Iteration-Free Fractal Image Coding Based on Efficient Domain Pool Design

Hsuan T. Chang[1] *and Chung J. Kuo*[2]

[1]Department of Information Management
Chao Yang University of Technology
Taichung, 413 TAIWAN

[2]Signal and Media (SAM) Laboratory
Department of Electrical Engineering
National Chung Cheng University
Chiayi, Taiwan 62107

ABSTRACT

The domain pool design is one of the dominant issues which affect the coding performance of fractal image compression. In this paper, we employ the LBG algorithm and propose a block-averaging method to design the efficient domain pools based on a proposed iteration-free fractal image codec. The redundancies between the generated domain blocks are reduced by the proposed methods. Therefore, we can obtain the domain pools that are more efficient than those in the conventional fractal coding schemes and thus the coding performance is improved. On the other hand, the iteration process in the conventional fractal coding scheme not only requires a large size of memory and a high computation complexity but also prolongs the decoding process. The proposed iteration-free fractal codec can overcome the problems above. In computer simulation, both the LBG-based and block-averaging methods for the domain pool design in the proposed iteration-free scheme achieve excellent performances. For example, based on the proposed block-averaging method, the decoded Lena image has at least a 0.5 dB higher PSNR (under the same bit rate) and an eight-time faster decoding speed than the conventional fractal coding schemes which require iterations.

Please send correspondence to Dr. Hsuan T. Chang
Address: Department of Information Management, Chao Yang University of Technology, Wufeng Taichung, 413 TAIWAN, R.O.C. Phone: 886-4-332-3000, ext 4232, Fax: 886-4-374-2337
E-mail: htchang@mail.cyut.edu.tw

1 Introduction

The fractal coding scheme is a new technique for image compression and has evolved greatly from its first version proposed by Jacquin [1], [2]. In conventional fractal coding schemes, an image is partitioned into non-overlapping range blocks. The larger domain blocks are selected from the same image and can overlap. A grayscale image is encoded by mapping the domain block D to the range block R with the contractive affine transformation [2]

$$\hat{R} = \iota\{\alpha \cdot (S \circ D) + \triangle g\}, \tag{1}$$

where $S\circ$ represents the contraction operation that maps the domain block to a range block. Then the parameters (called the *fractal code*) describing the contractive affine transformation that has the minimum matching error between the original range block R and the coded range block $\hat{R}$ are transmitted or stored. The fractal code consists of the contrast scaling α, luminance shift $\triangle g$ or the block mean (the average pixel value of the range block) μ_R [3], isometry ι, and the position P_D of the best-match domain block in the domain pool. In the decoding stage, an arbitrary image is given as the initial image and the decoded image is repeatedly reconstructed by applying the contractive affine transformation to the iterated image.

There are many modified versions proposed to improve the fractal coding techniques. Most of the studies focus on

i) the type of the image partition, *e.g.*, [26]$\sim$[30],

ii) refining the block transformation, *e.g.*, [4]$\sim$[10], [24],

iii) the reduction of the complexity of the encoding process, *e.g.*, [21], [31]$\sim$[33],

iv) speeding up the iterative decoding process [11]$\sim$[14], and

v) the combination of the conventional fractal coding scheme with traditional block-based image coding techniques, *e.g.*, [25], [34]$\sim$[37].

However, only few literatures [4], [23] make a study of designing an efficient domain pool in which the redundancies between the domain blocks can be reduced. Therefore, we aim to design an efficient domain pool for the fractal coding schemes in this paper. In Ref. [4], the LBG procedure is designed for codebook (domain pool) training in conventional fractal coding schemes. It considers the encoding procedure only and it is not easy to obtain an identical codebook in the decoder unless an off-line transmission is used. In our design, the domain pool is on-line transmitted and hence we obtain an identical codebook in the decoder. On the other hand, a still image coding based on vector quantization (VQ) and fractal approximation utilized the LBG algorithm to design the domain pool [23]. The domain pool generated in [23] is based on the image approximated by transform VQ and then decimated by a factor of four. And the static codebook is previously constructed with several images of different types. As we will describe in Section 4, it has a great difference with our domain pool design since we will construct the domain pool (*e.g.*, codebook) only with the mean image and not with several images.

In general, the domain pool in conventional schemes consists of the domain blocks obtained by subsampling the original image [1], [17], [26], or choosing some neighboring blocks of the range block [5], [21], [37]. A better coding performance can be achieved if we use a larger domain pool in the encoding stage. However, there exists some redundancies between the domain blocks, especially for a large domain pool or the domain blocks chosen from the neighboring blocks of the range block. If we can reduce the redundancies between the domain blocks, then the constructed domain pool becomes more efficient. Therefore, a better performance can be expected because the domain blocks in our domain pool contain more information than those in the domain pools of conventional fractal coding schemes.

The LBG algorithm [18] used to generate the codebook in the VQ techniques [16] has shown its ability to reduce the redundancies between the training vectors. Based on the same codebook in both the encoder and decoder[1], we can encode/decode an image. Since there is no transmission of domain blocks in conventional fractal coding schemes, the LBG algorithm cannot be directly applied to generate the domain blocks. In order to obtain the same domain blocks in both the encoder and decoder in the fractal coding scheme, here we propose an iteration-free fractal coding scheme that can satisfy this requirement.

The block mean can be found in the fractal code in a modified contractive affine transformation [3]. We can generate the same mean image whose pixel values are the block means of all the range blocks in both the encoder and the decoder. Therefore, the LBG-based method can be used to design the domain pool based on the mean image. Here two novel methods for designing the domain pool are employed. First, the domain pool is consisted of the domain blocks generated by the LBG-based

[1]The VQ techniques use an off-line transmission of codebook.

method. Next, the block-averaging method is proposed to avoid the training process in the LBG-based method. The LBG-based method reduces the redundancies between the generated domain blocks and thus the constructed domain pool is more efficient. Compared with the conventional fractal schemes those require iterations, the coding performance is improved based on the LBG-based and the block-averaging methods for the domain pool design in the proposed iteration-free fractal coding scheme. The computer simulation shows that the decoding time is greatly reduced and the decoded image quality is also improved.

The organization of this paper is as follows: We introduce the conventional fractal coding scheme that requires iterations in the decoding stage in Section 2. Section 3 describes the proposed iteration-free codec for fractal image compression. Two design methods of the domain pool are employed to improve the performance of the iteration-free coding scheme in Section 4. We perform the computer simulation in Section 5 to verify the improvement of the proposed domain pool design for the iteration-free scheme. Finally, a conclusion is given in Section 6.

2 Conventional Fractal Coding Scheme

The employed domain pool designs based on the LBG-based and the block-averaging methods are compared with those in the conventional fractal coding schemes. In conventional fractal coding schemes, the contrast scaling is no more than one to avoid the possible divergence in the iterative decoding process. On the other hand, we replace luminance shift in the fractal code by the block mean [3] to obtain a good initial image in the decoding stage.

The domain pool designs in the conventional fractal coding schemes are described as follows: Basically, the domain blocks are selected from the original image and the block size is four times as large as the range block (*i.e.,* $D=4R$). We investigate the coding performances of the conventional fractal coding scheme that uses two methods to design the domain pool. First of all, the domain pool consists of the domain blocks subsampled from the original image and it is denoted as the 'subsampling' method. For an image of size $M \times M$, the sampling period (T) in both the horizontal and vertical directions is determined by

$$T = \lfloor \frac{M - B}{\sqrt{N_D} - 1} \rfloor, \quad T \geq 1, \tag{2}$$

where $B \times B$ is the domain block size and N_D is the number of the domain blocks in the domain pool, and $\lfloor \cdot \rfloor$ denotes choosing a smaller and the closest integer of the real number in the bracket. Next, we choose the N_D domain blocks that are neighboring to the range block and it is denoted as the 'neighboring' method. Then the contractive affine transformation is used to find the fractal code for each range block.

In the decoding stage, the initial image consists of the range blocks whose pixel values are equal to each block mean. The decoded image is iteratively reconstructed with the same contractive affine transformation that was denoted in the fractal code. Since the initial image in the decoder is different from the original image in the encoder, the domain blocks in the encoder are different from that found in the decoder. There exists a distortion between the coded image in the encoder and the decoded image in the decoder. To reduce this distortion, it is desirable to generate the same domain pool in both the encoder and decoder.

The criterion for the decoded image to achieve a convergence is determined as follows: Let the nth iterated image be denoted as $f^{(n)}$. The average error $e(n)$ between the nth and $(n-1)$th decoded images is calculated by

$$e(n) = \frac{1}{512^2} \sum_{i=1}^{512} \sum_{j=1}^{512} (f_{i,j}^{(n)} - f_{i,j}^{(n-1)})^2, \tag{3}$$

where $f_{i,j}^{(n)}$ denotes the (i,j)th pixel in nth decoded image. If the ratio

$$\gamma = \frac{|e(n) - e(n-1)|}{e(n-1)} \tag{4}$$

is smaller than a threshold value γ_{th}, the decoded image converges and the iteration process terminates. Otherwise, the iteration process will not stop until the criterion $\gamma \leq \gamma_{th}$ is satisfied.

