Journal of Signal and Information Processing, **2013, 4, 176-181**
doi:10.4236/jsip.2013.43B032 Published Online August 2013 (http://www.scirp.org/journal/jsip)

Scientific Research

Study of Symmetry Process Behavior in Fractal Gray Image Compression by Traditional Method

Eman A. Al-Hilo, Kawther H. Al-Khafaji

College of Education for Girls, Physics Departments, Kufa University, Najaf, Iraq.
Email: emanalhilo@yahoo.com

Received June, 2013

ABSTRACT

This paper studies the effect of symmetry process on the compression parameters of the fractal image compression technique proposed by Jacquin. Two kinds of tests have been conducted. The first all kind of the symmetry operations [0-7] were taken; while the second tests were concentrated on studying the effect of the following parameters Block Size, Step Size, Domain Size on the probability distribution of symmetry operation. The results show that the higher value of PSNR and the lower value of ET occur at even symmetry operation only, but compression ratio is not affected with symmetry process. Also the occurrence probability of even symmetry is more than odd symmetry for all compression parameters. This behaviour can be utilized to reduce the encoding time to 50% with preserving PSNR.

Keywords: Image Compression; Zero-Mean; Fractal Image Compression; Symmetry process

1. Introduction

Image coding through fractal geometry has proven to be very effective, and fractal image compression is a promising field that has gained efficiency in both memory consumption and high speed transmission. Fractal image coding introduced by Barnsley and Jacquin is the outcome of the study of the iterated function system developed in the last decade [1]. By using IFS techniques and simple deterministic algorithms, images with spatial complexity can be encoded through certain fractal relationships that describe mapping of blocks of images within themselves. This fractal image technique finds similar patterns that exist on different scales/orientations, and at different places in an image. Hence, it allows encoding of an image file by systematically analyzing it and saving a much smaller set of instructions that can be used to iteratively reconstruct the entire image from those patterns. In fact, this compression technique can be thought of a method eliminating as much redundancy as possible. [2-3]

Saup [4] represents selected studies about the influence of symmetry on the quality of fractal codes. Afterwards an empirically comparison of the efficiency of domain pools with same size with or without using symmetry blocks follows. The presented results show that the using of symmetry doesn't increase the efficiency of codes.

The aim of our project is studding the behavior of symmetry process in the fractal gray image compression.

2. Fractal Image Coding

The basic idea of fractal image compression is partitioning the image into non overlapping range blocks. For every range block a similar but larger domain block is found. There are many ways to partition images. The fixed size partitioning are used in this research because it requires less computational time than the other.[5]

For a range block with pixel values ($r_o, r_1, \ldots, r_{m-1}$), and the domain block ($d_o, d_1 \ldots, d_{m-1}$) the contractive affine approximation is [6]:

$$r_i' = sd_i + o\,, \tag{1}$$

where, r_i' is the optimally approximated i[th] pixel value in the range block. d_i is the corresponding pixel value in the domain block. The symbols s,o represent the scaling and offset coefficients, respectively.

The scale (s) and offset (o) coefficients are determined by applying the least mean square difference (χ^2) criteria between (r') and (r) values [7].

$$\chi^2 = \frac{1}{m}\sum_{i=0}^{m-1}\left(r_1' - r_1\right)^2\,, \tag{2}$$

$$\frac{\sigma\chi^2}{\sigma s} = 0, \frac{\sigma\chi^2}{\sigma o} = 0, \tag{3}$$

The straightforward manipulation of the above equation leads to:

$$s = \frac{n\sum_{i=0}^{n-1} r_i d_i - \sum_{i=0}^{n-1} r_i \sum_{i=0}^{n-1} d_i}{n\sum_{i=0}^{n-1} d_i^{\,2} - \left(\sum_{i=0}^{n-1} d_i\right)^2}, \quad (4)$$

$$o = \frac{\sum_{i=0}^{n-1} r_i \sum_{i=0}^{n-1} d_i^{\,2} - \sum_{i=0}^{n-1} r_i d_i \sum_{i=0}^{n-1} d_i}{n\sum_{i=0}^{n-1} d_i^{\,2} - \left(\sum_{i=0}^{n-1} d_i\right)^2}, \quad (5)$$

$$\chi^2 = \left[\sum_{i=0}^{n-1} r_i^2 + s\left(s\sum_{i=0}^{n-1} d_i^{\,2} - 2\sum_{i=0}^{n-1} r_i d_i + 2o\sum_{i=0}^{n-1} d_i\right) + o\left(no - 2\sum_{i=0}^{n-1} r_i\right)\right] \quad (6)$$

