilities:

- A contrast and brightness adjustment for each copy.
- A mask which selects for each copy a part of the original to be copied.

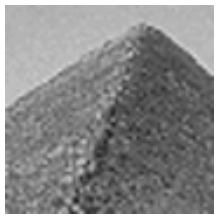
These extra features are sufficient to allow the encoding of grey-scale images. It partitions an image into pieces which are each transformed separately. By partitioning the image into pieces, we allow the encoding of many shapes that are difficult to encode using an IFS[6][7][8].

2-6 Massic part of affine transformations

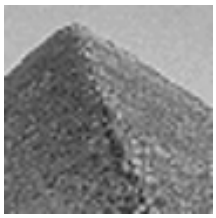
The massic part of the transformation produces modification of the pixel values inside the block. It allows to change the gray level information in order to get good approximation of the block Ri. In the implementation, we will consider only 8 possible shuffles of pixels (4 rotations and 4 reflections called isometries), a contrast scaling s_i and brightness shift o_i. The massic part of the transformation can therefore be expressed as

$$\mathrm{Si}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ o_i \end{bmatrix}$$

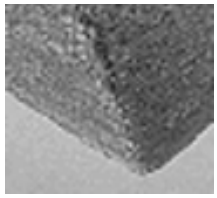
a_i, b_i, c_i, d_i correspond, applied to an image, to one of the eight isometrics, s_i represents a contrast scaling, o_i represents a brightness shift. The eight affine transformations considered are shown in Figure 4 [1].