3 Iteration-Free Codec Design

In order to obtain the same domain blocks in both the encoder and decoder without using an off-line transmission, here we propose an iteration-free fractal image codec that the information of the domain blocks are hidden in the fractal codes. Therefore, the LBG-based and the proposed block-averaging methods can be applied to reduce the redundancies between the generated domain blocks. The proposed encoder and decoder are described in the following subsections.

3.1 Encoder

The basic flow chart of the encoder in the proposed iteration-free scheme is shown in Fig. 1. The input $M\times M$ image is partitioned into the non-overlapping range blocks of size $B\times B$. First of all, we sequentially measure the mean and variance of each range block. After all the means of the range blocks are obtained, we can generate a mean image of size $M/B\times M/B$ with each pixel corresponding to the block mean. If the variance of the range block

$$\text{Var}\{R\} = \frac{1}{B^2} \sum_{0\leq i,j<B} (r_{i,j} - \mu_R)^2 \tag{5}$$

(where $r_{i,j}$ denotes the (i,j)th pixel in the range block) is smaller than the threshold value E_{th}, then the range block is coded by the mean. Otherwise, the range block will be coded by the contractive affine transformation. Note that in this case, the size of the mean image should be much larger than that of the domain block, *i.e.,* $M/B\times M/B \gg B\times B$. Otherwise, it will not be easy to find a good mapping between the domain and range blocks because only a few domain blocks can be taken from the mean image. The size of the domain block is the same as that of the range block and thus the contraction procedure in conventional fractal coding schemes is eliminated. We therefore proceed with a new contractive affine transformation between the range block and the domain block generated from the mean image. The generation of the domain block will be discussed in Section 4.

The parameters used in the new contractive affine transformation are specified as follows: The luminance shift is replaced by the mean [3] which is coded by six bits. The contrast scaling is usually smaller than 1.0 to avoid the divergence caused by the iterations in conventional fractal coding schemes. However, we can make the contrast scaling be greater than 1.0 because our scheme is iteration-free. As shown in [17], the contrast scaling can be greater than one to achieve the minimum distortion between the range block and the transformed domain block. Therefore, we use an extended range for the contrast scaling. In our design, the contrast scaling is determined by testing all the values in the following set $\{n/4,\ n=1,\ 2,\ 3,\ \cdots,\ 8\}$ to find the best one that minimizes the distortion. We thus need three bits to denote the contrast scaling. On the other hand, the eight isometries for shuffling the pixels in the block are the same as those in [2] and are coded by three bits.

The new contractive affine transformation can be expressed by

$$\hat{R} = \iota\{\alpha \cdot D + \mu_R - \alpha \cdot \mu_D\} = \iota\{\alpha \cdot (D - \mu_D) + \mu_R\}, \tag{6}$$

where $\hat{R}$ is the coded range block and μ_D is the mean of domain block. Note that the contraction procedure is eliminated and the term $\mu_R - \alpha \cdot \mu_D$ is equal to the luminance shift in [2]. After testing all the combinations of the parameters in Eqn. (6), the fractal code is determined while the coded block $\hat{R}$ has the minimum distortion from the original range block R. The distortion between the original and coded range blocks is represented by the mean-squared-error (MSE) measurement defined as

$$\text{MSE}(R, \hat{R}) = \frac{1}{B^2} \sum_{0<i,j\leq B} (r_{i,j} - \hat{r}_{i,j})^2, \tag{7}$$

where $\hat{r}_{i,j}$ denotes the (i, j)th pixel in the coded range block. We finally attach a header for each range block to denote its coding status (either coded by the mean or affine transformation). Therefore, the decoder can correctly reconstruct each coded range block according to the header.

3.2 Decoder

Fig. 2 shows the flow chart of the decoder in the proposed iteration-free scheme. We firstly receive the entire fractal code and determine whether or not the range block is coded by the mean from its header. The mean image is reconstructed with the mean information in the fractal codes. Note that this mean image is identical to the mean image used in the encoder since both are constructed by the same block means. Therefore, the domain blocks generated from both mean images are also the same. If the block is coded by the mean, the value of each pixel in the decoded block is equal to the mean value. Otherwise, we perform the contractive affine transformation to reconstruct the coded range block. The decoding process ends when the last range block is reconstructed.

At this point, no iterations are required and thus no convergence criterion and divergence problem for the decoded image to be concerned with. The decoding of the conventional fractal coding scheme may require two iteratively refreshed images or one image memory [12] in the iteration process. However, only the fixed mean image that can be reconstructed from the received fractal codes is required in our iteration-free scheme. Hence the required memory size in the proposed iteration-free decoder is much smaller than that in the conventional fractal image decoder. On the other hand, having no iterations means that the range blocks can be decoded in parallel. The architectural complexity of the proposed decoder is obviously lower than that of the conventional decoder that requires iterations. Therefore, the proposed decoder is very suitable for the hardware implementation and high speed applications.

4 Efficient Domain Pool Design

In order to obtain an efficient domain pool in which the redundancies between the domain blocks are reduced, here we firstly employ the LBG algorithm and secondly propose a novel black-averaging method to generate the domain blocks. Therefore, we expect that the coding performance will be improved compared with the conventional fractal schemes.

4.1 LBG-Based Design

Fractal coding techniques have shown a similarity to VQ techniques and can be considered as the self-VQ for images [1,19]. Therefore, the codebook and codevector used in the VQ technique are similar to the domain pool and domain block used in fractal coding schemes, respectively. In VQ, the encoder and decoder use the same codebook and the coded and decoded images in both the encoder and decoder are also the same. As shown in Section 3, we can obtain the same mean image in both the encoder and the decoder without using an off-line transmission in the proposed iteration-free codec. We thus design an efficient domain pool based on the mean image.

The LBG algorithm [18] is usually used to design an efficient codebook in the VQ techniques. We apply the LBG algorithm to design the domain pool in the proposed iteration-free scheme. Here we use the mean image as the training image and all the possible image blocks (with the same size as the range block) in the mean image as the training vectors. Suppose that there are K training vectors denoted by $\mathbf{v}_i$ for $1\leq i \leq K$ in the training image, we estimate L reconstruction vectors (*i.e.,* domain blocks) from K training vectors. The reconstruction vectors are determined by minimizing the average distortion defined by

$$E_{av} = \frac{1}{K}\sum_{i=1}^{K}\mathrm{MSE}(\mathbf{v}_i, \hat{\mathbf{v}}_i), \tag{8}$$

where $\hat{\mathbf{v}}_i$ denotes $\mathbf{v}_i$ that has been quantized into one of the reconstruction vectors. In the LBG algorithm, we begin with an initial estimate of the reconstruction vectors $\mathbf{s}_i$ for $1\leq i \leq L$. We then classify the K training vectors into L different clusters corresponding to each reconstruction vector. This can be done by comparing a training vector with each of the reconstruction vectors and choosing the vector that results in the smallest distortion. A new reconstruction vector is determined from the vectors in each cluster. This completes one iteration of the procedure, which can be stopped when the average distortion E_{av} does not change significantly between two consecutive iterations.

By applying the LBG algorithm to all the possible image blocks in the mean image, each generated domain block has the smallest average distortion with those image blocks in the same cluster. Therefore, we construct the domain pool by specifying an L value to obtain a desired number of domain blocks. The LBG algorithm reduces the redundancies of similar image blocks in the mean image. Hence the generated domain blocks have fewer redundancies than the domain blocks directly obtained from the mean image. Apparently, the constructed domain pool is more efficient.

However, the training process in the LBG algorithm requires iterations to obtain the minimum quantization error. The larger the codebook size, the bigger the iteration number. The LBG algorithm is employed in both the encoder and the decoder to generate the domain blocks. Therefore, it also requires the iteration process to generate the domain blocks in the decoder. This is the limitation in applying this method to the domain pool design for the proposed iteration-free scheme. To solve this problem, we next propose the block-averaging method by which the domain block is directly generated without iterations.

4.2 Block-Averaging Method

In the mean image, the training vectors chosen from the neighboring blocks have a high similarity because most parts of the blocks are overlapped. They can be considered as the vectors of the same cluster in the LBG algorithm. Here the block-averaging method is proposed based on the similar concept of the LBG algorithm. We compute the centroid of four blocks which are adjacent and partly overlapped in the mean image to generate a domain block. More image blocks can be averaged to reduce more redundancies among them. However, it is expected that the correlation between the four neighboring blocks will be higher than that between more neighboring blocks. Therefore, we use only four neighboring blocks in the block-averaging method.

Fig. 3(a) shows some sets of four neighboring blocks with a four-pixel sampling period in the mean image. In this figure, each black point denotes the top-left corner of an image block and we use it to represent a $B \times B$ image block. Their relative positions are shown in Fig. 3(b). The pixel $\bar{d}_{i,j}$ in the averaged block $\bar{D}$ can be calculated by

$$\bar{d}_{i,j} = \frac{1}{4}(d1_{i,j} + d2_{i,j} + d3_{i,j} + d4_{i,j}), \quad 0 \le i, j < B, \tag{9}$$

where $d1_{i,j} \sim d4_{i,j}$ represents the (i,j)th pixel in the image bocks $D1 \sim D4$. Therefore, the calculated pixel $\bar{d}_{i,j}$ relates to the information of four adjacent pixels in the original image blocks. The averaged block replaces the original four adjacent image blocks and the redundancies in the four adjacent image blocks are thus reduced. With these averaged blocks, the constructed domain pool is more efficient than that consists of the domain blocks directly selected from the mean image.