3. Symmetry Process

In order to increase the size of domain pool and, consequently, to increase the probability of finding the best (near optimal) approximations for the range blocks, each domain block is transformed by using a set of symmetry transforms (i.e., rotation and flipping), such that eight versions (blocks) are produced for each domain block [8]. The eight symmetry mappings are (identity, rotation 90, rotation 180, rotation 270, reflection-x, reflection with rotation 90, reflection with rotation 180, reflection with rotation 270), (see **Table 1**). The number of bits required to represent the eight symmetry mapping cases is (3) bits.

Table 1. The eight symmetry transformations.

Sym	Equations	Results
0. Identity	$x' = x\cos(0) + y\sin(0)$ $y' = -x\sin(0) + y\cos(0)$	$x' = x$ $y' = y$
1. Rot.(+90)	$x' = x\cos(90) + y\sin(90)$ $y' = -x\sin(90) + y\cos(90)$	$x' = y$ $y' = -x$
2. Rot.(+180)	$x' = x\cos(180) + y\sin(180)$ $y' = -x\sin(180) + y\cos(180)$	$x' = -x$ $y' = -y$
3. Rot.(+270)	$x' = x\cos(270) + y\sin(270)$ $y' = -x\sin(270) + y\cos(270)$	$x' = -y$ $y' = x$
4. Ref. at x-axis	$x' = -x\cos(0) + y\sin(0)$ $y' = -x\sin(0) + y\cos(0)$	$x' = -x$ $y' = y$
5. Ref.& Rot. (90)	$x' = -x\cos(90) + y\sin(90)$ $y' = -x\sin(90) + y\cos(90)$	$x' = -y$ $y' = -x$
6. Ref.& Rot. (180)	$x' = -x\cos(180) + y\sin(180)$ $y' = -x\sin(180) + y\cos(180)$	$x' = x$ $y' = -y$
7. Ref & Rot. (270)	$x' = -x\cos(270) + y\sin(270)$ $y' = -x\sin(270) + y\cos(270)$	$x' = y$ $y' = x$

The main disadvantage of using the symmetry mappings in the encoding process is that more computational time will be required to perform the extra matching processes.

4. Encoding Process

The encoding method could be summarized by the following steps:

1) Load BMP image and put it in (2D arrays).

2) Establish the range image (array).

3) Down sample the range image to produce the domain array.

4) Great range and domain pool by partitioning:

a) The range array must be partitioned into non-overlapping fixed blocks, to generate the range blocks ($r_1,\ldots,r_n$).

b) The domain must be partitioned into overlapping blocks, using specific step size, to generate the domain blocks ($d_1,\ldots,d_n$). They should have the same size of range blocks.

5) Searching: For each range block do the following:

a) Pick up a domain block from the domain pool.

b) Perform one of the symmetry mappings that mentioned in table (1).

c) Calculate the scale (s) and offset (o) coefficient using equations (4) and (5).

d) Apply the following condition to bound the value of (s) and offset (o) coefficient:

If $s < s_{min}$ *then* $s = s_{min}$
Else if $s > s_{max}$ *then* $s = s_{max}$
If $o < o_{min}$ *then* $o = o_{min}$
Else if $o > o_{max}$ *then* $o = o_{max}$

e) Quantize the value (s) and offset (o) using equations that referred in [9].

f) Compute the approximation error (χ^2) using equation (6).

g) After the computation of IFS code and the sum of error (χ^2) of the matching between the range and the tested domain block, the (χ^2) is compared with registered minimum error (χ^2_{min}); such that:

If $\chi^2 < (\chi^2_{min})$ *then*
$s_{opt} = i_s$; $o_{opt} = i_o$, $\chi^2_{min} = \chi^2$
PosI=domain block index
Sym=symmetry index
End if

h) If $\chi^2_{min} < \varepsilon$ then the search across the domain blocks is stopped, and the registered domain block is considered as the best matched block.

i) Repeat steps (4) to (10) for all symmetry states of the tested domain block.

j) Repeat steps (3) to (11) for all the domain blocks listed in the domain pool.

k) The output is the set of IFS parameters

$(i.e., i_s, i_o, posI, Sym)$ which should be registered as a set of fractal coding parameters for the tested range block.

l) Repeat steps (1) to (12) for all range blocks listed in the range pool

m) Store all IFS mapping parameters as an array of record. The length of this array is equal to the number of range blocks in the range pool.