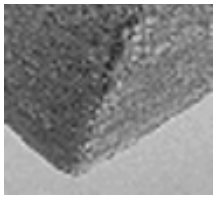
0: No



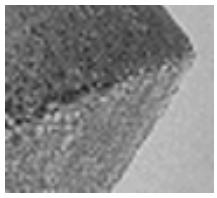
1: Reflection in the y axis



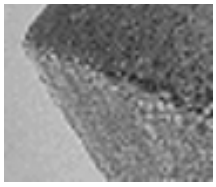
2: Reflection in the x axis



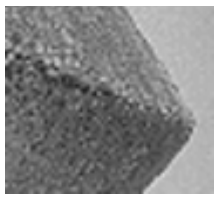
3: 180° Rotation



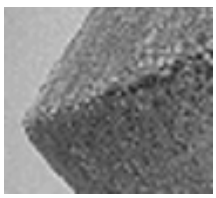
4: Reflection in the line y = x



5: Rotate 90° counter-clockwise



6: Rotate 90° clockwise



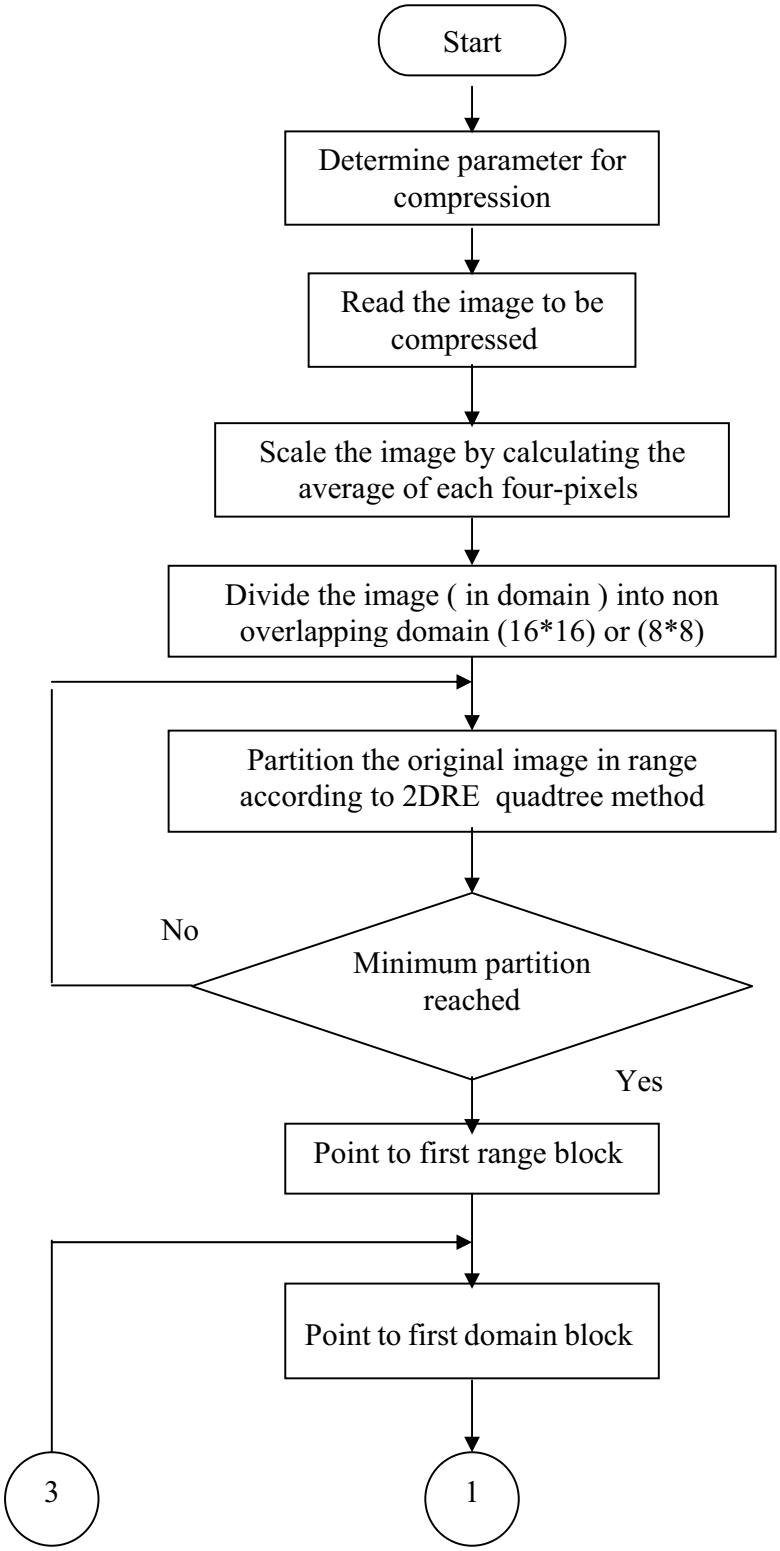
7: Reflection in the line y = –x

Figure 4 Massic part of affine transformations

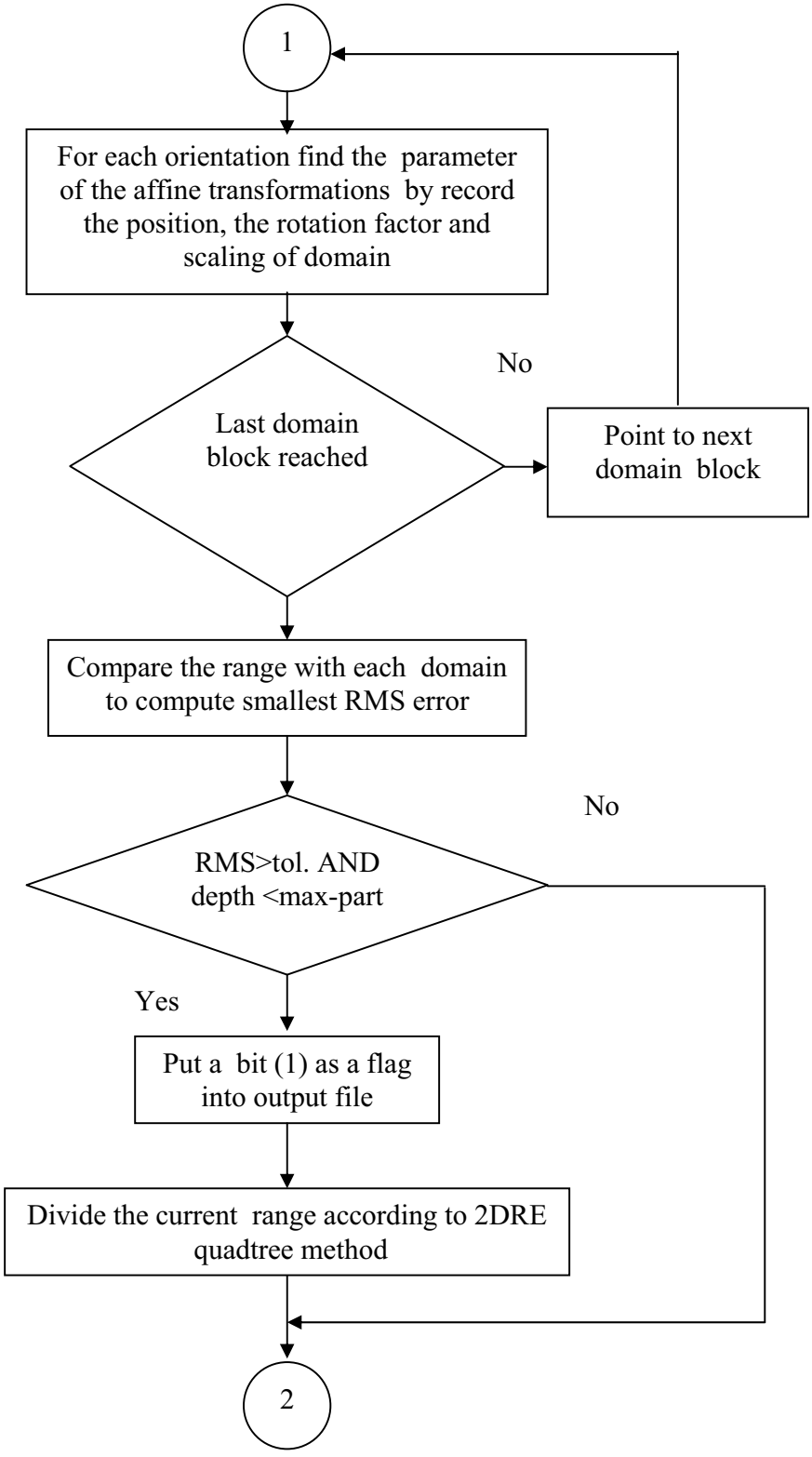
3-Fractal Image Compression Implementation And Evaluation:

The flowing flowcharts implement the compression and decompression algorithm, based on Fisher and Jacquin algorithm.

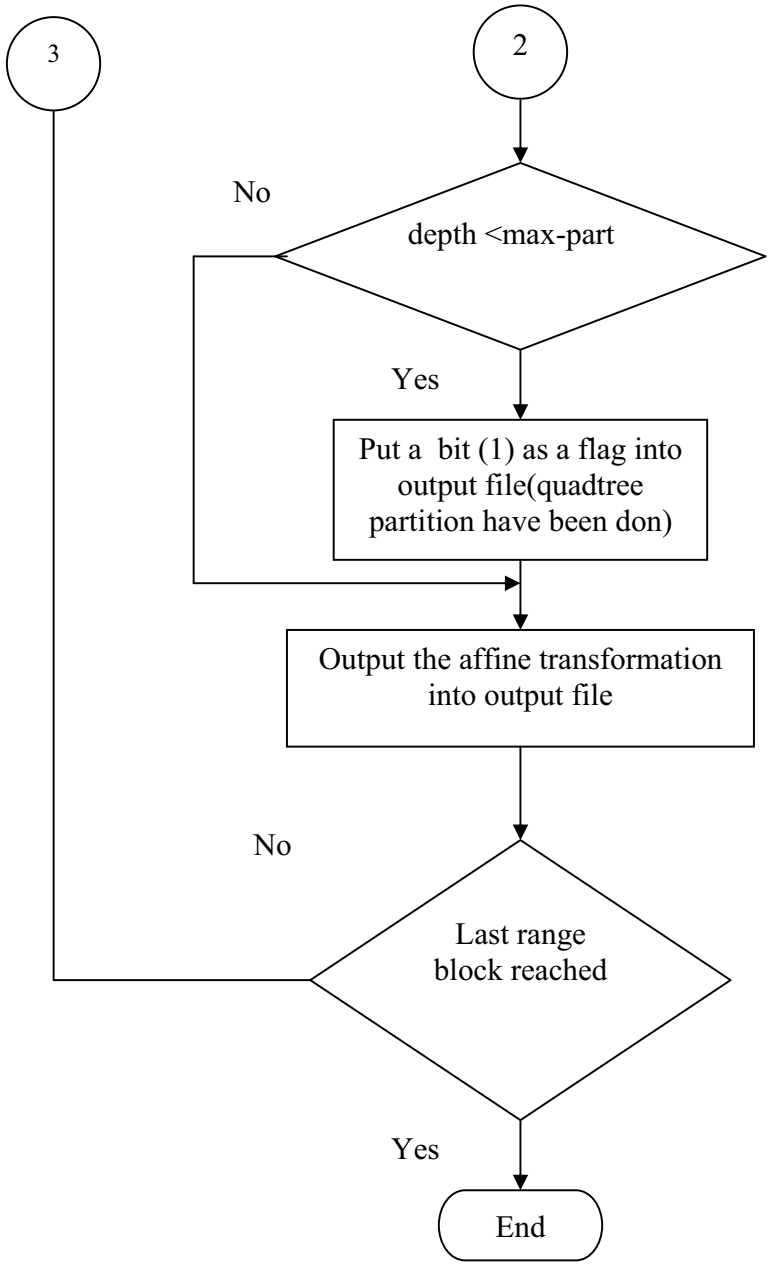
3-1 Encoding (Compressing) Algorithm: The basis for the encoding procedure is like this: an image is partitioned into parts that can be approximated by other parts after some scaling operations (affine transformations). The result of the procedure is a set of transformations, which, when iterated from any initial image, possess a fixed point approximating the original image.



