Research Journal of Applies Sciences, Engineering and Technology 3(8): 757-764, 2011
ISSN:

Received: May 24, 2011 Accepted: July 02, 2011 Published: August 30, 2011

# Two Suggested Methods for Faster Fractal Image Compression

[1]Taha Mohammed Hasan, [1]Xiangqian Wu and [2]Eman Abdul-Jabar Saad
[1]School of Computer Science and Technology, Harbin Institute of Technology,
Harbin 150001, Heilongjiang, China
[2]Electronic Computer Center, Mustansiriyah University, Baghdad, Iraq

**Abstract:** In this study, two methods have been suggested for breaking the long encoding time problem of the encoding process in Fractal Image Compression (FIC). The First, called Zero Mean Intensity Level (ZMIL), which is based on using an unconventional affine parameter (namely the range block mean) that has better properties than the conventional offset parameter. As result, it is found that ZMIL gives a high value of compression ratio (around 18.9% additional compression value), with a high reconstructed image quality and a reduction in the encoding time of about 27% in comparison with the traditional FIC. The second, called the speed up (ZMIL) method which is responsible for the reduction of the number of domain blocks needed to be IFS matched with tested range block by eliminating the symmetry orientations and Reduction Domain Image Size (RDIS). From the results of this method, the encoding time will be reduced to only one second (i.e., reduction about 95% of the time required in the full cases), with a higher increase in the value of compression ratio and still has a good reconstructed image quality (PSNR).

**Key words:** Encoding time, fractal, image compression, partition, range block

## INTRODUCTION

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. Banach's fixed point theorem guarantees that, within a complete metric space, the fixed point of such a transform may be recovered by iterated application thereof to an arbitrary initial element of that space (Hamzaoui, 2000). Encoding is not so simple, since there is no known algorithm for constructing the transform with the smallest possible distance, given the constraints on the transform, between the corresponding fixed point and the image to be encoded. 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 transform in terms of the distance between transform of the image and the image itself. Sub optimal transform may be constructed as a "collage" or union of mappings from the image to itself, a sufficiently small "collage error" guaranteeing that the fixed point of that transform is close to the original (Ruhl, 1997). Fractal Image Compression (FIC) really became practical with the introduction given by Jacquin of the partitioned IFS (i.e., PIFS), which differs from an IFS in that each of the individual mappings operates on a subset of the image, rather than the entire image (Fisher, 1995). FIC is to find a loss compression algorithm that takes the advantage of the self-similarities in an image. Barnsley applied his knowledge of fractal and mathematics to image compression, creating optimal forms of image compression (Radha, 2004; Kadhim, 2001). The encoding process of fractal image compression is extremely computationally intensive (long encoding time). This weak aspect makes the fractal compression method still not widely used as standard compression, although it has the advantage of fast decompression as well as gets high values of compression ratios. According to the fractal coding algorithm that was suggested by Jacquin, the image is partitioned into non-overlapping blocks ($R$), where each block will be transformed separately using affine transform. The same image also houses blocks, which are twice the size of the range blocks and overlap. This collection of all large blocks known as domain block ($D$), constructs a codebook called domain pool (Ω) (Lisa, 2000). After partitioning a given image into R-blocks and D-blocks should be found pieces of $D_j$ and a collection of contractive maps: $W_1, W_2, \ldots, W_n$

$$Wi\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a\ b_i\ 0 \\ c\ d_i\ 0 \\ 0\ 0\ s_i \end{bmatrix} . \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ o_i \end{bmatrix} \quad (1)$$

So that when $w_i$ applied to the part of the image over $D_j$, should get something that is very close to any part of the image over $R_i$. The crux, then of the encoding image

**Corresponding Author:** Taha Mohammed Hasan, School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, Heilongjiang, China. Tel.: 0086145073381

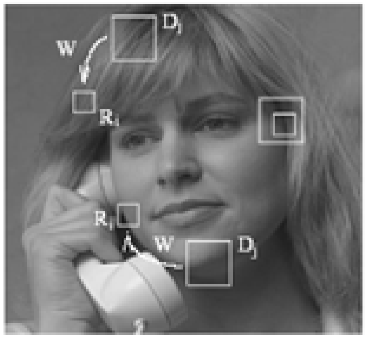

Fig. 1: The transform between domain block ($D_j$) and range block ($R_i$)

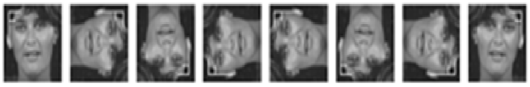

Fig. 2: The eight spatial orientations (symmetry) of a square block

is to find contractive maps $w_i$ that minimize the distances between $R_i$ and corresponding $D_j$ as illustrated in Fig. 1 (Lisa, 2000).

