Modified No Search Scheme based Domain Blocks Sorting Strategies for Fractal Image Coding

Xing-Yuan Wang, Dou-Dou Zhang, and Na Wei

Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, China

Reprint requests to X.-Y. W.; E-mail: wangxy@dlut.edu.cn

Z. Naturforsch. **69a**, 511 – 520 (2014) / DOI: 10.5560/ZNA.2014-0042
Received January 7, 2014 / revised May 10, 2014 / published online August 13, 2014

A novel fractal image coding algorithm based on domain blocks sorting strategies and modified no search scheme is proposed in this paper. On one hand, in order to improve the encoding time, a modified no search (MNS) scheme is adopted. Firstly, the image is divided into blocks of different size utilizing an adaptive quadtree partition method. Secondly, one finds the location of the best matching domain block using the MNS scheme for the range blocks, whose sizes are larger than the preset minimum value. Thirdly, the types of the range block and domain block are computed employing the proposed approach, and then the corresponding computation of mean square error (MSE) is determined. The computation of the MSE is reduced and the encoding phase speeds up. On the other hand, the range blocks with the minimal sizes are encoded applying the proposed domain blocks sorting (DBS) method. Contrast experiment results show that the proposed algorithm can obtain good quality of the reconstructed images and shorten the encoding time significantly.

Key words: Fractal Image Coding; Adaptive Quadtree; Partition; No Search Scheme; Classification; Domain Blocks Sorting.

1. Introduction

At present, fractal image compression (FIC) has become one of the most promising encoding technology in the new generation of image compression for its novel idea, high compression ratio, and resolution independence. The idea of fractal image compression based on iterated function system (IFS) was first introduced by Barnsley [1] in 1988 and Jacquin implemented the practical coding algorithm by using a partitioned iterated function system (PIFS) in 1992 [2]. For the traditional encoding method, i. e. full search method, each range block needs to find a best-matched domain block from the domain pool, which must do a large amount of similarity computations in order to find the best-matched domain block, so it is time-consuming. Moreover, in order to obtain global optimization, the absolute positions of the matching blocks have to be recorded and thus the fractal codes require a large space for saving. As mentioned above, the primary goals of fractal image compression are to reduce the encoding time.

Hence some researches present many accelerated encoding techniques in an effort to reduce the huge encoding time. In Fisher's classification method [3], the image blocks were constructed in 72 classes according to the variance and intensity. This method reduced the encoding time but the arrangement of the 72 classes was complex. Duh et al. [4] introduced an adaptive fractal coding based on discrete cosine transform (DCT) coefficients. The thresholds were determined adaptively and could guarantee a stable speedup ratio of 3 in their paper. However, it is concluded from their paper that the encoding time was still high, and the compression ratio is small. In 2011, Lin and Wu [5] proposed a search strategy using the edge property of the image block. This algorithm has demonstrated better performance than Duh's method. Wu et al. [6] present a fast fractal image encoding method based on intelligent search of standard deviation. Compared to the paper of Tong and Pi [7], their method obtained a significant improvement without significant loss in the reconstructed image quality. In 2007, Zhou et al. [8] classified the image blocks using the unified feature, but the compression ratio was less than 7 in their paper.

In order to improve the encoding time largely, some methods based on spatial correlation and no search

strategies [9 – 14] have been proposed. Wang et al. [10] first proposed a hybrid method combining the genetic algorithm and spatial correlation in 2009. The encoding time in Wang's paper [10] was only half of the spatial correlation genetic algorithm (SC-GA) method proposed by Wu et al. [11]. In 2010, Wang et al. [14] constructed a no search image coding method based on a fitting plane. In comparison to the method of Furao and Hasegawa [12] and Wang and Wang [13], the compression ratio, quality, and encoding time are better in paper [14]. Based on Wang's fitting plane-based fractal image compression using least square regression (LS-FPFIC) in [14], Lu et al. [15] presented an efficient Huber fitting plane-based fractal image compression (HF-PFIC) method.