The domain blocks are uniformly selected from the averaged blocks with a sampling period (T') in the mean image. Let the number of domain blocks in the domain pool be N_D, the sampling period in both the horizontal and vertical directions can be calculated by

$$T' = \lfloor \frac{M/B - B}{\sqrt{N_D} - 1} \rfloor, \quad T' \ge 1. \tag{10}$$

Instead of using the training process in the LBG algorithm, here we only use the four-to-one averaging operation to generate the domain blocks. Thus the required computation complexity is much less than that in the LBG-based method and the high decoding speed property in our iteration-free scheme is preserved.

5 Computer Simulation

In computer simulation, four 512×512 images (shown in Fig. 4(a)∼(d)) with eight-bit grayscale resolution are used to test the proposed iteration-free fractal coding scheme. The performance of the decoded image quality is evaluated by the peak signal-to-noise-ratio (PSNR) and the bit rate (the required bits per pixel). In our simulation, an image is partitioned into range blocks with the single size, either 8×8 or 4×4, or with two-level sizes (both 8×8 and 4×4). Therefore, a general form for the PSNR of the decoded image is defined as

$$\text{PSNR} = 10 \log_{10} \frac{255^2}{\sum_{i=1}^{N_8} \text{MSE}(R_{8_i}, \hat{R}_{8_i}) + \sum_{i=1}^{N_4} \text{MSE}(R_{4_i}, \hat{R}_{4_i})} \quad \text{dB}, \tag{11}$$

where N_8 and N_4 are the total numbers of the 8×8 range block R_8 and the 4×4 range block R_4, respectively. As for the bit rate calculation, it will be given in the following subsections.

For all the schemes used in our simulation, we set the threshold values E_{th} for the variance of 8×8 and 4×4 range blocks to be 25. The size of the domain pool is represented by the number of domain blocks in it. There are four sizes used in our domain pool design: N_D=16, 64, 256, and 1024. On the other hand, the sampling periods T' and T, for the block-averaging method and the conventional domain pool design, are determined according to Eqns. (2) and (10).

5.1 Single Block Size

First of all, the range block with a single size (8×8 or 4×4) is considered. The length of the attached header I_h to the fractal code for each range block is only one bit (*i.e.*, I_h=1) because it only denotes whether or not the range block is coded by the mean. Therefore, the bit rate can be calculated by

$$\mathcal{B}_1 = \frac{(N_\mu + N_{af})(I_h + I_\mu) + N_{af}(I_\alpha + I_\iota + I_{P_D})}{512^2} \quad \text{bit/pixel,} \tag{12}$$

for a single block size, where I_μ, I_α, I_ι, and I_{P_D} denote the required bits for the block mean, contrast scaling, isometry, and the position of the domain pool, respectively. In addition, N_μ and N_{af} denote the numbers of the blocks coded by the mean and affine transform, respectively.

For an image partitioned by 8×8 range blocks, we measure every block mean and obtain a 64×64 mean image. Fig. 5(a) shows that the mean image is very similar to its original Lena image except its size. We therefore construct the domain pools of different sizes using the LBG-based and block-averaging methods. There are 57×57 possible image blocks used as the training vectors in the the LBG-based method. We demonstrate an example of the constructed domain pools that consist of N_D=256 8×8 domain blocks in Fig. 6(a) and (b) for the LBG-based and block-averaging methods, respectively. As shown in Fig. 6(a), all the trained domain blocks are different with each other and hence the redundancies between them are reduced. On the other hand, the generated domain blocks in Fig. 6(b) show a high correlation with the original Lena image.

We determine the coding performance with the contractive affine transformation under the different sizes for the domain pool. Fig. 7(a) shows the simulation results for the Lena image. The numbers shown in the figure represent the different sizes of the domain pool. The bit rates of all the schemes while using the same size of the domain pool are the same. A smaller size for the domain pool leads to a lower bit rate and vice versa. The LBG-based method has an excellent performance by using smaller domain pools, while the block-averaging method has the best performance when the size of domain pool is 1024.

For the image partitioned by 4×4 range blocks, the 128×128 mean image for Lena is obtained and shown in Fig. 5(b). We also construct the domain pools of different sizes using the LBG-based and block-averaging methods. The simulation results based on the same sizes for the domain pool are shown in Fig. 7(b). Two proposed design methods provide better performances than the conventional fractal coding scheme when the size of the domain pool is above 64. As the size of the domain pool increases, the improvement of the performance becomes more obvious. The PSNR of the decoded image partitioned by the 4×4 block size is much higher than that partitioned by the 8×8 block size since a smaller block size leads to a smaller matching error for the affine transformation. However, the bit rate increases significantly because the number of the 4×4 range blocks is four times the number of the 8×8 range blocks.

5.2 Two-Level Block Sizes

From the results shown in Fig. 7(a) and (b), the chosen block size greatly affects the bit rate and the PSNR of the coded image. In order to compromise the bit rate and PSNR for the coded image, it is desired to partition an image into the range blocks with two-level (parent 8×8 and child 4×4) sizes. An image is first partitioned into parent range blocks and the coding procedures are the same as that in Subsection 5.1. If the parent range block is coded by the contractive affine transformation and the distortion between the original and coded range blocks, MSE(R_8,$\hat{R}_8$), is greater than the threshold value E_{th}=25, the parent range block is split into four child range blocks. The coding procedures for the child range block are the same as that described in Subsection 5.1.

Now, the bit rate is affected by the number of the partitioned parent and child range blocks. The more the parent range blocks in the coded image, the lower the final bit rate. If we choose a larger size for the domain pool in the parent level, more parent range blocks can satisfy the criterion MSE(R_8,$\hat{R}_8$)$\leq$25. We thus choose a large domain pool size (N_D=1024) for the parent range block such that the number of the coded parent range block can be increased. At the same time, the number of child range blocks is decreased to obtain a lower bit rate. Finally, the size of the child domain pool is varied to examine the PSNR performance of the proposed iteration-free scheme under different bit rates.

To identify the different partitions for the parent range block, we attach a variable-length header to the fractal code. Table 1 shows the header and the bit allocation for the parent range block R_8. We assign '0' as the header of the mean-coded parent range block. For the parent range block coded by the affine transformation, '10' is the header. The header '11' represents that a parent range block is split into four child blocks. Then, the subheaders '0' and '1' represent the child range block coded by the mean and the affine transformation, respectively. Therefore, the header has various lengths (one, two, and six bits) for different parent range blocks. The bit rate can be calculated by

$$\mathcal{B}_2 = \frac{N_{8_\mu} + 2N_{8_{at}} + 6N_{84} + (N_8 + N_4)I_\mu + (N_{8_{at}} + N_{4_{at}})(I_\alpha + I_\iota + I_{P_D})}{512^2} \quad \text{bit/pixel}, \tag{13}$$

where $N_{8_\mu}, N_{8_{at}}, N_{84}$, and $N_{4_{at}}$ denote the number of the parent range blocks coded by the mean, coded by affine transformation, partitioned into four child range blocks, and the child range blocks coded by the affine transformation, respectively.

Fig. 8(a) shows the simulation results of the Lena image based on the proposed iteration-free and conventional fractal coding schemes. Using two-level block sizes, the resultant bit rate and PSNR performance of the proposed methods are within a moderate range. The LBG-based and the proposed block-averaging methods significantly improve the PSNR under the same bit rate (except for the case that the domain pool size N_D=16 in the LBG-based method). In order to verify that the proposed methods also perform well for other images, the simulation results for three other images: Jetplane, Building, and Harbour (shown in Fig. 4(b)$\sim$(d)) are also given in Fig. 8(b)$\sim$(d). Apparently, the performance of these images are greatly improved by using the domain pools that are designed based on the LBG-based and the proposed block-averaging methods. For example, the PSNR improvement of using the block-averaging method to design the domain pool for the Jetplane image is more than 1 dB (in average) compared with both the conventional subsampling and neighboring methods at the same bit rate. Based on these simulation results, we verify that the proposed methods design efficient domain pools and thus achieve a good coding performance.

Here we also list the computation time on a SUN Ultra–1 workstation for the proposed iteration-free scheme, whose domain pool is design based on the block-averaging method, and two conventional fractal coding schemes. The threshold value γ_{th} for the convergence criterion in the conventional fractal coding scheme is set by 0.005. Table 2 shows their CPU time (in seconds, the decoding program is not optimized) for decoding the Lena image. The proposed iteration-free scheme saves about 87% the decoding time required in the conventional fractal coding scheme. With the proposed domain pool design for the iteration-free fractal coding scheme, we not only greatly speed up the decoding procedure but also improve the decoded image quality.

5.3 Comparison

As shown in Figs. 7 and 8, the performances of the proposed iteration-free fractal coding scheme whose domain pool design is based on the LBG-based and block-averaging methods are better than conventional fractal coding schemes that require iterations. The proposed iteration-free scheme performs coding only in the spatial domain, *i.e.,* it does not combine other coding techniques such as the transform coding and the subband coding. Therefore, here we compare the bit rate and PSNR of the decoded Lena image between the proposed iteration-free scheme and the existing fractal coding schemes that also perform coding in the spatial domain only.