This shows why fractal compression is a slow technique, since each range block must be compared to all domain blocks including eighth symmetry orientations (Fig. 2). This operation allows the best match to be found, the best matching between domain and range blocks which is satisfy the minimum distortion error $E(R, D)$ of Eq. (2) (Farhad, 2001).

$$E(R < D) = \frac{1}{n}\sum_{i=1}^{n}(S.d_i + o - r_i)^2 \tag{2}$$

In other words, we seek to minimize the quality of distortion over $D \in \Omega$ in Eq. (3) with respect to the parameters scale ($s$) and offset ($o$) in Eq. (4) and (5) respectively (Fisher, 1995). Where,

$$E(R,D) = \frac{1}{n}\left[\sum_{i=1}^{n} r_i^2 + S\left(S\sum_{i=1}^{n} d_i^2 - 2\sum_{i=1}^{n} d_i r_i + 2O\sum_{i=1}^{n} d_i\right) + O\left(nO - 2\sum_{i=1}^{n} r_i\right)\right] \tag{3}$$

$$S = \frac{\left[n\left(\sum_{i=1}^{n} d_i r_i\right) - \left(\sum_{i=1}^{n} d_i\right)\left(\sum_{i=1}^{n} r_i\right)\right]}{\left[n\sum_{i=1}^{n} d_i^2 - \left(\sum_{i=1}^{n} d_i\right)^2\right]} \tag{4}$$

and

$$o = \frac{\left[\sum_{i=1}^{n} r_i \sum_{i=1}^{n} d_i^2 - \sum_{i=1}^{n} d_i \sum_{i=1}^{n} d_i r_i\right]}{\left[n\sum_{i=1}^{n} d_i^2 - \left(\sum_{i=1}^{n} d \quad _i\right)^2\right]} \tag{5}$$

## MATERIALS AND METHODS

**Zero-Mean Intensity Level (ZMIL) method:** Various techniques have been proposed in fractal encoding to overcome the large amounts of time needed during the searching process (matching) between the domain-range pairs. This study introduces the transforms of the full search problem using a more convenient form by adopting an unconventional affine parameter that has better properties than the conventional offset parameter and a new search algorithm has been developed. In fact, this idea was advocated by Oien and Lepsoy and also implicity used by Bani-Egbal and it was applied by Tong and Pi (2001). And in this research it was used and developed by merging it with other speeding up techniques to get hi-performance method. As mentioned in the encoding unit, the optimal approximation for every range block must be obtained from Eq. (3) and with respect to the affine parameters *s* and *o*. In the traditional choice of using the *o* in Eq. (3) and many calculations that required to computes this parameter as described in Eq. (5), the full search scheme can also be converted to a one-parameter optimization problem. The question here is: "how does the two parameters problem become a one-parameter optimization, and how to perform the new search operation? The answer for this question can be illustrated in the following:

The optimal approximation in decoding unit for every range block can be obtained from Eq. (6).

$$\bar{r} = s\bar{d} + o \tag{6}$$

where,

$\bar{r}$ represents the average (mean) for a specific range block

$\bar{d}$ represents the average (mean) for the mapped domain block for this range block

It is noted from Eq. (6) *o* coefficient given by the difference of the range block mean $\bar{r}$ and the scaled domain block mean $\bar{d}$ (Tong and Pi, 2001), using this idea to formulate a new affine transform Eq. (7):

$$r_i = s\left(d_i - \bar{d}\right) + \bar{r}\,\forall\, i \tag{7}$$

As shown in Eq. (7), the fractal parameter $\bar{r}$ is used instead of the conventional *o* coefficient. So the new parameters are:

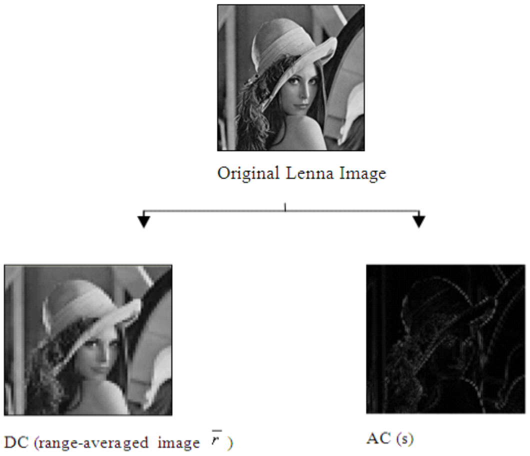

Fig. 3: Show the DC and AC-components of the original Lenna image

- $\bar{r}$ which is the (DC-component) of the range block and it is independent of the domain block.
- s which is related to the (AC-component) of the range block