Compression algorithm



Compression algorithm continued



Compression algorithm continued

3-2 Decoding (Decompressing) Algorithm:

Decoding an image consists of iterating *W* (recovered image) from any initial image. Fractal characteristics can be seen: 'zooming' into the restored image, finer and finer details will appear, the recovered image converges to the original image after a number of iterations (usually around five) see Figure 5.

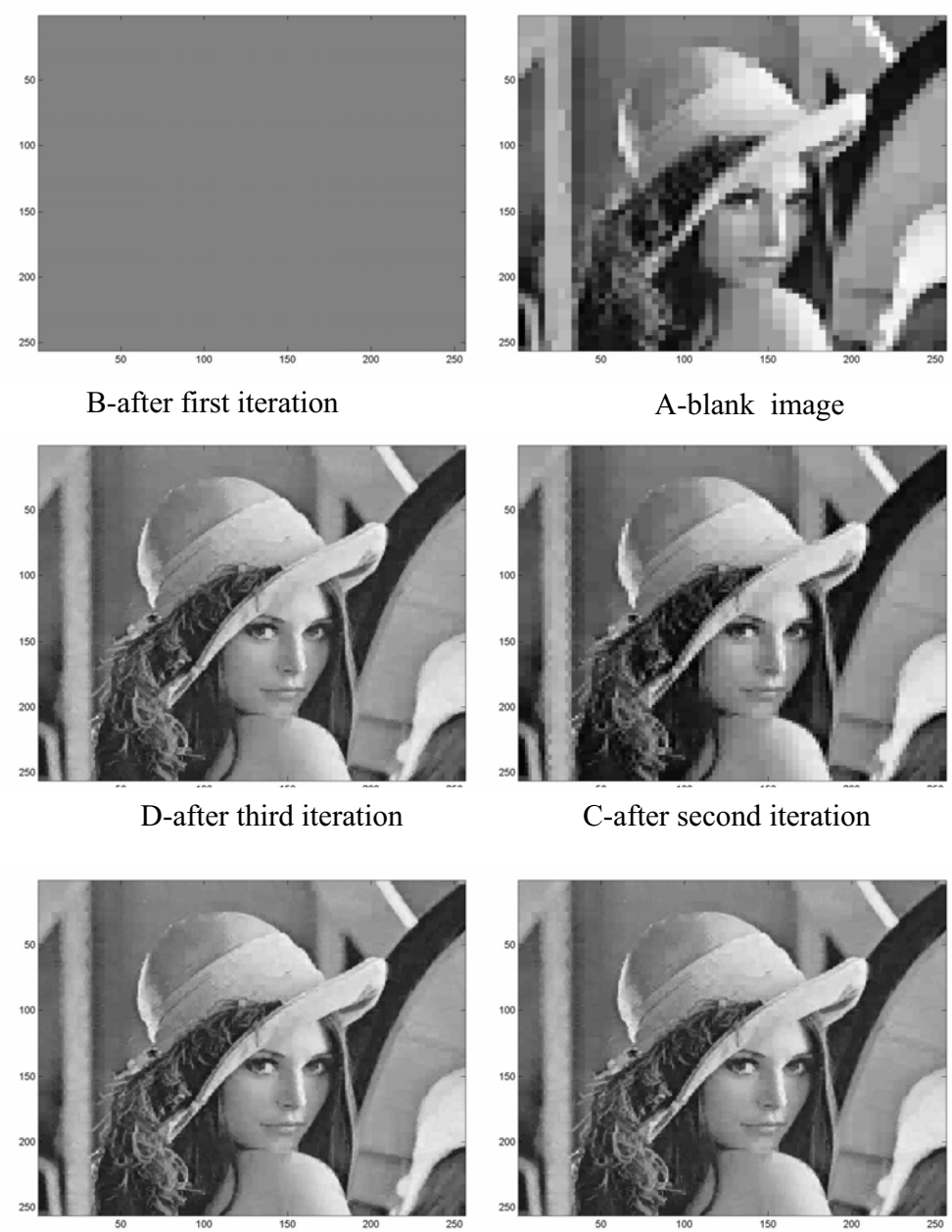
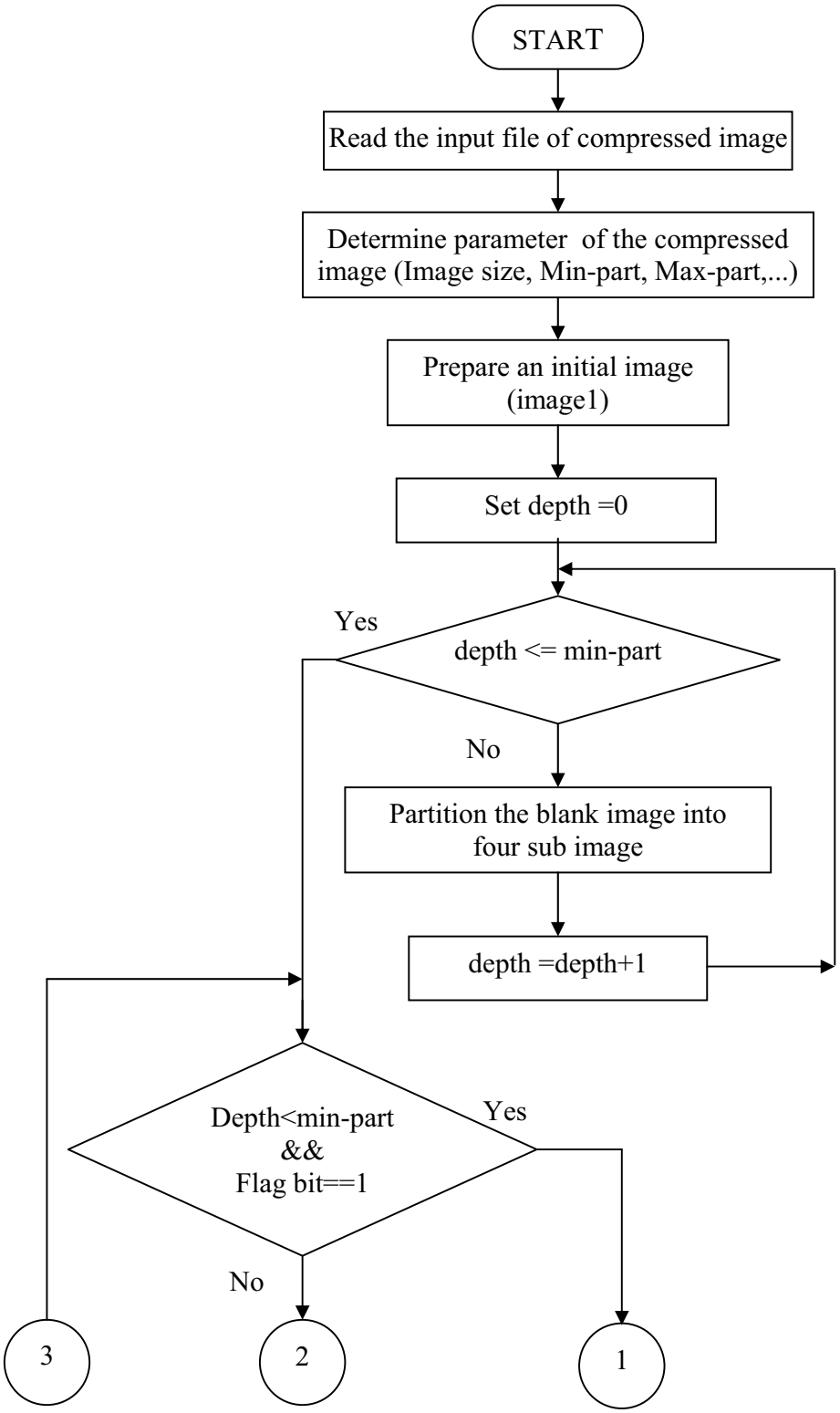
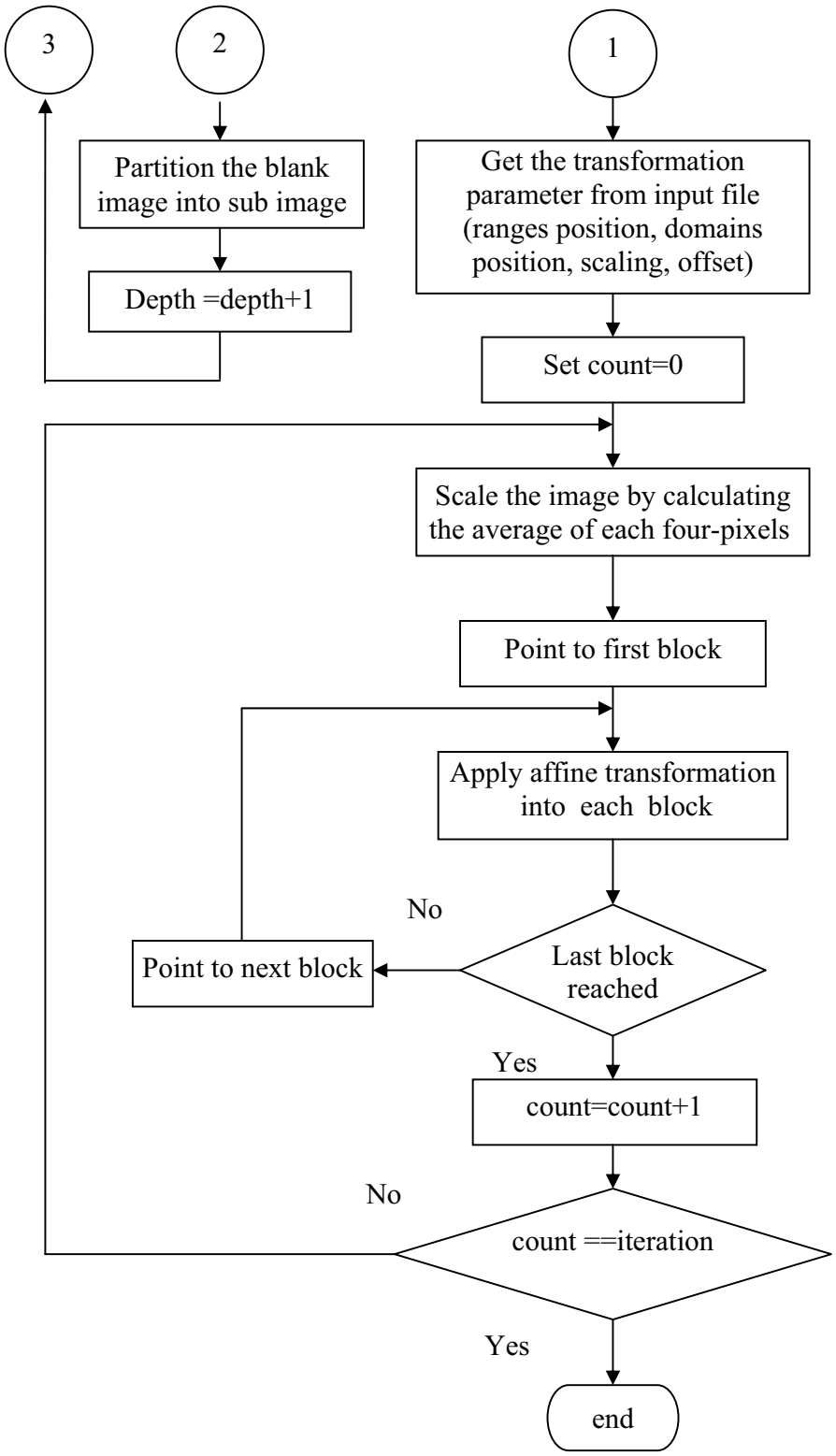