Another speed-up technique [16 – 18] utilizing the evolutionary strategy has been introduced in recent years. In particular, the particle swarm optimization (PSO) is focused gradually. Tseng and Hsieh [17] proposed a PSO method by using the visual information of the edge property in 2008. Meanwhile some other academic, such as Lin and Chen [18], have been focused on studying the self-similarity of the images after the wavelet transforms. In 2012, Lin and Chen [18] adopted the PSO method performed under classification using the third-level wavelet coefficients. The method reduced the searching space, but it also required large amount of computations, and the compression ratio was small. In papers [19 – 21], the algorithm is implemented using hybrid schemes such as fuzzy clustering and wavelet etc. Lu et al. [22] proposed an enhanced fractal predictive denoising algorithm for denoising the images corrupted by an additive white Gaussian noise by using a quadratic gray-level function.

In this paper, an efficient algorithm is presented. Firstly, one partitions the original image into range blocks of different size using an adaptive quadtree partition approach. Secondly, the proposed modified no search (MNS) and domain blocks sorting (DBS) schemes are adopted for the various range blocks, respectively. It is shown that this algorithm can reduce the encoding time largely and obtains a good reconstructed quality.

The paper is organized as follows: we introduce the conventional full search fractal image compression algorithm and quadtree partition scheme in Section 2. Section 3 describes the DBS scheme and MNS schemes, and Section 4 presents the simulation experimental results. Finally, a conclusion is made in Section 5.

2. Background

2.1. Full Search Fractal Image Compression

The conventional fractal image compression algorithm is based on a contractive affine transformations and partition iteration function system. It includes two stages: encoding and decoding.

During the encoding stage, for simple, we assume that the original image size is 512×512. Firstly, we partition the image into non-overlapping subblocks of size 8×8, called range block pool $\{R\}$. Then, we partition the image into overlapping subblocks of size 16×16 in accordance with step length $\delta = 8$, called domain block pool $\{D\}$. Secondly, the whole domain blocks are contracted into blocks of size 8×8 by averaging four pixel values to one pixel value (AFO). Figure 1 illustrates the arrangements of R and D blocks. Finally, the domain blocks after AFO are extended with eight symmetrical transformations (identity T_0, 90° clockwise rotation T_1, 180° clockwise rotation T_2, 270° clockwise rotation T_3, x reflection T_4, y reflection T_5, $y = x$ reflection T_6, and $y = -x$ reflection T_7) depicted in Figure 2, which form the extended domain block pool $\{\tilde{D}\}$.

In the extended domain pool $\{\tilde{D}\}$, to find the best matched D block for every R block, an affine contractive mapping transformation $\{W_i\}$ is defined as follows:

$$W_i \begin{bmatrix} x \\ y \\ z \end{bmatrix} \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ o_i \end{bmatrix}, \tag{1}$$

where x, y are the spatial coordinates, and z is the pixel value; a_i, b_i, c_i, d_i denote one of the eight symmetrical transformations; s_i controls the contrast, and $s_i < 1$; o_i controls the brightness. (e_i, f_i) is the coordinate of the domain block in the domain pool.

The best matching search between R block and D block in $\{\tilde{D}\}$ is to minimize the following equation:

$$E(R, \tilde{D}) = \left|\left| R - (s \cdot \tilde{D} + o \cdot I) \right|\right|, \tag{2}$$

where $||\cdot||$ is the two-norm. I is the constant vector with elements that are all ones.

In order to minimize (2), the coefficients s_i and o_i can be inferred by the least squares method:

$$s_i = \frac{n\sum_{i=1}^{n} d_i r_i - \left(\sum_{i=1}^{n} d_i\right)\left(\sum_{i=1}^{n} r_i\right)}{n\sum_{i=1}^{n} d_i^2 - \left(\sum_{i=1}^{n} d_i\right)^2}, \tag{3}$$

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

where n denotes the number of the pixel values; d_i is the pixel value in each $\tilde{D}$ block, and r_i is the pixel value in each R block.