Interestingly, the new transformation splits an image into DC and AC, as illustrated in Fig. 3. Clearly, using $\bar{r}$ as one of the affine parameters instead of the *o* allows de-coupling of the optimization of the two affine parameters and thus furthering speed up the search for the best matching domain block (Lee, 2000; Tong and Pi, 2001). Moreover, from Eq. (3), notice that *s* and *o* are strongly correlated in the traditional transformation. Hence, dependence complicates the quantization for *s* and *o*. Within the introduced transformation, the alternate fractal parameters *s* and $\bar{r}$ are independent, therefore separable quantization can be done for *s* and $\bar{r}$ . Thus it is more efficient to quantize $\bar{r}$ , especially as it has a much smaller dynamic range [0, 255], than the *o* parameter [-255, 255] (Tong and Pi, 2001). So, it is more cost effective (in terms of minimizing the quantization error per code) to code the quantization of $\bar{r}$ than to code that of the *o* parameter. In the introduced transformation, $\bar{r}$ is uniformly quantized by 6 bits and *s* is uniformly quantized by 2 bits. So, by substituting Eq. (7) in Eq. (2), the distortion error will be in Eq. (8) and (9):

$$E(R,D)=\frac{1}{n}\sum_{i-1}^{n}\left(s\left(d_i-\bar{d}\right)-\left(r_i-\bar{r}\right)\right)^2 \tag{8}$$

$$E\left(R',D'\right)=\frac{1}{n}\sum_{i-1}^{n}\left(sD^t-R^t\right)^2 \tag{9}$$

where; $R' = r_i - \bar{r}, \qquad D' = d_i - \bar{d}$

So, this method will depend on this alternative equation of distortion meaning that the range-domain blocks have been adjusted to a Zero Mean Intensity Level (ZMIL) by subtracting the mean $\bar{r},\bar{d}$ from all the range-domain blocks. Moreover, the $\bar{r}$ can be quantized and coded at the start of the compression to provide a coarse level of compression and hence it can be used in the design of a progressive fractal compression. Minimizing Eq. (9), one can solve for *s*. This can be achieved by taking the derivative of E(R',D')with respect to *s* as zero, i.e.,

$$\frac{\partial E\left(R',D'\right)}{\partial S}=\frac{-2}{n}\sum\left(SD^t{}_i-R^t{}_i\right)D^t{}_i=0$$

$$\sum\left(R^t{}_iD^t{}_i\right)=S\sum D^{t2}{}_i \tag{10}$$

$$\therefore S=\frac{\sum R^t{}_iD^t{}_i}{\sum D^{t2}{}_i}$$

A significant improvement in fidelity can be obtained if the quantized *s* value is used only when the err E(R',D') is computed during encoding but the drawback is that post-quantization of *s* often leads to a poorer result as compared with pre-quantization, where the quantity that we actually want to minimize is:

$$E(R,D)=\frac{1}{n}\left[s^2\sum D^{t2}{}_i+\sum R^{t2}{}_i-2s\sum R^t{}_iD^t{}_i\right] \tag{11}$$

and the root mean square error is equal to $\sqrt{E\left(R',D'\right)}$ . In this prosed method, the search for the best matching domain block only depends on the quantized scaling and the inner product <R',D'>, and is independent of the $\bar{r}$ . This is because the $\bar{r}$ depends only on the range block and thus the parameter can be quantized independent of the search for the best matching domain block. This quantization does not involve searching at all. Thus effectively, two parameters optimization problem becomes a one-parameter optimization, and hence it is much more efficient (Tong and Pi, 2001).

**Encoding algorithm based on ZMIL method:** To encode an image, it must be partitioned into non overlapping range blocks $R_i$. For every range block, a similar but larger domain block is found. There are many ways to partition images, the partitioning used in this research is a fixed size partitioning scheme, because it requires less computational time than the other methods. The domain blocks will be created by down sampling the original image by averaging method. Then the matching process will be started. In ZMIL method, the encoding process at first subtracts the mean of range *R* and domain *D* blocks then the matching process will be done for *R*` to find the best *D*`. The overall best matching blocks are obtained by minimizing the weighted $E(R^t.D^t)$ over the quantized level for $s_i$. Thus Eq.(11) leads to the following encoding algorithm that illustrated ZMIL encoding steps.