5. Decoding Process

The decoding process can be summarized by the following steps:

1) Generate arbitrary the domain pool, the domain pool could be initialized as a blank image or a piece of image extracted from any available image.

2) The values of the indices of (i_s) and (i_o) for each range block should be mapped to reconstruct the quantized values of the scale (s_q) and offset (o_q) coefficients.

3) Choose the number of possible iterations, and the threshold value of the mean square error (TMSE). At each iteration, do the following steps:

a) For each range block determine the coordinates (x_d,y_d), of the best matched domain, from the IFS parameters (*posI*), in order to extract the domain block (*d*) from the arbitrary domain image.

b) For each range block, its approximation r_i' is obtained by multiplying the corresponding best matched domain block (d) by the scale value (s_q) and adding to the result the offset value (o_q), according to equation (1).

c) The generated r_i' block is transformed (rotated, reflected, or both) according to its corresponding IFS symmetry parameter value (*Iso*).

d) Put the generated r_i' block in its position in the decoded image array (i.e., range image).

e) Check whether there is another range block, if yes then repeat steps (b,c,d)

f) Down sample the reconstructed image (range pool) in order to produce the domain pool using the averaging sampling.

g) Calculate the mean square error MSE between the reconstructed range and the previous reconstructed range image. If the MSE is greater than TMSE value then the iteration continues and the above steps (a-f) should repeated; this iteration is continued till reaching the attractor state (i.e., the newly reconstructed range image is very similar to the previous reconstructed image). Otherwise the iteration continues till reaching the predefined maximum number of iterations.

6. Tests Results

The proposed system was established using Visual Basic (Ver.6.0) and tested on Aser laptop with (2.20GHZ, RMA 956MB)

The proposed system had been tested on Lena image (256x256 pixels, 8 bits). The value of the parameters MaxOffset and MinOffset were fixed in all these tests at (255) and (-256) respectively but the other coding parameters were taken as: BlockSize=(4x4), StepSize=2, DomSize= (128x128), ScaleBits=6, OffsetBits=6, MinScale=-1.5, MaxScale=3, $\mathcal{E}$ =0.4,TMSE=0.05. These tests were taking two aspects:

6.1. Symmetry Tests

These tests were conducted to investigate the effect of each symmetry operation, subset [0-3], subset [4-7], and full symmetry [0-7], on compression performance parameters. **Table 1** shows these effects on the compression parameters MSE, RMSE, PSNR, CR and ET.

Table 1. Effect of each, subsets, and full Symmetry on the compression performance parameters.

Sym	MSE	RMSE	PSNR	CR	ET(sec)
0	31.73	5.63	33.12	4.741	11.97
1	40.60	6.37	32.05	4.741	12.38
2	31.71	5.63	33.12	4.741	12.14
3	39.21	6.26	32.19	4.741	12.37
4	33.09	5.75	32.93	4.741	12.03
5	38.60	6.21	32.26	4.741	12.38
6	32.73	5.72	32.98	4.741	12.01
7	38.39	6.19	32.29	4.741	12.38
Subsets and Full Symmetry Operation					
(0-3)	25.16	5.02	34.12	4.741	45.42
(4-7)	24.39	4.94	34.26	4.741	45.49
(0-7)	21.79	4.67	34.75	4.741	89.67

The results show that best PSNR occur when all symmetry operations are used. The symmetry (0) appears higher PSNR and lower ET among (8) symmetry operation. CR is not affected by symmetry operation.

Figure 1 shows the effects of each symmetry operation alone on the compression performance parameters MSE, RMSE, PSNR, and ET respectively.

The results show that the value of MSE, RMSE and ET appear lower at symmetry operations (0,2,4,6) with respect to their values at symmetry (1,3,5,7), But PSNR appears higher at symmetry operations (0,2,4,6) with respect to their values at symmetry operations (1,3,5,7). This indicate that PSNR is behaves inversely with ET.