Finally, the location x, y of the $\tilde{D}$ block, coefficients T of the symmetrical transformation, and s_i and o_i form the fractal code for each R block.

The decoding of the fractal image compression algorithm is very simple. Firstly, one chooses an arbitrary image as the original image and performs 4096 affine transforms based on the obtained fractal codes on it. Then the process is proceeded recursively. Finally, we stop the recursion when the decoded image meets the user's needs.

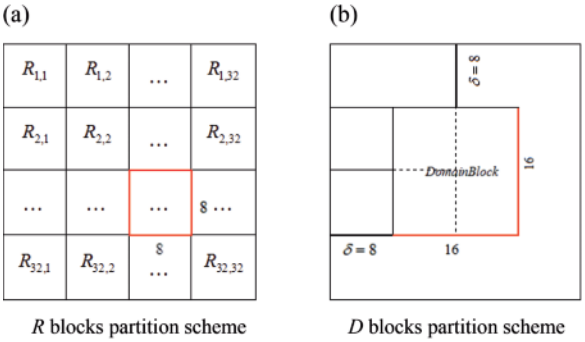

Fig. 1 (colour online). Arrangements of R and D blocks: (a) R blocks partition scheme, (b) D blocks partition scheme.

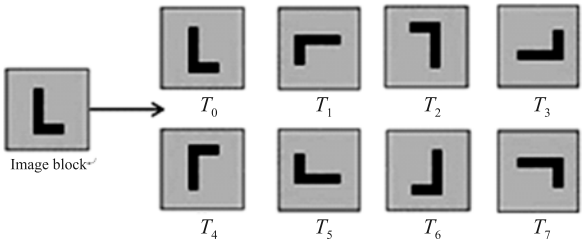

Fig. 2. Eight transformations of an image block.

2.2. Quadtree Partition Scheme

In the traditional fractal algorithm, the image is partitioned into fixed-size image blocks. However, it can't demonstrate the feature of the image. Actually, if the size of the image blocks is large, it is difficult to find the best matching blocks, specifically for the fine and complex area, such as Lena's eyes. Moreover, the quality of the retrieved image is poorer. On the other hand, if the block size is much smaller, it can obtain good peak signal-to-noise ratio (PSNR) of the reconstructed image. But the number of the R blocks increases, which restricts the compression ratio and enhances the encoding time.

In order to solve the problems mentioned above, Fisher [3] proposed the quadtree partition scheme. It is shown that if the matching error is larger than the set threshold, the current R block would be divided into four different level sub-blocks of the same size. Then each sub-block finds the best matching D block repeating the previous process. Otherwise, if the matching error is smaller than the threshold, then encode the R block directly.

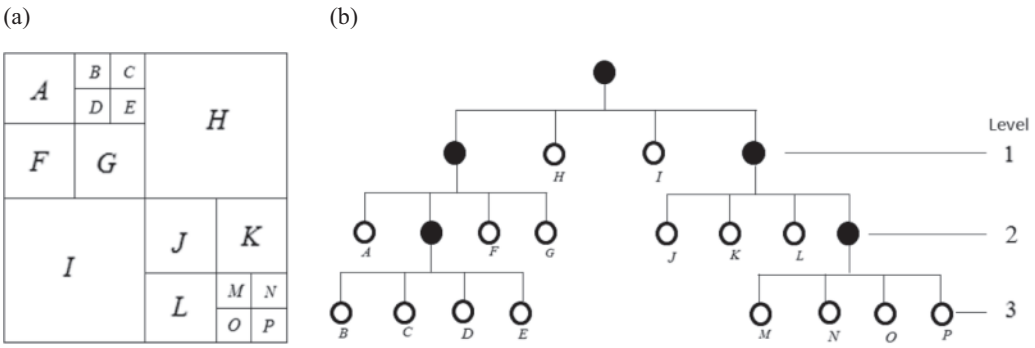

Fig. 3. Process of the qaudtree partition: (a) the sketch of the partition, (b) the tree structure of the partition.