Fig. 9 shows the comparison between the proposed iteration-free scheme and other competitive fractal coding schemes. Obviously, not only our iteration-free scheme speeds up the decoding process but also the proposed domain pool design based on the LBG-based and block-averaging methods achieves superior performances on the bit rate and PSNR for the decoded image. We also make a comparison with the JPEG standard[2] in Figure 10. The simulation results are obtained by varying the threshold value E_{th} and the Q-factor in the proposed and JPEG schemes, respectively. For the bit rate higher than 0.33 bit/pixel, the performance of the proposed method is close to that in JPEG standard. However, when the bit rate is smaller than 0.33 bit/pixel, the proposed method shows a significant improvement compared with the JPEG standard. Obviously, the proposed method is more suitable than JPEG standard in the applications of the very low bit rate coding.

6 Conclusion

In this paper, we employ the LBG-based and propose the block-averaging methods to design efficient domain pools for the iteration-free fractal image codec. The redundancies between the generated domain blocks are reduced and thus the constructed domain pool is more efficient than those in conventional

[2]We use the "cjpeg" and "djepg" files in the software package HIPS to execute the compression and decompression of the test image.

fractal schemes. Simulation results show that we make a significant improvement on both the decoding speed and the coding performance. The main drawback of the LBG-based method is that it also needs iterations in the training process. However, this limitation dose not appear in the block-averaging method. Compared with the existing fractal coding schemes, the proposed iteration-free scheme, utilizing the LBG-based or block-averaging methods for the domain pool design, achieves a superior performance. Therefore, based on the proposed domain pool design, the iteration-free fractal scheme shows its characteristics of high decoding speed and excellent image quality for fractal image compression. For the cases of very low bit rate coding, the performance for the proposed scheme is also better than that for JPEG standard.

Acknowledgment

This research was partially supported by National Science Council, Taiwan, under contract NSC 88-2612-E-324-001.

References

[1] A.E. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations," *IEEE Trans. on Image Processing*, vol. 1, no. 1, pp. 18–30, January 1992.

[2] A.E. Jacquin, "Fractal image coding: a review," *Proceedings of the IEEE*, vol. 81, pp. 1451–1465, October 1993.

[3] J.M. Beaumont, "Advances in block based fractal coding of still images," *IEE Colloquium on 'Application of Fractal Techniques in Image Processing'*, pp. 3/1–3/5, London, UK 1990

[4] Y. Fisher, *Fractal image compression: Theory and applications*, Springer-Verlag, New York, 1995

[5] H.T. Chang and C.J. Kuo, "An improved scheme for fractal image coding," *IEEE 1995 International Symposium on Circuits and Systems*, vol. 3, pp. 1624–1627. Seattle, May 1995

[6] H.T. Chang and C.J. Kuo, "Finite-state fractal block coding of images," *IEEE 1996 International Conference on Image Processing*, vol. 1, pp. 133–136, Switzerland, September 1996

[7] G. Oien, A. Sollid and T. Ramstad, "An inner product space approach to image coding by contractive transformations," *IEEE 1991 International Conference on Acoustics, Speech, and Signal Processings*, vol. 4 pp. 2773–2776, Toronto, Apr. 1991

[8] T. Bedford, F. Dekking, M. Breeuwer, M. Kenae, and D. Schooneveld, "Fractal coding of monochrome images," *Signal Processing: Image Communication*, vol. 6, pp. 405–419, 1994

[9] H. Zhang, X. Gao, and Z. He, "A modified fractal transform," *IEEE 1995 International Conference on Acoustics, Speech, and Signal Processings*, vol. 4, pp. 2567–2570, Detroit, May 1995

[10] D. Saupe, "A new view of fractal image compression as convolutional transform coding," *IEEE Signal Processing Letters*, vol. 3, no. 7, pp. 193–195, July 1996

[11] S. Lepsoy, G. Oien and T. Ramstad, "Attractor image compression with a fast non-iterative decoding algorithm", *IEEE 1993 International Conference on Acoustics, Speech, and Signal Processings*, Minneapolis, vol. 5, pp. 337–340, Apr. 1993

[12] H. Kang and S. Kim, "Fractal decoding algorithm for fast convergence," *Optical Engineering*, vol. 35, no. 11, pp. 3191–3198, Nov. 1996

[13] R. Hamzaoui, "Decoding algorithm for fractal image compression," *Electronics Letters*, vol. 32, no. 14, pp. 1273–1274, July 1996

[14] C. Kim, R. Kim and S. Lee, "Novel fractal image compression method with non-iterative decoder," *IEEE 1995 International Conference on Image Processing*, vol. 3, pp. 268–271, Washington, D.C., Oct. 1995

[15] H.T. Chang and C.J. Kuo, "A universal iteration-free fractal image codec," *1997 International Conference on Consumer Electronics*, Chicago, June 1997

[16] A. Gersho and R.M. Gray, *Vector quantization and signal compression*, Kluwer Academic Publishers, Taiwan, 1992.

[17] F. Jocob, Y. Fisher, and R. Boss, "Image compression: a study of the iterated transform method," *Signal Processing*, vol. 29, no. 3, pp. 251–263, Dec. 1992

[18] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantization design," *IEEE Trans. on Communications*, vol. 28, no. 1, pp. 84–95, Jan. 1980

[19] Y. Fisher, D. Rogovin and T. Shen, "Fractal (self-VQ) encoding of video sequences," *Proceedings of SPIE*, vol. 2308, pp. 1359–1370, 1994 Chicago, Sep. 1994

[20] G. Lu and T. Yew, "Image compression using partitioned iterated function system," *Proceedings of SPIE*, vol. 2186, pp. 122–132, 1994

[21] H. Kuroda, D. Popescu, and H. Yan, "Fast block matching method for image data compression based on fractal methods," *Proceedings of SPIE*, vol. 2501, pp. 1257–1266, 1995

[22] D. Saupe, "The futility of square isometries in fractal image compression," *IEEE International Conference on Image Processing*, vol. 1, pp. 161–164, Switzerland, Sep. 1996

[23] I. Kim and R. Park, "Still image coding based on vector quantization and fractal approximation," *IEEE Trans. on Image Processing,* vol. 5, no. 4, pp. 587–597, Apr. 1996

[24] D. Popescu, A. Dimca, and H. Yan, "A nonlinear model for fractal image coding," *IEEE Trans. on Image Processing,* vol. 6, no. 3, pp. 373–382, Mar. 1997

[25] L. Thomas and F. Deravi, "Pruning of the transform space in block-based fractal image compression," *IEEE 1993 International Conference on Acoustics, Speech, and Signal Processings*, vol. 5, pp. 341–344, Minneapolis, Apr. 1993

[26] G. Lu and T.L. Yew, "Image compression using quadtree partitioned iterated function systems," *Electronic Letters*, vol. 30, no. 1, pp. 23–24, Jan. 1994

[27] E. Reusens, "Overlapped adaptive partitioning for image coding based on the theory of iterated functions systems," *IEEE 1994 International Conference on Acoustics, Speech, and Signal Processings*, vol. 5, pp. 569–572, Adelaide, Australia, 1994

[28] L. Thomas and F. Deravi, "Region-based fractal image compression using heuristic search," *IEEE Trans. on Image Processing,* vol. 4, no. 6, pp. 832–838, June 1995

[29] F. Davoine, M Antonini, J. Chassery, and M. Barlaud, "Fractal image compression based on Delaunay triangulation and vector quantization," *IEEE Trans. on Image Processing,* vol. 5, no. 2, pp. 338–346, Feb. 1996

[30] M. Tanimoto, H. Ohyama, T. Kimoto, S. Katsuyama and T. Fujii, "A new fractal image coding scheme employing blocks of variable shapes," *IEEE 1996 International Conference on Image Processing*, vol. 1, pp. 137–140, Switzerland, Sep. 1996

[31] C. Wein and I. Blake, "On the performance of fractal compression with clustering," *IEEE Trans. on Image Processing,* vol. 5, no. 3, pp. 522–526, Mar. 1996

[32] H. Lin and A. Venetsanopoulos, "Fast pyramid search for perceptually based fractal image compression," *IEEE 1996 International Conference on Image Processing*, vol. 1, pp. 173–176, Switzerland, Sep. 1996

[33] B. Bani-Eqbal, "Enhancing the speed of fractal image compression," *Optical Engineering,* vol. 34, no. 6, pp. 1705–1710, June 1996

[34] R. Rinaldo and G. Calvagno, "Image coding by block prediction of multiresolution subimages," *IEEE Trans. on Image Processing,* vol. 4, no. 7, pp. 909–920, July 1996

[35] B. Simon, "Image coding using overlapping fractal transform in the wavelet domain," *IEEE 1996 International Conference on Image Processing*, vol. 1, pp. 177–180, Switzerland, Sep. 1996

[36] Y. Tang and W. Wee, "Fractal-based image compression - a fast algorithm using wavelet transform," *Proceedings of SPIE*, vol. 2308, pp. 1674–1682, 1994

[37] K. Barthel, J. Schuttemeyer, T. Voye and P. Noll, "A new image coding technique unifying fractal and transform coding," *IEEE 1994 International Conference on Image Processing*, vol. 3, pp. 112–116, Austin, Nov. 1994

Table and Figure Captions:

Table 1 Header and bit allocation for both 8×8 and 4×4 range blocks.