Figure 2 shows a set of the reconstructed images when no symmetry (identical) and full symmetry operations were applied**.**

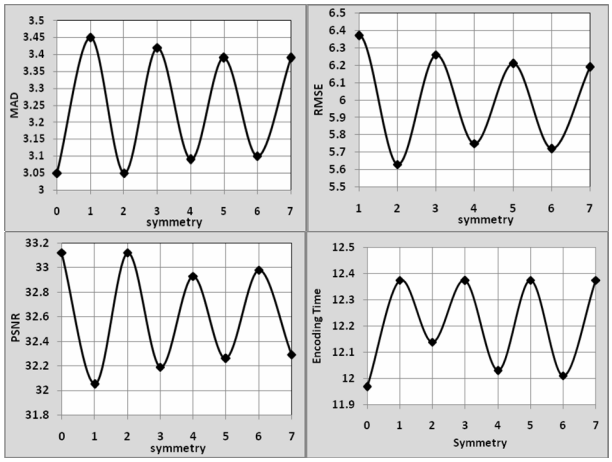


Figure 1. Effect of each Symmetry operation on MSE, RMSE, PSNR and ET respectively.

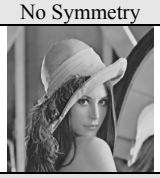
Original	No Symmetry	Full (0-7)
		
PSNR	33.12	34.75
ET	12.36	89.67

Figure 2. Effects of different symmetry on the compression performance parameters.

6.2. Symmetry Distribution Tests

In these tests, the numbers of blocks that have the same Symmetry status from [0-7] were counted. The results of these tests are useful to know the most redundant symmetry case, which is useful to reduce the consumed encoding time.

These tests were concentrated on studying the effect of the following parameters:

1) *Block Size:* In this test set different BlockSize values (i.e., 2x2, 4x4, 8x8, 16x16) were taken into consideration, while the values of the StepSize = 2, DomSize = (128x128) were kept fixed. **Table 2** shows the effect of BlockSize parameter on symmetry distribution.

The results show that the occurrence probability of even symmetry increases with the increase of BlockSize value. The highest probability is (76%) at BlockSize (16x16)

Figure 3 shows the effect of BlockSize on the symmetry distribution. The results show that the even states (0,2,4,6) have the highest populations of symmetry.

2) *Step Size:* This subset of tests investigates the effect of different StepSize on symmetry distribution. In this test, different values of StepSize (1,2,3,4) were taken and the values of other coding parameters were fixed at BlockSize = (4x4), DomSize = (128x128). **Table 3** illustrates the effect of StepSize on symmetry distribution.

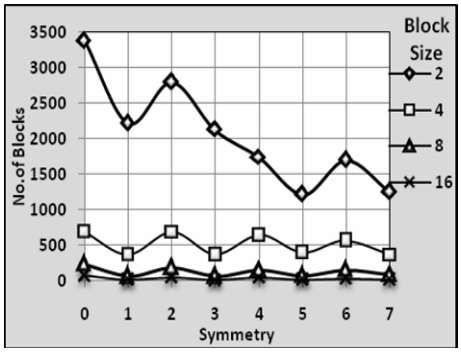


Figure 3. BlockSize effect on the symmetry distribution.

Table 2. Effect of BlockSize on the Sym distribution.

Sym	(BlockSize)			
	2x2	4x4	8x8	16x16
0	3369	689	236	74
1	2209	366	65	15
2	2792	685	183	45
3	2125	371	66	11
4	1735	648	154	45
5	1210	404	75	17
6	1698	568	156	32
7	1246	365	89	17
Tot	16384	4096	1024	256
Even%	58.56%	63.23%	71.19%	76.56%
Odd%	41.44%	36.77%	28.8%	23.44%

Table 3. Effect of StepSize on the symmetry distribution.

Sym	(StepSize)			
	1	2	3	4
0	735	689	725	686
1	365	366	368	380
2	665	685	674	674
3	369	371	371	366
4	620	648	578	598
5	401	404	405	399
6	589	568	602	623
7	352	365	373	370
Tot	4096	4096	4096	4096
Even%	63.69%	63.23%	62.96%	63.01%
Odd%	36.3%	36.76%	37.04%	36.99%

The results show that the probability of occurrence of even and odd symmetry is not affected by the variation of StepSize value. The highest probability at even symmetry is (63.69%) at StepSize (1) and at odd symmetry is (37.04) at StepSize (3).

Figure 4 shows the effect of StepSize on symmetry distribution.