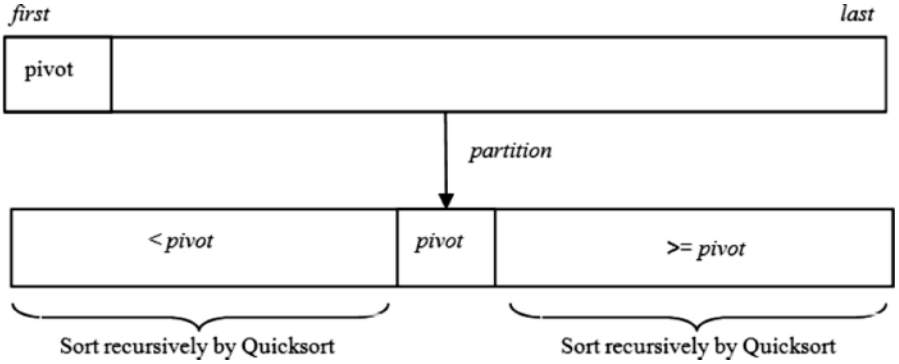

Fig. 4. Quicksort.

Tong and Pi [7] introduced the quadtree partition search method based on an adaptive threshold. In comparison to the fixed threshold partition scheme, the PSNR of the rebuilt image utilizing Tong and Pi [7] algorithm is better. In their paper, if the threshold of the ith level is t_i, then the threshold of the next level is determined by

$$t_{i+1} = kt_i\,. \tag{5}$$

k is set to 2 in the paper. Figure 3 demonstrates the process of the partition of three levels.

3. Proposed Improved Fractal Image Encoding Algorithm

3.1. Domain Blocks Sorting (DBS) Scheme

According to (3) and (4), (2) can be inferred as

$$E(R,D)^2 = \left|\left|R-\bar{R}\cdot I\right|\right|^2 - s^2\left|\left|D-\bar{D}\cdot I\right|\right|^2. \tag{6}$$

From (6), let

$$\sigma_Y = \frac{\left|\left|Y-\bar{Y}\cdot I\right|\right|}{\sqrt{n}}\,,$$

we can get

$$E(R,D)^2 = n\left(\sigma_R - s^2\sigma_D\right), \tag{7}$$

where n denotes the number of the pixel values; σ_Y indicates the variance for Y. I is the constant vector with elements that are all ones.

Since $E(R,D)^2 \geq 0$, then $s^2 \leq \sigma_R/\sigma_D$. We can obtain that the larger σ_D is, the more s meets $s \in (-1,1)$. In other words, the larger σ_D is, the better the reconstructed quality of the image is. Therefore, the D blocks, whose variances are smaller than the critical threshold for D blocks T_d, would be discarded and only these satisfying the condition will form the searching space.

In order to further reduce the encoding time, the D blocks in the search space will be sorted, which is referred to as the reduced search space Ω_r. Many of the best-known applications of the divided-and-conquer algorithm design paradigm are sorting algorithms. We adopt Quicksort algorithm as the sorting strategy. Quicksort is one of the earlier divide-and-conquer algorithms. In the paper, we assume that the key of each element in the set to be sorted is the σ_D value of the domain block. Other information of each element in the set is aside from the key, such as the location of the domain blocks. Quicksort's strategy is to rearrange the elements to be sorted so that all the 'small' keys precede the 'large' keys in the set. Then Quicksort sorts the two subranges of 'small' and 'large' keys recursively, with the result that the entire set is sorted.

Let first and last be the indexes of the first and last entries, respectively. The Quicksort algorithm chooses an element, called the pivot element. Quicksort passes the pivot to the partition subroutine, which is depicted in Figure 4.

After that, each R block searches the best matching D block using the dichotomy method in Ω_r based on the preset matching threshold value k.