Figure 5 Image decompression iterations



Decompression algorithm



Decompression algorithm continued

3-3 Results:

The above algorithms are programmed using MATLAB 6.5 and the programs are executed on PC computer with CPU speed equal to 1.8 GHZ .

Table 1 describes the results produced from compression and decompression programs, the tol. (tolerance) implement the threshold (we divide the current range of image into four sub-ranges if error grater than tolerance) , min-error implements minimum acceptable error between current range and domain , MSE and PSNR implement peak signal to noise ratio and mean square error of decompressed image where:

$$\text{MSE}=\frac{1}{MN}\sum_{i-0}^{M-1}\sum_{j=0}^{N-1}\left[f'(i,j)-f(i,j)\right]^2$$

$$\text{PSNR}=10\log\left[\frac{255*255}{MSE}\right]\text{dB}$$

Where: M and N are image dimension

f' is decoded image, f is original image

Where:

1- Figure 6-a. illustrates original image of lenna (f).
2- Figure 6-b. shows the decoded image (f') of lenna 256*256, tol=20, miner=10, PSNR=27.7, CR=6.6.
3- Figure 6-c. shows the decoded image (f') of lenna 256*256, tol=50, miner=10, PSNR=27.6519, CR=10.569.
4- Figure 6-d. shows the decoded image (f') of lenna 256*256, tol=110, miner=10, PSNR=25.8852, CR=19.345.

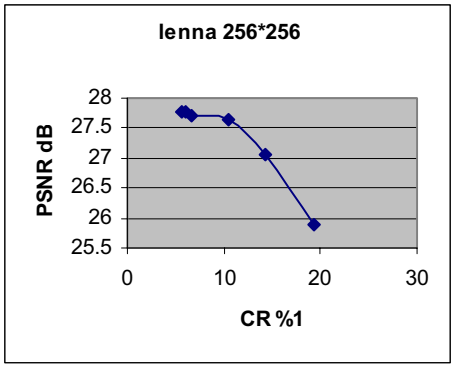
A-Original lena image. **B- Retrieved lena image CR=6.6**

Figure 6. Original and decompressed lena images

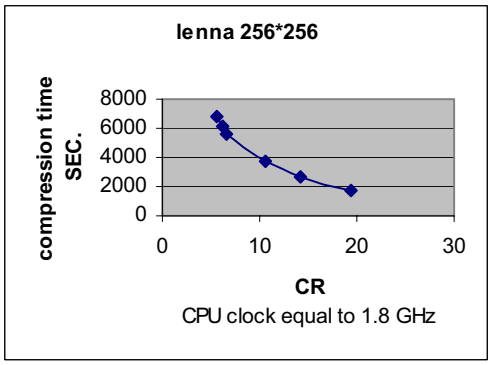
C- Retrieved lena image CR=10.56. D- Retrieved lena image CR=19.35.

Figure 6. Original and decompressed lena images (countued)

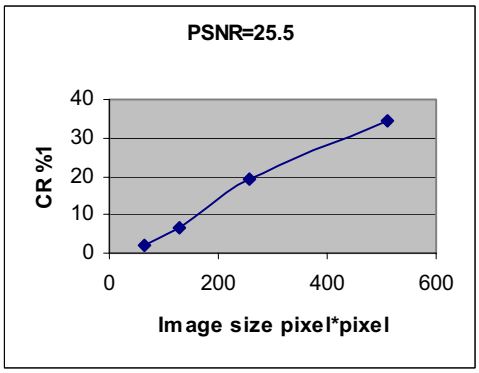
To study the variation of compression ratio (CR) with other parameters, Figure 6 describes the relationships of CR with PSNR, compression time and image size .



A



B



C

Figure 7 Relationship between CR and other parameters.