Table 2 Decoding time (in seconds) for decoding the Lena image.

Figure 1 The flow chart of the encoder for the proposed iteration-free scheme.

Figure 2 The flow chart of the decoder for the proposed iteration-free scheme.

Figure 3 (a) Some sets of the image blocks used to generate the domain blocks in the block-averaging method, (b) The relative position for four neighboring and partly overlapped image blocks $D1 \sim D4$.

Figure 4 The original images (512×512, 8 bit/pixel) used in the proposed iteration-free fractal coding scheme: (a) Lena, (b) Jetplane, (c) Building, and (d) Harbour.

Figure 5 Two mean images of size (a) 64×64 and (b) 128×128 for the Lena image.

Figure 6 The constructed domain pools (consist of 256 domain blocks) from the mean image shown in Fig. 5(a) by using (a) the LBG-based method (b) the block-averaging method.

Figure 7 Coding results of the proposed iteration-free scheme using the single-size design for the range block: (a) 8×8, (b) 4×4.

Figure 8 Coding results of the proposed iteration-free scheme using two-level sizes for the range block. (a) Lena, (b) Jetplane, (c) Building, (d) Harbour.

Figure 9 Performance comparison of the Lena image for the proposed iteration-free scheme and other fractal coding schemes.

Figure 10 Performance comparison of Lena image between the proposed iteration-free scheme and JPEG image standard.

BLOCK TYPE	HEADER	BIT ALLOCATION			
		I_ι	I_{μ_R}	I_α	I_{P_D}
R_8 CODED BY MEAN μ_R	'0'		6		
R_8 CODED BY AFFINE TRANSFORM	'10'	3	6	3	6
R_8 SPLIT INTO FOUR R_4	'11'				
R_4 CODED BY MEAN μ_R	'0'		6		
R_4 CODED BY AFFINE TRANSFORM	'1'	3	6	3	6

Table 1. Header and bit allocation for both parent and child range blocks.

FRACTAL CODING SCHEME	DOMAIN POOL SIZE				AVERAGE TIME
	16	64	256	1024	
CONVENTIONAL (SUBSAMPLING)	13.2	16.7	19.0	15.5	16.1
CONVENTIONAL (NEIGHBORING)	13.4	12.2	14.7	20.9	15.3
ITERATIONAL-FREE (BLOCK-AVERAGING)	1.9	2.0	1.9	2.0	1.95

Table 2. Decoding time (in seconds) for decoding the Lena image.

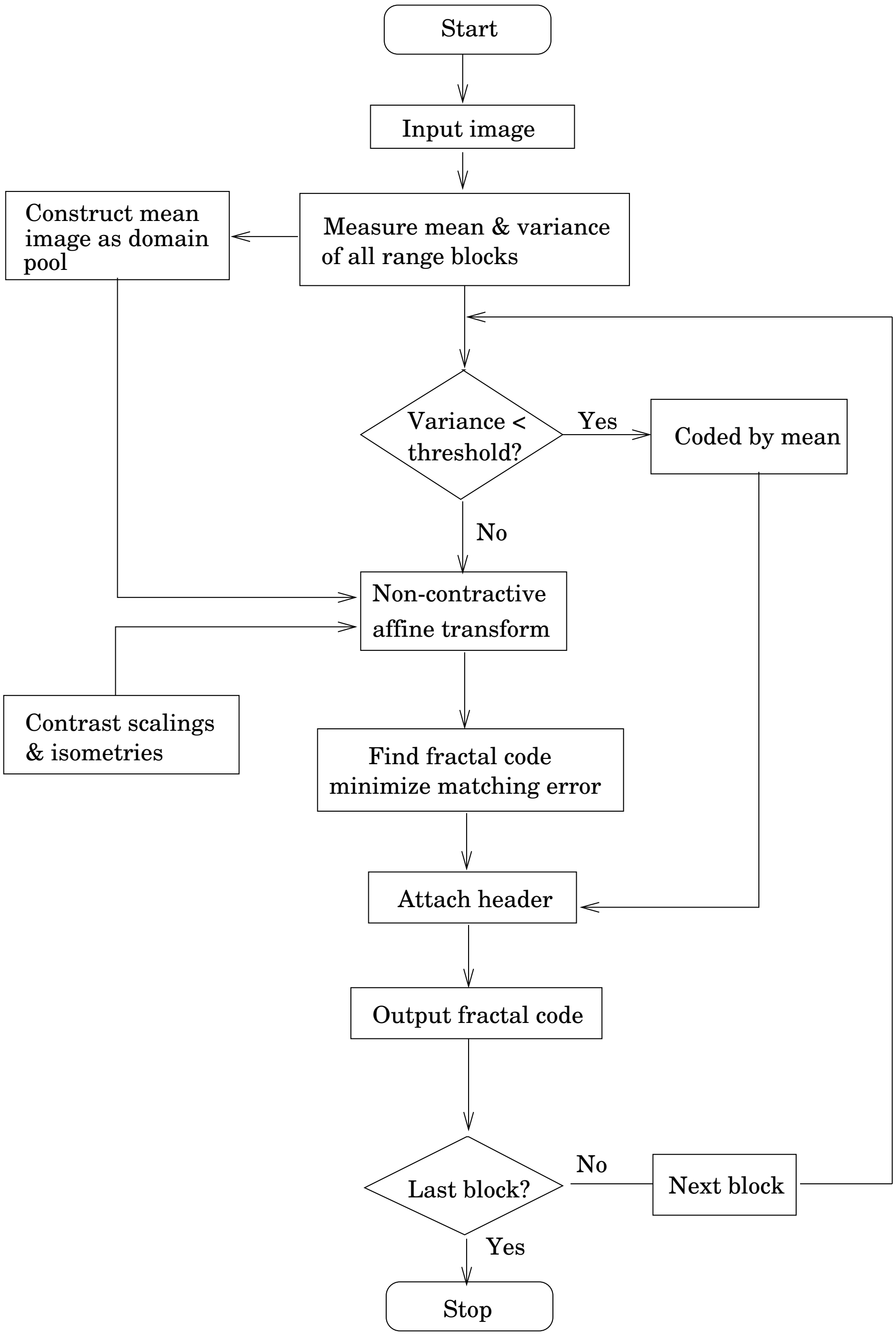


Figure 1: The flow chart of the encoder for the proposed iteration-free coding scheme.

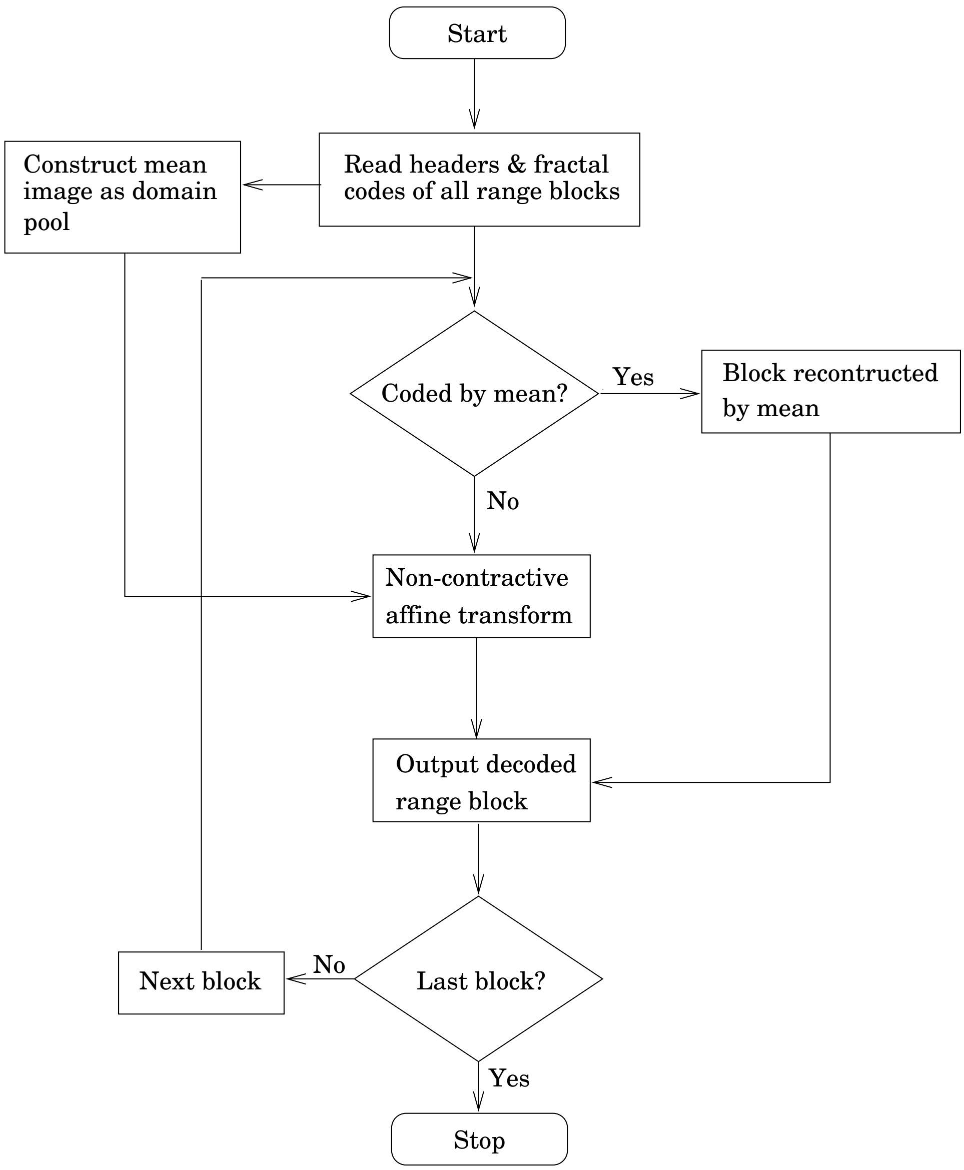
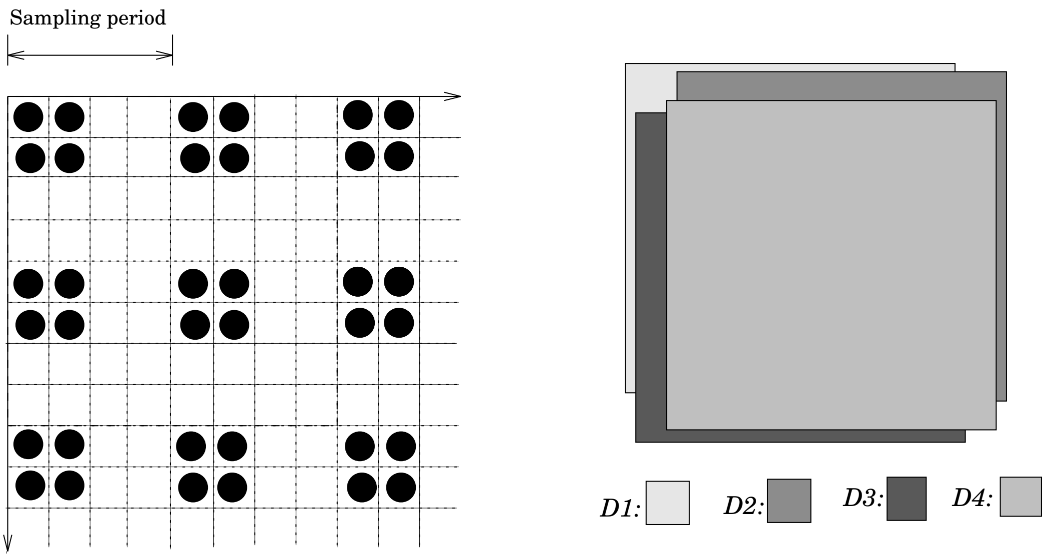