A dichotomy method or half-interval search algorithm finds the position of a search key value within an array sorted by key value. For the dichotomy method, the array should be arranged in ascending or descend-

ing order. In each step, the algorithm compares the search key value with the key value of the middle element of the array. If the keys match, then a matching element has been found and its index is returned. Otherwise, if the search key is less than the middle element's key, then the algorithm repeats the step on the sub-array to the left of the middle element or, if the search key is greater, on the sub-array to the right. If the remaining array to be searched is empty, then the key cannot be found in the array and a special 'not found' indication is returned.

In the paper, each R block denotes the search key value and the reduced search space Ω_r after using the Quicksort method indicates the sorted array. Moreover, the definition of 'match' does not mean that the search key value is equal to the element in the array, but approximately equal. More specifically, if the matching error between the R block and the D block in Ω_r is less than the preset value, then they are 'equal'.

Meanwhile, if σ_R is smaller than the critical threshold for R blocks T_r, it indicates that the brightness of the R block fluctuates little. We define this R block as smooth block and store its pixel average value as the fractal code. Therefore, the encoding process would be speed up greatly. The detailed steps are depicted as follows:

Step 1: Initialize all the parameters.

Step 2: Generate the R block pool and D block pool.

Step 3: Contract every D block to the equal size of the R block.

Step 4: In each searching entry, compute the variance of the R block V_r and compare it with the value T_r: if $V_r < T_r$, which is named as smooth R block R_s, store the average pixel value as its code and go to Step 5; otherwise the R block is named as R_n block and search the best matching D block in Ω_R.

Step 5: Repeat Step 4 if there are still R blocks not being encoded.

3.2. A Modified no Search (MNS) Algorithm

Furao and Hasegawa [12] presented a fast no search fractal image coding method. In this paper, for most R blocks, their best matching space is located at the $(\mathrm{row}_R - B/2, \mathrm{col}_R - B/2)$, the probability of which is greater than other location, where row_R and col_R are the position of the R block; B is the R block size. In order to reduce the amount of MSE computation further, a modified no search algorithm is presented:

$$F(m,n) = \frac{2}{N} C_m C_n \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(i,j) \cos\left(\frac{(2i+1)m\pi}{2N}\right) \cdot \cos\left(\frac{(2j+1)n\pi}{2N}\right), \tag{8}$$

where $m,n = 0,1,\ldots,N-1$, and

$$C_k = \begin{cases} 1/\sqrt{2} & \text{if } k = 0 \\ 1, & \text{else} \end{cases}.$$

Based on Chang and Kang's paper [21], in which a classification method utilizing edge properties was introduced, the types of the R blocks are computed by (9). $F(0,1)$, $F(1,0)$, and $F(1,1)$ are three DCT coefficients computed by (8).

$$\begin{aligned} \lambda_0 &= \left|F(0,1)\right|, \\ \lambda_1 &= \frac{2}{3} \max\left\{ \left|F(0,1)+F(1,0)+F(1,1)\right|, \right. \\ &\qquad \left. \left|F(0,1)+F(1,0)-F(1,1)\right| \right\}, \\ \lambda_2 &= \left|F(1,0)\right|, \\ \lambda_3 &= \frac{2}{3} \max\left\{ \left|F(0,1)-F(1,0)+F(1,1)\right|, \right. \\ &\qquad \left. \left|F(0,1)-F(1,0)-F(1,1)\right| \right\}. \end{aligned} \tag{9}$$

Find the maximum value in $\{\lambda_0, \lambda_1, \lambda_2, \lambda_3\}$. If the value obtained is λ_0, R is marked as I class; else if the value obtained is λ_1, R is marked as II class; else if the value obtained is λ_2, R is marked as III class; else the value obtained is λ_3, R is marked as IV class.

Table 1. Selection of transformations T_k for eight combinations.