### Algorithm

```
Input   : The original image
Output  : The IFS code
Method:
Step1   : Load the image into buffer
Step2   : Partitioning the image into fixed blocks size
          with non- overlap ($R_1$… $R_n$)
Step3   : Generate the domain image from the original
          image by the averaging method
Step4   : Build a new domain blocks $D^t{}_1 ... D^t{}_m$
Step5   : For each range block $R_i$ build a new $R'_i$ and
          do:
          Quantize $\bar{r}$
          Check all the $D^t{}_m$ for the best matching
          $D^t{}_i$ by:
          Compute the $S = \frac{\sum R^t{}_i D^t{}_i}{\sum D^{t2}{}_i}$
          Quantize the $s$;
          Compute $E(R'_i, D'_i; s)$;
          If $E(R'_i, D'_i, s)$ is minimum; store the IFS-
          code; else go to the next domain block;
```

**Decoding algorithm based on ZMIL method:** The decoding algorithm based on ZMIL method, which replaces *o* parameter with $\bar{r}$ , and the assumption that domain block mean $\bar{d}$ could be obtained from the reconstructed image. Since $\bar{d}$ is not a fixed value, but a variable depending on a domain block *D*, it may be difficult to recover in practice. Based on this observation, the transformation *W* can be modified as follows:

$$W\left(\overline{d_{ij}}\right) = \bar{r} + s*\left(d_{ij} - \sum_{m=1}^{b}\sum_{n=1}^{b} d_{mn} / B^2\right) \qquad (12)$$

Modifying transformation can be written as:

$$\text{W= DC+S(AC)} \qquad (13)$$

As we see, for the new transformation *W*, parameter $\bar{r}$ is the *DC* of the range block and is independent of the domain block; *s* is related to the *AC* of the range block and is used to refine the range block iteratively. At the decoder, the reconstructed image is generated by recursive iterations on the basis of an arbitrary initial image. In the new iteration algorithm, $\bar{d}$ varies as iteration proceeds. The first decoded image will be the range-averaged image (*DC*) and hence the uses of the range-averaged image as an initial image for the next iterations will cause a faster converge to the attractor. As the iterations proceed, the *AC* components will be added to the *DC* component, after 2 iterations, the reconstructed image becomes stable. The following algorithm explains the decoding of ZMIL method.

### Algorithm

```
Input   : The IFS code
Output  : The decoded image
Method:
Step1   : Generate the first domain image arbitrary
Step2   : Determine the iterations number
Step3   : Load IFS code
Step4   : Dequantize the value of scale $s_i$ and range
          block mean $\bar{r}k$
Step5   : Build a new domains blocks $D'_1$ ... $D'_m$
Step6   : Reconstruct the range block: $R_i = sD'_i + \bar{r}$
Step7   : Each range block is reconstructed will be
          located in its position in the decoded image
          plane.
Step8   : Down sample the decoded (reconstructed)
          image into the size of domain image by the
          averaging method
Step9   : Repeat from step 5 until the attractor state is
          reached (i.e., decoded image will not be
          changed with farther iterations)
```

**Speed up ZMIL method:** As mentioned previously in FIC, the encoding process is computationally intensive. A large number of sequential searches through a list of domains are carried out while trying to find a best match for a range block. The number of possible domain blocks is huge in comparison to the number of range blocks. In ZMIL method, each block transform is identified, normally, by five values, that need 21 bits: 3(bits for symmetry)+5(bits for *x*)+5(bits for *y*)+2(bits for *s*)+6(bits for $\bar{r}$ ), therefore encoding the original image will require (4096 × 21= 86,016 bits) = (10,752 bytes) = (10.5 KB). A large domain pool will increase the number of comparisons that have to be made to find the best domain block and this where most of the computing time is used. The ZMIL method significantly reduces the encoding time, also, in this section, investigation of how the search time can be decrease to speed up fractal encoding based on ZMIL method is performed. The key of the idea is to reduce the number of domain blocks searched for each range block. This can be done by the following ways:

- **Symmetry orientations:** When reducing or removing symmetry orientations (i.e., taking only the identical case and its orientations (half symmetry) or removing all the symmetry cases (no symmetry)) from the encoding process, a considerable increases will happen in the compression ratio, a significant increase will happen in the speed of the encoding process, with only little decreases in the PSNR. Reducing the symmetry from full (8 cases of each domain to be searched) to half (4 cases), one bit from each range block transform will be reduced, because only four cases of each domain will be searched and only two bits will be required instead of 3 bits which are required to store the 8 cases of full symmetry, if this bit is given to the scale parameter, then, in this case the compression ratio and the quality will be preserved and more reduction will be gotten in the encoding time. But if all symmetry cases is eliminated, in this case one bit from the 3 bits that are required in full symmetry case can be given to the scale parameter trying to keep the image quality and 2 bits will be reduced from each range block transform and this will increase the compression ratio, for example, the transform after eliminating the symmetry will be identified by 19 bits; 5(bits for *x*) + 5(bits for *y*) + 3(bits for *s*) + 6(bits for $\bar{r}$ ). So encoding the original image will require (4096×19 = 77,824 bits) = (9,728 bytes) = (9.5 KB), therefore, the value of compression ratio will be increased. Also, the computations needed in the encoding process after eliminating the symmetry will be reduced to (4096×1024) = 4,194,304 operations instead of (4096×1024×8) = 33,554,432 operations and this will reduce the encoding time significantly.

- **Reduction Domain Image Size (RDIS):** After the elimination of the symmetry cases from domain pool, now we want to reduce the size of domain image to the 1/16th (64×64) of the original image size instead of the conventional domain size (1/4th (128×128) of the original size) by down sampling every 4×4 pixels in the original image (using the average method), as illustrated in Fig. 4, to one pixel in the reduced domain image. In this case, the number of the domain blocks will be reduced from 1024 to the 256 domain blocks. So, the computations needed in the encoding process will be reduced to be (4096×256) = 1,048,576 and this will decrease the encoding time significantly. In addition to the reduction that occurs in the number of bits required to encode the original image which is obtained from elimination of symmetry, a reduction in the number of bits needed to store each of *x* and *y* coordinates of the best matched domains will occur too because of reducing the domain size. In our example when the domain of size (64×64), the maximum value for each *x* and *y* coordinates will be (60) by dividing it on the jump step (60/4 =15) then the encoder will need 4 bits to store each of *x* and *y* coordinates. Accordingly, this will increase the compression ratio achieved by the reduction of domain size. Therefore, as a result, each block transform will be identified by only 17 bits; 4(bits for *x*) +4(bits for *y*) +3(bits for *s*) +6(bits for $\bar{r}$ ), so encoding the original image will require (4096×17 = 69,632 bits) = (8,704 bytes) = (8.5 KB) and this will increase the value of compression ratio.

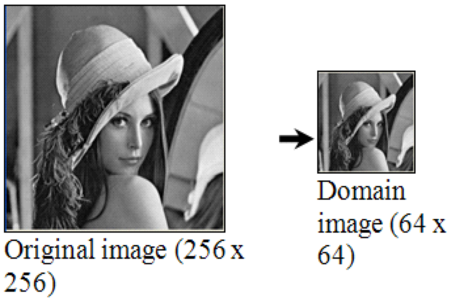

Fig. 4: Down sampling method using the average of (4×4) pixels

Table 1: The effects of different quantization levels of s and $\bar{r}$ parameters on the reconstructed Lenna and Parrots images

| Images | Totalno. bits | Scale bits | $\bar{r}$ bits | PSNR (dB) | C.R | E.T. (sec) |
|---|---|---|---|---|---|---|
| Lenna | 8 | 2 | 6 | 31.25 | 6.09 | 21 |
| | 8 | 3 | 5 | 30.80 | 6.09 | 21 |
| | 9 | 3 | 6 | 31.72 | 5.81 | 21 |
| | 10 | 4 | 6 | 32.11 | 5.56 | 21 |
| | 10 | 3 | 7 | 31.95 | 5.56 | 21 |
| | 11 | 5 | 6 | 32.27 | 5.33 | 21 |
| | 11 | 6 | 5 | 31.55 | 5.33 | 21 |
| | 12 | 5 | 7 | 32.49 | 5.12 | 21 |
| | 12 | 7 | 5 | 31.61 | 5.12 | 21 |

## RESULTS AND DISCUSSION

In this section, the discussion of the parameters that show significant effects on the results of the encoding process of the proposed ZMIL method will be done, and then it will show the effect of them on the encoding time, PSNR and compression ratio to evaluate these new results with the results of traditional FIC.