Figure 2: The flow chart of the decoder for the proposed iteration-free coding scheme.



(a) (b)

Figure 3: (a) Some sets of the image blocks used to generate the domain blocks in the block-averaging method; (b) The relative position for four neighboring and partly overlapped image blocks $D1 \sim D4$.

(a)

(b)

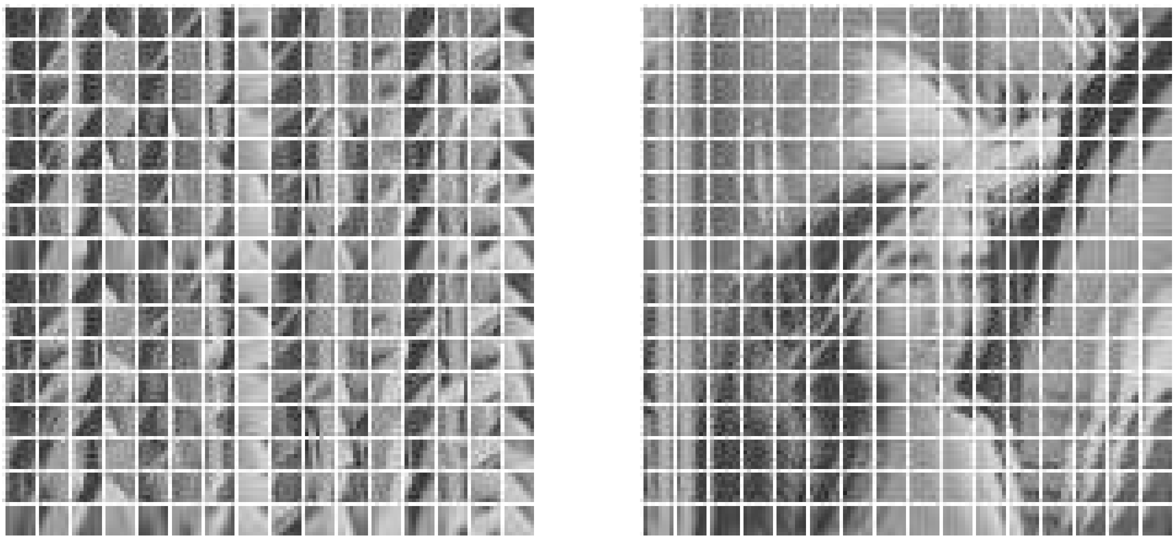
(c)

(d)

Figure 4: The original images (512×512, 8 bit/pixel) used in the proposed iteration-free fractal coding scheme: (a) Lena, (b) Jetplane, (c) Building, and (d) Harbour.

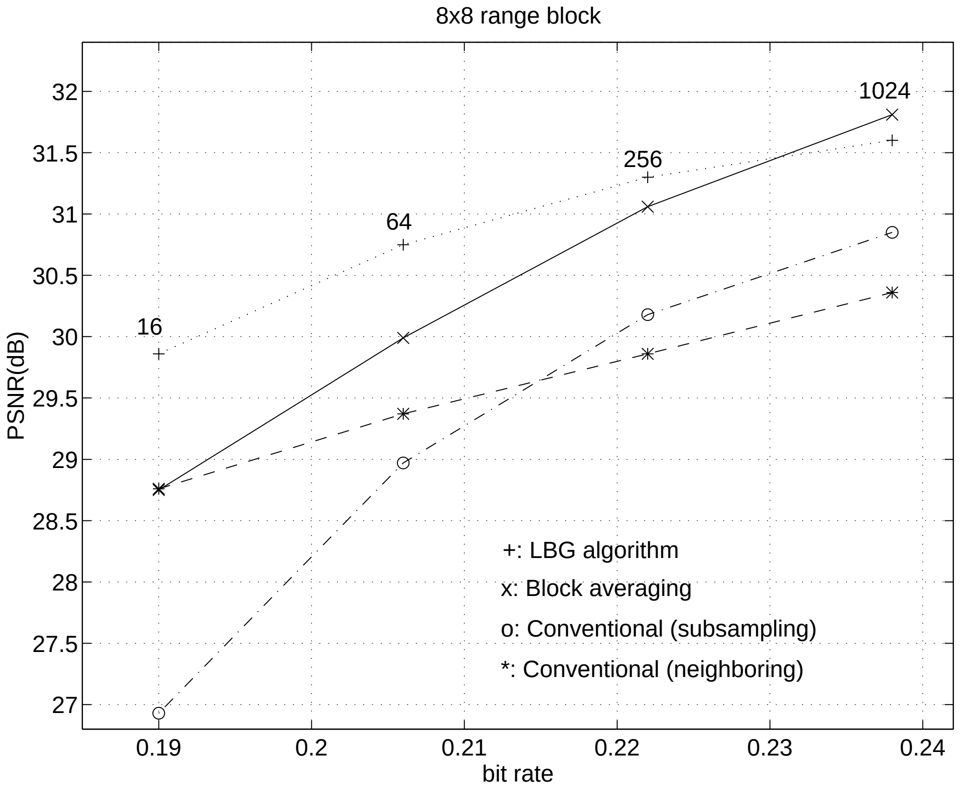
(a) (b)

Figure 5: Two mean images of size (a) 64×64 and (b) 128×128 for the Lena image.

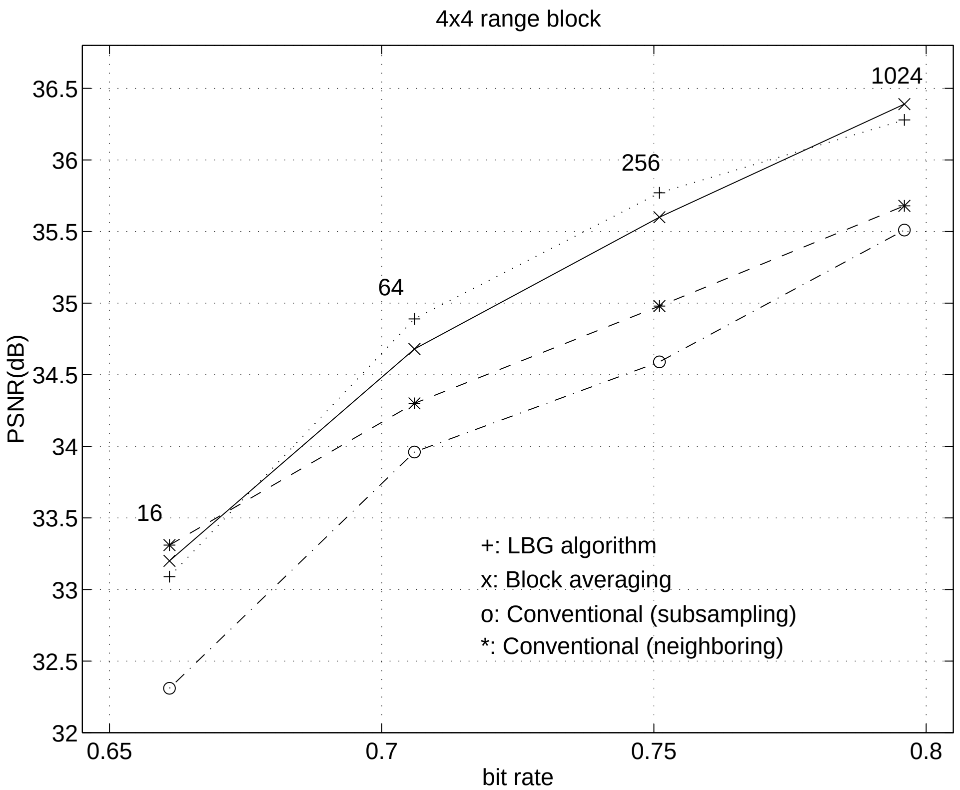


(a) (b)

Figure 6: The constructed domain pools (consist of 256 domain blocks) from the mean image shown in Fig. 5(a) by using (a) the LBG-based method, (b) the block-averaging method.