Type of R block	Type of D block	Transformation T_k
I	I	T_0, T_2, T_4, T_6
	III	T_1, T_3, T_5, T_7
	II, IV	Quadtree partition for R
II	II	T_0, T_2, T_5, T_7
	IV	T_1, T_3, T_4, T_6
	I, III	Quadtree partition for R
III	I	T_1, T_3, T_5, T_7
	III	T_0, T_2, T_4, T_6
	II, IV	Quadtree partition for R
IV	II	T_1, T_3, T_4, T_6
	IV	T_0, T_2, T_5, T_7
	I, III	Quadtree partition for R

The MNS algorithm is performed as follows:

At each search entry, if the location of R marked as 'Level' is $(\text{row}_R,\text{col}_R)$, the corresponding best matching D block is located at $(\text{row}_R - B/2, \text{col} - B/2)$. When finding the D block, compute its type exploiting (9). Therefore, there are 16 combinations between D and R block. Among these 16 combinations, eight of them will produce smaller MSE values. For example, if both the D and R blocks are I class, the transformation T_1 will change the D block to a block with III class. In this case, they have large MSE, and so do T_3, T_5, and T_7. Therefore, one will confine the search only on T_0, T_2, T_4, and T_6. The possible transformations for the pairs of R and D block are list in Table 1. Then if the error matching between R and D blocks is larger than the scale threshold value T, partition the R block and mark it as 'Level + 1'. For the other eight combinations, the R block will be divided into four sub-blocks of the same size directly, which are marked as 'Level + 1'. Repeat the process until all the R blocks are encoded.

Obviously, this method can reduce the encoding time significantly. Because of not searching the location of the D blocks using the no search method, each R block doesn't need to record the position code, which improves the compression ratio largely. Combining with the DBS scheme, the retrieved images could obtain perfect qualities. The experimental results mentioned in Section 4 also prove the effectiveness of the proposed method.