**Quantization levels of scale and range block mean:**We will show the effects of various quantization levels of the affine parameters *s* and $\bar{r}$ on the compression results using the uniform quantization. The results are tabulated in Table 1, which illustrate a limited number of total bits ranging from (8 to 12) has been adopted for *s* and $\bar{r}$ parameters. From the tests, we have found that if we used 8 bits as a total number of bits for *s* and $\bar{r}$ (i.e., *s* quantized by 2 bits and $\bar{r}$ is quantized by 6 bits) we will get more compression ratio and still Preserve High Quality (PSNR) for the reconstructed image. Therefore, ZMIL method reduces the number of bits which are required for representing the affine parameters over the traditional method, while it gets higher values of PSNR and

Table 2: Comparison between traditional and new FIC based on ZMIL method using different quantization levels of *s*, *o* and $\bar{r}$ .

| Totalno. bits | Scale bits | Off set bits | $\bar{r}$ bit | PSNR(dB) trad.FIC | PSNR(dB) new FIC |
|---|---|---|---|---|---|
| 8 | 2 | 6 | 6 | *28.42* | *31.25* |
| 9 | 3 | 6 | 6 | *29.44* | *31.72* |
| 10 | 4 | 6 | 6 | *29.90* | *32.11* |
| 11 | 5 | 6 | 6 | *30.22* | *32.27* |
| 12 | 5 | 7 | 7 | *30.30* | *32.49* |

Table 3: The effect of different block size on the reconstructed images

| Test image | Block size | No. of blocks | PSNR(dB) | C.R. | E.T.(sec) |
|---|---|---|---|---|---|
| Lenna | 4x4 | 4096 | 31.25 | 6.09 | 21 |
| | 8x8 | 1024 | 25.15 | 24.30 | 11 |
| | 16x16 | 256 | 21.10 | 96.2 | 27 |

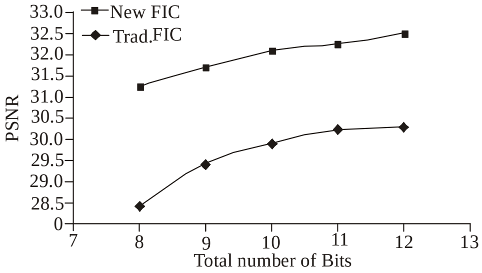

Fig.5: Total number of bits of traditional and new FIC parameters versus PSNR for Lenna image

compression ratio. Also, it is obvious from the Table 1, that the encoding time is reduced because the computations, involved with the mapping search operations of the ZMIL method, are simplified and reduced. ZMIL method shows a considerable increase in the PSNR values than that of the traditional one, as illustrated in Table 2 and shown in Fig. 5 and 6. So, we will use Scale Bits = 2 and $\bar{r}$ Bits = 6 for all the following tests.

**Block size effect:** Block size parameter affects directly in the encoding time, compression ratio and the quality of reconstructed image (PSNR). If the image is partitioned into small block size (4×4) we have large number of range blocks. These blocks are used in the searching stage, and have effects on the reconstructed values of PSNR and compression ratio. Also, these blocks need many computational processes in the encoding stage. While, if the image is partitioned into big block size (16x16), this means that it is allowed to increase the region with approximately self-similar, used in the searching stage, in this case the number of range blocks will be reduced and this will decrease the encoding time but with a low PSNR value. Table 3 shows these effects with various partitioning sizes of range blocks.

From traditional and new (ZMIL) FIC, as it is shown in Fig. 7, the big block size in the traditional method reconstructed poor image quality in the boundary regions like the eyes, mouth and hat of the Lenna image. While in the proposed method, obviously all the boundary details are reconstructed and in this case the PSNR will be increased.

**Decoding process of ZMIL FIC:** The results shown in Fig. 8 illustrate that from the first iteration; the decoded image will be the range-averaged image (DC) and then it will be used as an initial image for the next iterations. As the iterations proceeds, the AC components are added to the DC component (i.e., the reconstructed image equals the DC image (the range-averaged image) + *AC* images.), after 2 iterations the reconstructed image becomes stable. So, as it is shown in Fig. 8, the reconstructed image can reach its attractor (fixed point) at the 3rd iteration. In our ZMIL method, the distance between the original and reconstructed image is very contractive, this means that we can use a small value of contractivity factor (*MaxScl* = 0.5) and still get high PSNR value. But, in the traditional method, the contractivity factor (*MaxScl*=2) gets best PSNR value and if it was less than (*MaxScl* <2), the PSNR values are decreased. Table 4 shows this effect. ZMIL method is more powerful than the traditional method, because most researchers use high values of *MaxScl*, Fisher said that the best value lays in *MaxScl* >1.2., However in this research work it is noticed that this value may be reduced to be less than this value (i.e., until when *MaxScl* = 0.5) and the quality of the reconstructed image will still be good.