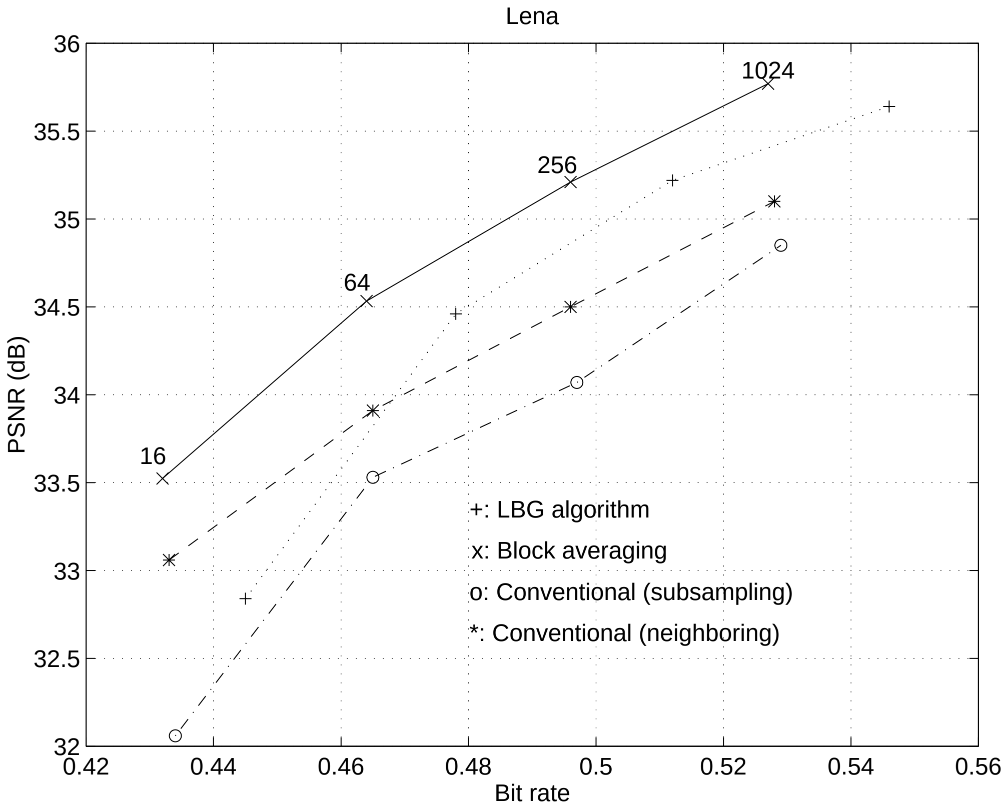


(a)

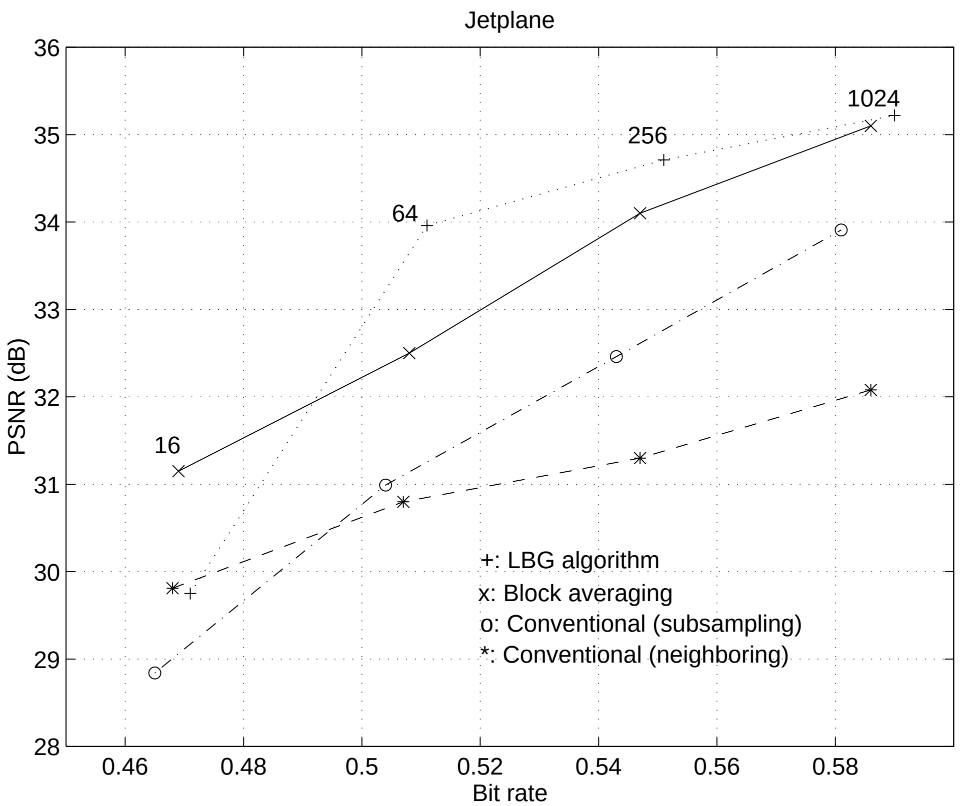


(b)

Figure 7: Coding results of the proposed iteration-free scheme using the single-size design for the range block: (a) 8×8 (b) 4×4.

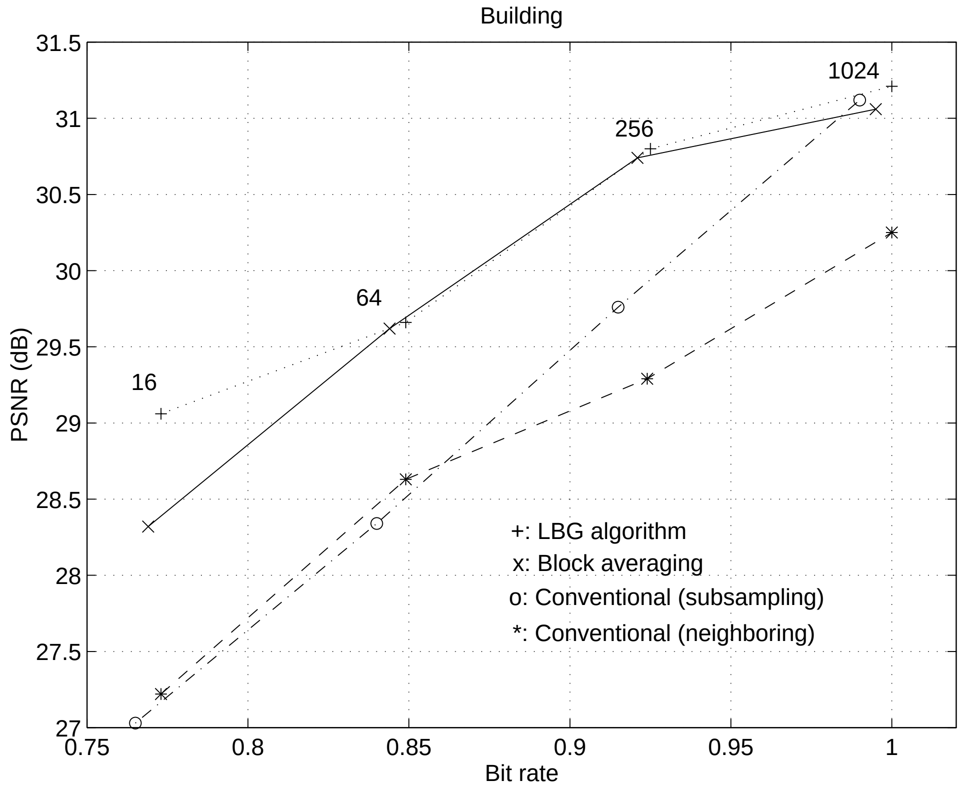


(a)