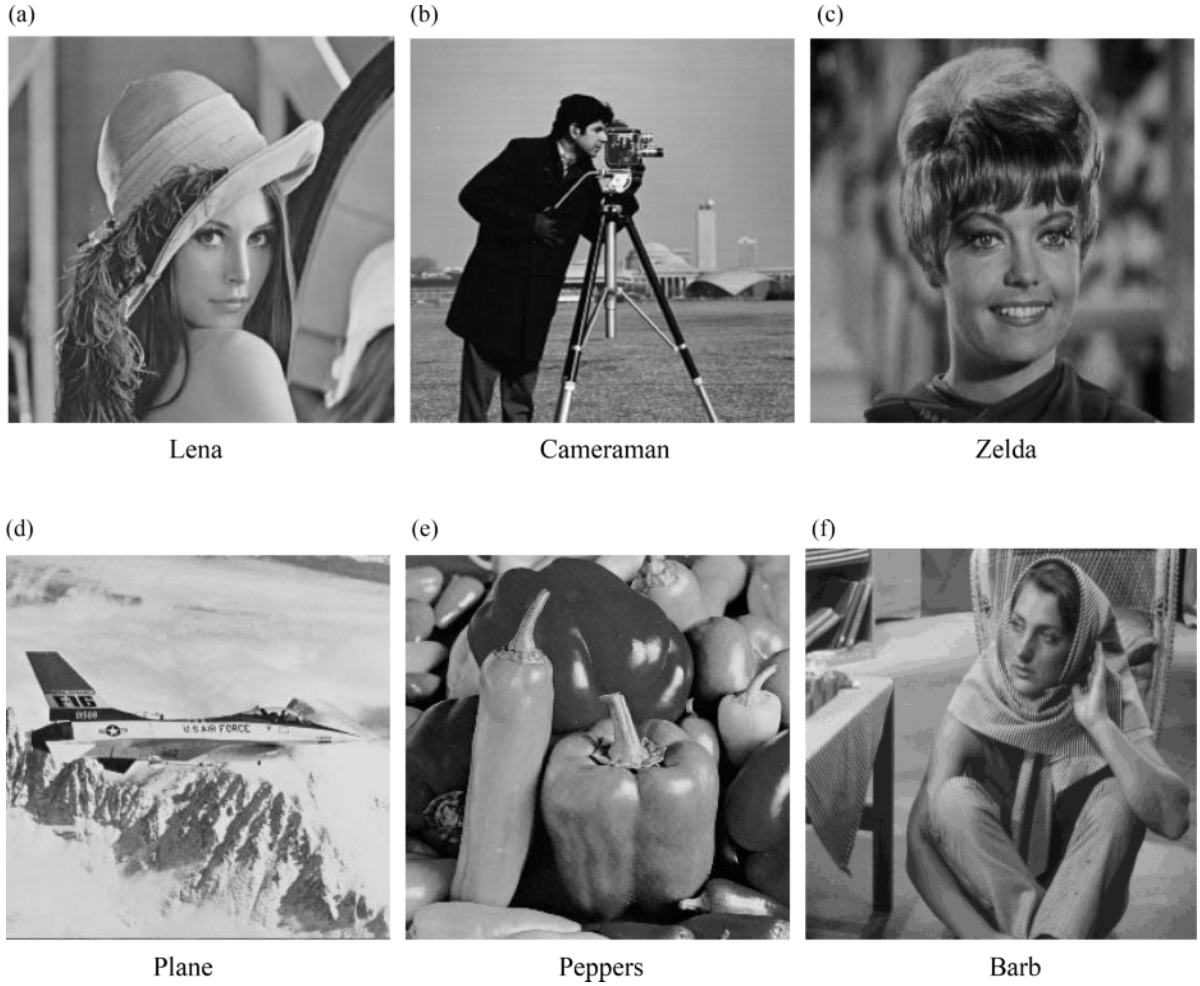

Fig. 5. Six original images: (a) Lena, (b) Cameraman, (c) Zelda, (d) Plane, (e) Peppers, and (f) Barb.