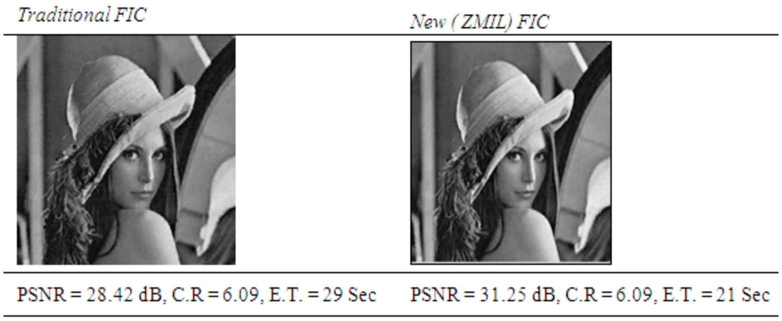
Traditional FIC
New (ZMIL) FIC
PSNR = 28.42 dB, C.R = 6.09, E.T. = 29 Sec
PSNR = 31.25 dB, C.R = 6.09, E.T. = 21 Sec

Fig. 6: Comparison between traditional and new FIC results for Lenna image when ScaleBits =2 and OffsetBits ( $\bar{r}$ Bits) = 6

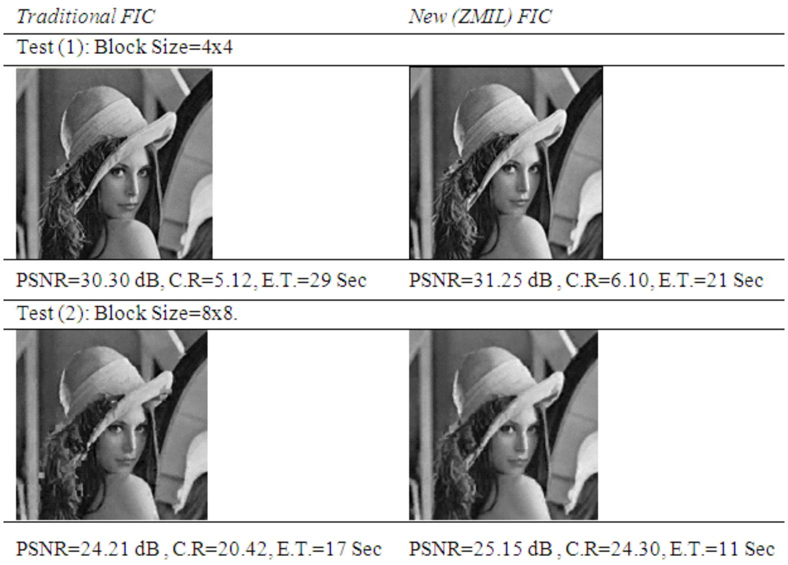
Traditional FIC
New (ZMIL) FIC
Test (1): Block Size=4x4
PSNR=30.30 dB, C.R=5.12, E.T.=29 Sec
PSNR=31.25 dB , C.R=6.10, E.T.=21 Sec
Test (2): Block Size=8x8.
PSNR=24.21 dB , C.R=20.42, E.T.=17 Sec
PSNR=25.15 dB , C.R=24.30, E.T.=11 Sec

Fig. 7: Comparison the results between traditional and new FIC based on ZMIL method when different bock size are used

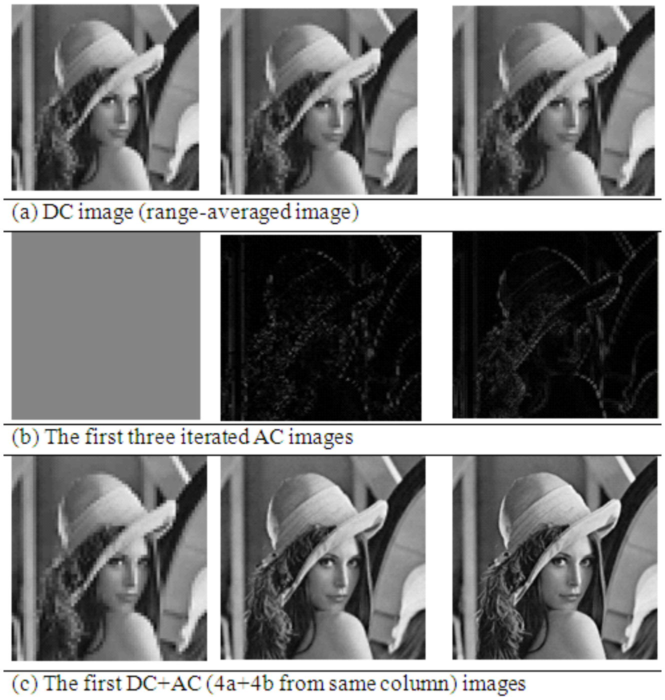
(a) DC image (range-averaged image)
(b) The first three iterated AC images
(c) The first DC+AC (4a+4b from same column) images