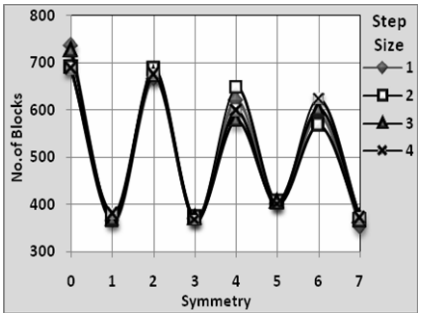


Figure 4. StepSize effect on the Symmetry distribution.

3) *Domain Size*: This subset of tests was conducted to investigate the effect of DomSize on the symmetry distribution. The value of DomSize was varied to (128x128, 64x64, 32x32, 16x16), while the values of other coding parameters were taken at BlockSize = (4x4), StepSize = (2). **Table 4** shows the effect of DomSize on symmetry distribution.

Table 4. Effect of DomSize on the Symmetry distribution.

Sym	(DomSize)			
	128	64	32	16
0	689	676	697	855
1	366	353	333	312
2	685	693	728	720
3	371	355	337	300
4	648	624	639	617
5	404	359	346	315
6	568	679	639	622
7	365	357	377	355
Tot	4096	4096	4096	4096
Even%	63.23%	65.23%	65.99%	68.70%
Odd%	36.77%	34.77%	34.01%	31.29%

The results show that the probability of occurrence of even symmetry increases with the increase of DomSize value, but at odd symmetry decrease with the increase of DomSize value. The highest probability is (68%) at DomSize (16x16) at even symmetry and (36.77%) at DomSize (128x128) at odd symmetry. **Figure 5** shows the effect of DomSize on symmetry distribution.

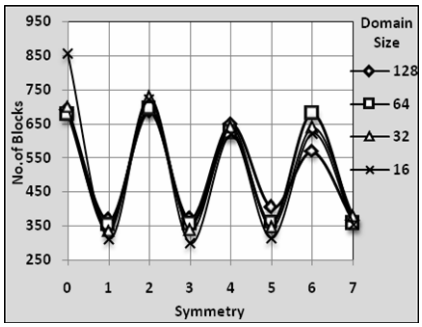


Figure 5. DomSize effect on the symmetry distribution.

7. Conclusions

From the above listed results could be concluding the following:

1) The higher value of PSNR at even symmetry and the lower value of ET at even symmetry also, but CR is not affected with symmetry process.

2) The Most probable symmetry state is (0: identical symmetry) for all compression parameters (BlockSize, StepSize and DomSize).

3) The occurrence probability of even symmetry (0,2,4, 6) is more than odd symmetry

The high probability of occurrence of even symmetry could be utilized as an efficient way to reduce the encoding time in matching process. So, instead of using (8) symmetry operations one can use only the even symmetry operations (2,4,6,8). In such a case, the expected reduction in encoding time will be around 50%.

REFERENCES

[1] A. Jacquin, "Fractal Image Coding a Review," *Process ding of the IEEE*, Vol. 81, 1993, pp. 1451-1465.

[2] M. F. Barnsley, "Fractals Everywhere," New York Academic, second edition.

[3] A. Kapoor, K. Arora, A. Jain and G. P. Kapoor, "Stochastic Image Compression Using Fractals," *International conference on Information Technology: Coding and Computing (ITCC 2003)* Las Vegas, Nevada, Pg. 574-579, 28-30 April 2003.

[4] D. Saupe, "The Futility of Square Isometries in Fractal Image Compression," *IEEE Int. Conf. On Image Processing (ICIP96)*, Lausanne, Sept. 1996.

[5] M. Schebe, "Square Isometries as Integer Part of Fractal Transformation- An Analysis ," FREQENZ,12 De,1996.

[6] Y. Fisher, "Fractal Image Compression Theory and Application," University of California, Institute for Nonlinear Science, Springer-Verlay, New York, Inc, 1995.

[7] Y. Fisher, "Fractal Image Compression," SIGARAPH 92 Course Notes, the San Diego Super Computer Center,

University of California, an Diego, 1992

[8] C. Frigaard, "Fast Fractal 2D/3D Image Compression, Report," Institute of Electronic Systems, Alborg University, Laboratory of Image Analysis, 1995.

[9] E. A. Al-Hilo, "Loay E. George, "Fractal Color Image Compression," *4th International Conference on Nov. 2008 (ICIMU' 2008)*, Malaysia.