Table 1 Results for image compression and decompression parameters

Image size pixels^2	Min, Max(1)	tol.	min-error	MSE	PSNR dB	CR	Comp time sec	Deco. time sec
64	2-5	5	5	173.53	25.77	1.87	132	5.7380
64	2-5	10	10	176.67	25.69	1.92	105	4.7260
64	2-5	20	10	153.43	26.30	2.42	106	3.5650
64	2-5	30	10	163.89	26.01	2.88	114	2.9240
64	2-5	70	10	303.05	23.35	4.67	90	2.4430
64	2-5	100	10	353.67	22.68	6.73	46	2.4530
64	2-4	5	5	445.91	21.67	7.20	35	1.6020
64	2-4	10	10	446.60	21.67	7.20	57	1.6430
64	2-4	20	10	446.60	21.67	7.20	44	1.3920
64	2-4	40	10	447.94	21.65	7.54	40	1.3600
64	2-4	70	10	468.56	21.46	8.91	34	1.2320
64	2-4	100	10	494.07	21.23	10.37	30	1.1610
128	3-5	5	5	248.27	24.22	6.37	744	7.3810
128	3-5	10	10	248.83	24.21	6.37	541	6.1390
128	3-5	10	20	254.70	24.10	6.37	413	6.2390
128	3-5	10	40	275.18	23.77	6.37	260	6.0090
128	3-5	10	60	353.57	22.68	6.37	168	5.9880
128	3-5	20	10	249.05	24.20	6.67	549	6.3800
128	3-5	40	10	252.61	24.14	7.73	506	5.5980
128	3-5	60	10	258.89	24.03	8.5	471	5.1180
128	3-5	120	10	406.39	22.08	12.88	294	4.7470
256	4-6	10	10	109.35	27.78	5.60	6177	35.250
256	4-6	20	10	109.57	27.77	6.13	6861	26.137
256	4-6	30	10	111.2	27.70	6.68	4378	24.557
256	4-6	50	10	112.53	27.65	10.56	3731	19.067
256	4-6	80	10	129.22	27.05	14.25	2730	15.632
256	4-6	110	10	169.02	25.89	19.35	1739	14.360
512	4-6	80	10	183.62	25.52	34.68	12412	56.691
512	5-7	80	10	83.168	28.97	23.15	18205	58.675
512	5-7	50	10	59.539	30.42	14.60	34532	99.483

(1) Min., Max. are minimum and maximum quadtree partitioning respectively.

4-Conclusions

1- In this paper non overlapped image partitioning in domain is introduced. This reduces the processing time of Fisher and Jacquin algorithm.
2- Fisher used 32 values for performing the contrast scaling (between ranges and domains). In this paper the Fisher algorithm is modified to work with 4 values only with expense of a negligible effect noticed on the quality of the retrieved image.
3- Refer to figure 7-a, it is noticed that increasing the compression ratio by a factor of 10 reduces the PSNR by no more than 2dB.
4- The decompression operation starts with a blank image. The decompression algorithm uses the compression parameter of the original image together with a blank image or any other initial image to retrieve the original image refer to figure 5.
5- Five iterations or more are needed to decompress the image.
6- When tolerance increases the compression ratio and the compression /decompression time decreases.
7- When the size of the original image increases the compression ratio increase keeping the PSNR unchanged. However, both the compression and the decompression operation need more time to perform.
8- For a single original image, a higher compression ratio mean less PSNR and vice-versa.

References:

[1] John Kominek, " Advances in fractal compression for multimedia applications ", July 1997, Publisher :Springer-Verlag New York, Inc. Secaucus, NJ, USA, Volume 5, Issue 4, Pages: 255 – 270 , ISSN:0942-4962.

[2] D. Saupe, R. Hamzaoui, “A review of the fractal image compression literature“, November 1994, ACM Computer Graphics volume 28, Issue 4 , pages:268-276.

[3] B. Wohlberg and G. de Jager, “A Review of the Fractal Image Coding Literature,”, Dec. 1999, IEEE Transactions on Image Processing, vol. 8, no. 12, pp. 1716-1729.

[4] D. Saupe and R. Hamzaoui and H. Hartenstein, “ factal image compression: an introductory overview“,1996, siggraph '96 Course Notes , New Orleans.

[5] Takanori Yokoyama, and et-al, "Similarity-based Image Retrieval System Using PIFS Codes", 2003, Proc. of Artificial Life and Robotics, Beppu, Oita, Japan, pp.106-109.

[6] Lisa A. Soberano, " The Mathematical Foundation of Image Compression", May 2000, MSc. Thesis, The University of North Carolina at Wilmington.

[7] Yuval Fisher " Fractal Image Compression ",1992, siggraph `92 course notes, vol 12 pp 7.1-7.19.

[8] Xiao, Ke, "Fractal Compression and Analysis on Remotely Sensed Imagery" , 2003-01-17, Ph.D Dissertation, Louisiana State University and Agricultural and Mechanical College.

[9] J. Kominek, “Convergence of fractal encoded images”, Mar. 1995, in Proceedings DCC’95 (IEEE Data Compression Conference), (Snowbird, UT, USA), pp. 242–251.