Fig. 8: DC+AC components for reconstructed image

**Results of speed up ZMIL method:** In this section, the effects of both the symmetry orientations and the reduction of the domain image size (RDIS) on the encoding results will be shown. It is obvious from Table 5, that the encoding time is reduced to about 33% with respect to the time required in the full cases when eliminate the last four symmetry orientations (reflection and its orientations). Also, it is discovered that excluding all symmetry cases (no symmetry); more reduction in encoding time is gained. From the results, the encoding time is reduced to about 80% as comparing with the time that is required in the full cases. It is noticed that the

Table 4: Show the effect of MaxScl on the PSNR of the reconstructed Lenna image in the traditional and new FIC

| | PSNR | |
|---|---|---|
| Max Scl | Traditional FIC | New FIC |
| 0.5 | 21.61 | 27.99 |
| 1.0 | 28.19 | 31.24 |
| 1.2 | 29.74 | 31.26 |
| 1.5 | 30.14 | 31.30 |
| 2.0 | 30.30 | 31.25 |
| 2.5 | 30.24 | 31.04 |

Table 5: The symmetry cases effects and RDIS on the encoding results

| Test image | Symmetry | PSNR(dB) | C.R | E.T.(ec) |
|---|---|---|---|---|
| Lenna | Full symmetry | 31.25 | 6.09 | 21 |
| | Half symmetry | 31.22 | 6.09 | 14 |
| | No symmetry | 30.44 | 6.73 | 4 |
| | No symmetry with RDIS | 30.18 | 7.52 | 1 |

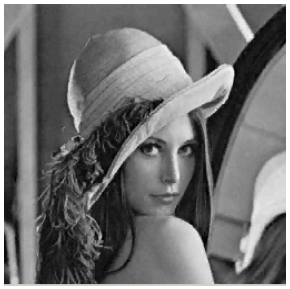

Fig. 9: Full symmetry and the reduction of domain image size

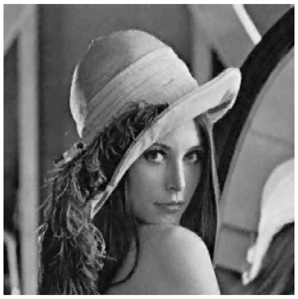

Fig. 10: Half symmetry and the reduction of domain image size

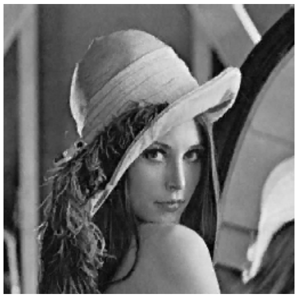

Fig. 11: Without symmetry and the reduction of domain image size

compression value is increased with good resolution of the reconstructed image. When the no symmetry is combined with RDIS, the encoding time will be reduced to only one second. Fig. 9, 10 and 11 show the effects of the symmetry cases.

## REFERENCES

Farhad, S., 2001. Fractal image compression. MAT 335, Tuesday, 15 May.

Fisher, Y., 1995. Fractal Image Compression: Theory and Application. John Wiley & Sons, New York, USA.

Hamzaoui R. and D. Saupe, 2000. Combining fractal image compression and vector quntization. IEEE T. Image Process., 9(2): 197-208.

Kadhim, G., 2001. Adaptive fractal image compression. M.Sc. Thesis, National Computer Center, Higher Education Institute of Computer and Informatics, Baghdad, Iraq.

Lee, S., 2000. Parallel processing architecture for fractal image compression Ph.D. Thesis, College of Engineering, Tohoku University, Japan.

Lisa, A.S., 2000. The Mathematical Foundation of Image Compression. A Paper Submitted in Partial Fulfillment of the Requirements of the Honors Program, Department of Mathematic and Statistics. The University of North Carolina at Wilmington, Wilmington, North Carolina. Retrieved from: http://people.uncw.edu/hermanr/signals/ImgComp_Soberano.PDF.

RADHA, H.R., 2004. Image Compression Using Fractal Algorithms. M. Sc. Thesis, Department of Computer Engineering, National Institute of Technology, Calicu, Kerala, Retrieved from: http://www.seminarprojects.com/attachment.php?aid=476.

Ruhl, M., H. Hartenstein and D. Suape, 1997. Adaptive Partitionings for Fractal Image Compression. Proceeding ICIP-97. IEEE International Conference an Image Processing, Santa, Barbara, California.

Tong, C.S. and M. Pi, 2001. Fast fractal image encoding based on adaptive search. IEEE Trans. Image Process., 10(9): 1269-1277.