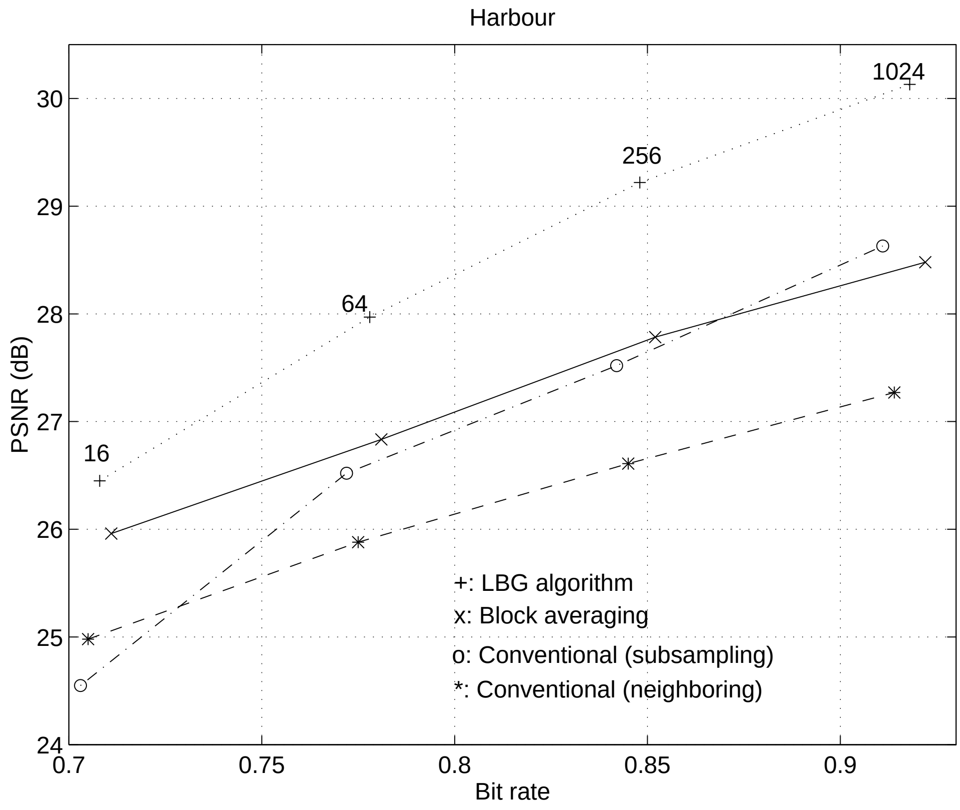


(b)

Figure 8: Coding results of the proposed iteration-free scheme using two-level sizes for the range block. (a) Lena, (b) Jetplane, (c) Building, (d) Harbour.


Building
31.5
31
30.5
30
29.5
29
28.5
28
27.5
27
0.75
0.8
0.85
0.9
0.95
1
Bit rate
PSNR (dB)
1024
256
64
16
+: LBG algorithm
x: Block averaging
o: Conventional (subsampling)
*: Conventional (neighboring)

(c)


Harbour
30
29
28
27
26
25
24
0.7
0.75
0.8
0.85
0.9
Bit rate
PSNR (dB)
1024
256
64
16
+: LBG algorithm
x: Block averaging
o: Conventional (subsampling)
*: Conventional (neighboring)

(d)

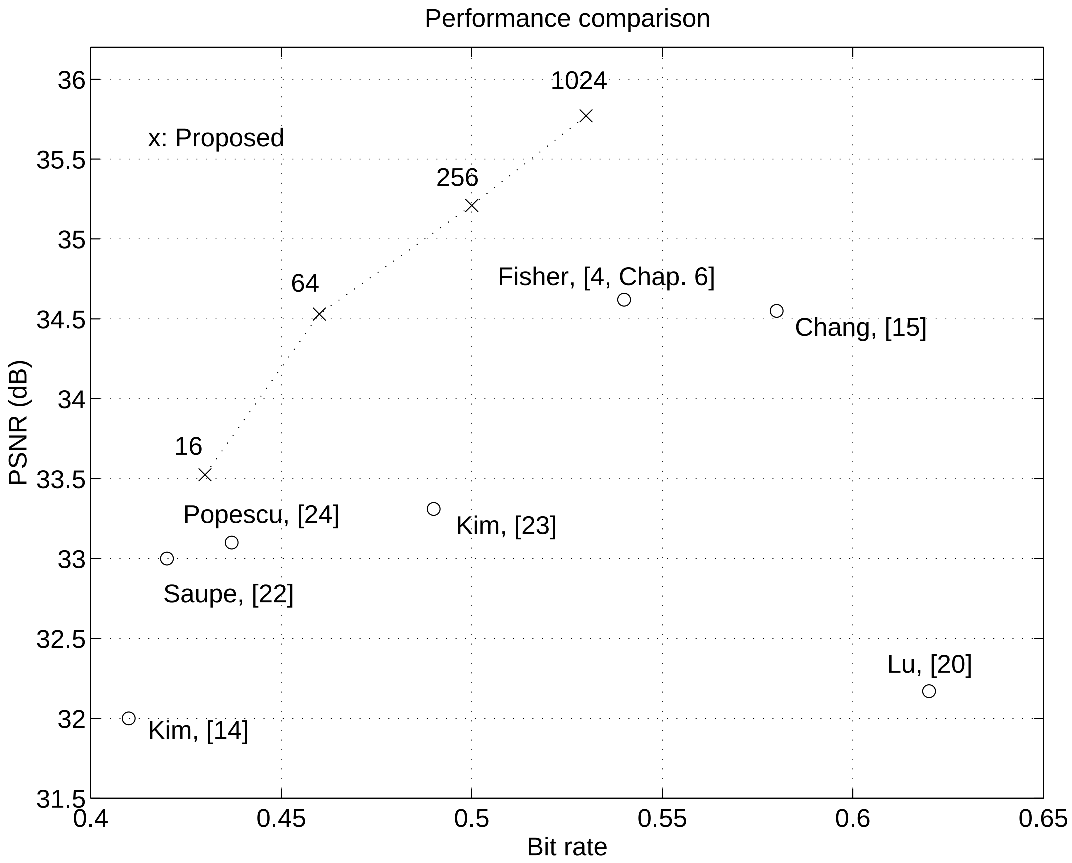


Figure 9: Performance comparison of the Lena image for the proposed iteration-free scheme and other fractal coding schemes.

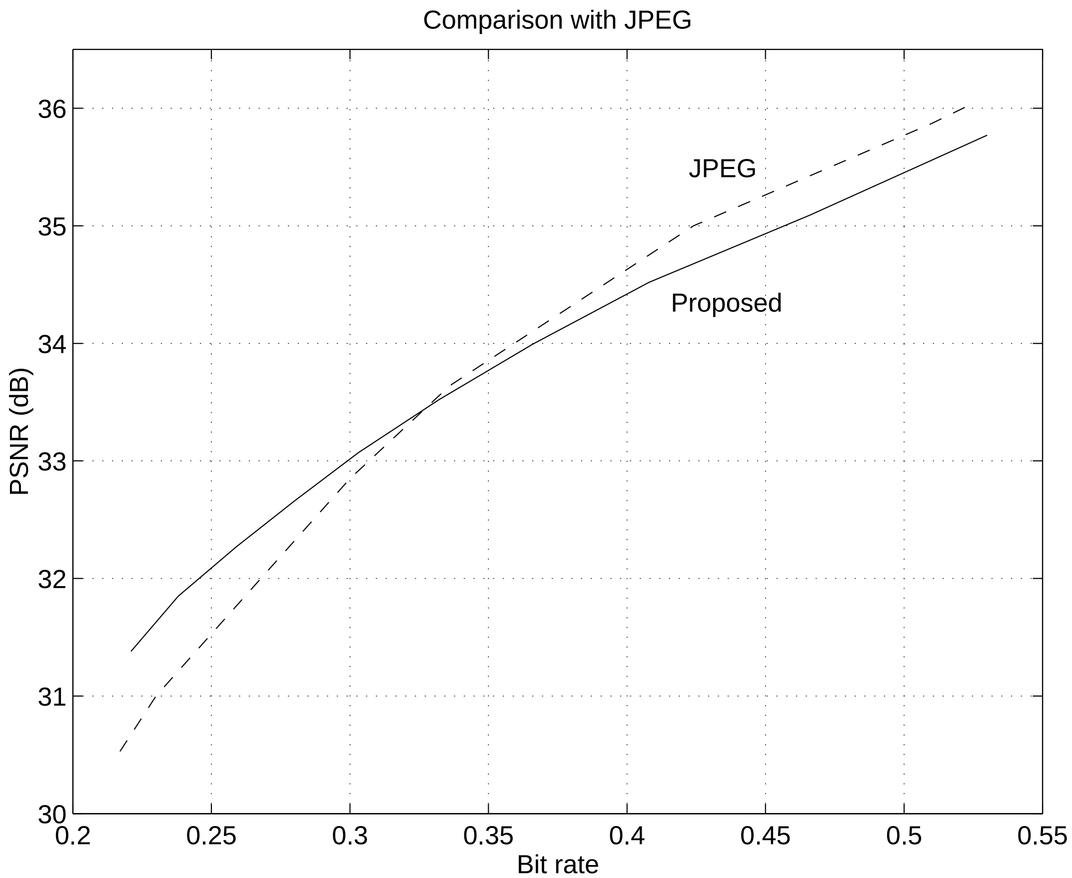


Figure 10: Performance comparison of Lena image between the proposed iteration-free scheme and JPEG image standard.