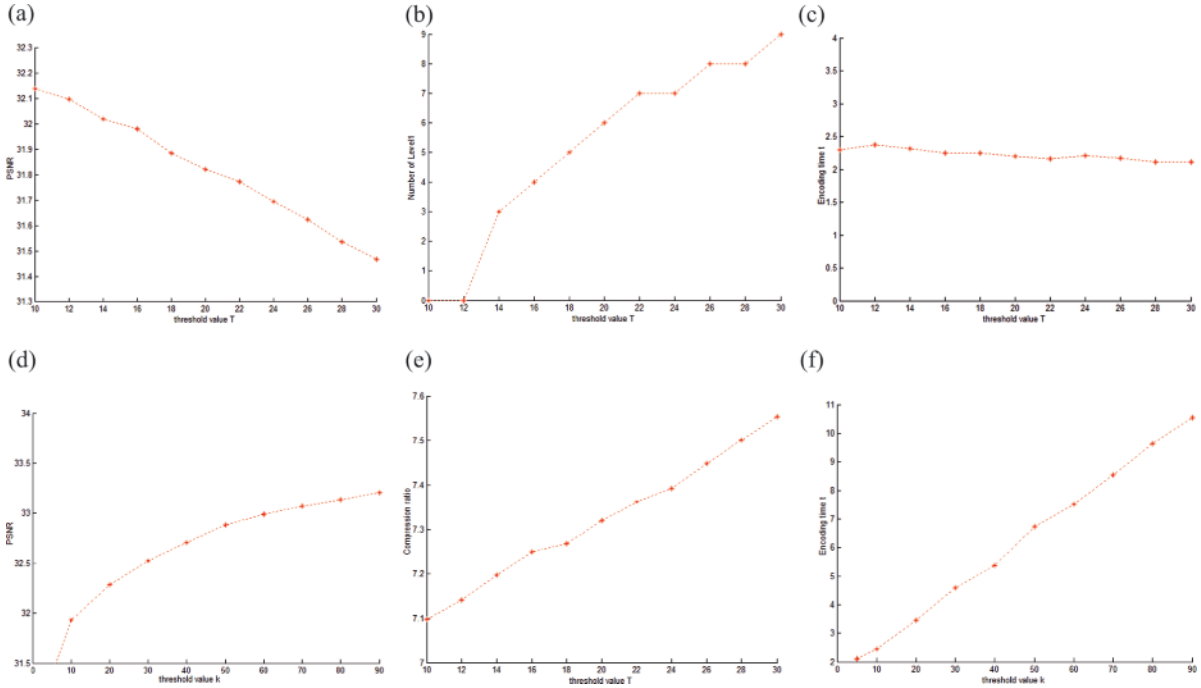

Fig. 6 (colour online). Experimental results using the proposed method: (a) Scale threshold value T, (b) Scale threshold value T, (c) Scale threshold value T, (d) Matching threshold value k, (e) Scale threshold value T, (f) Matching threshold value k.

3.3. The Proposed Method based on DBS and MNS Scheme

If we pay more attention to the reconstructed quality and encoding time, a method based on DBS and MNS scheme is proposed. The detail process is depicted as follows:

Step 1: Utilizing the adaptive quadtree partition scheme mentioned above, the original image is divided into the same size R blocks. In this paper, the initial block size is set to 16, and the R blocks are marked as 'Level 1'. The permitted minimum size is 4 and the R block is marked as 'Level 3'. The R block of intermediate size 8 is marked as 'Level 2'. The initial R blocks form the range pool.

Step 2: Design the searching space utilizing the proposed DBS scheme. The size of D blocks is set to 8 and the step length is 4. Moreover, one must record the location (d_x, d_y) and the variance values of the D blocks in the searching space.

Step 3: For both Level 1 and Level 2 R blocks, we utilize the proposed MNS scheme to search the best matching D block. If $E(R,D) \leq T$ (T is the preset scale threshold value), then we directly store the obtained fractal code. Otherwise if $E(R,D) > T$, we divide the R block using the adaptive quadtree partition scheme mentioned above. If the partitioned R blocks size is larger than 4, then go to Step 3. Otherwise go to Step 4.

Step 4: For Level 3 R blocks whose size is 4, we search the best matching D block utilizing the proposed DBS scheme mentioned in Section 3.1.

Step 5: If still there are R blocks which have not been coded, go to Step 2; otherwise end the encoding process.

Practically, for every R block of Level 1 or Level 2 encoded using MNS scheme, one needs store 13 bits (2 bits for marking the level, 3 bits for T, and 8 bits for the shift of pixel value) as its fractal code. The Level 3 R blocks are classified in two types: R_s and R_n. One utilizes 27 bits (14 bits for the location of the best matching block, 3 bits for T, 2 bits for marking the level, and 8 bits for the shift of pixel value)

for R_s and 10 bits (8 bits for shift of pixel value and 2 bits for marking the level) for R_n as the fractal code.

The reconstruction phase is a relative simple iterative process. The procedure is similar to what is mentioned in Section 2.

4. Experiment

The proposed algorithm is conducted on an Intel (R) Core2 Duo 2.0GHz, 1.99GB RAM, Windows XP PC. In the experiment, six international standard images of size 512×512 are used (Lena, Cameraman, Zelda, Plane, Pepper, and Barb). Figure 5 shows the six various original images. In order to demonstrate the good performance of the proposed algorithm, one compares it with full search and no search strategies. The programs of all algorithms are implemented using C++ in the same environment to improve the reliability. Table 3 shows the PSNR and encoding time among three methods.

Table 2. Number of different R blocks.

Images	Level 1	Level 2	R_n of Level 3	R_s of Level 3
Lena	9	716	7800	5576
Cameraman	91	878	5979	5437
Plane	57	841	6634	5474
Peppers	32	661	8495	4733
Barb	4	380	10 044	4756
Zelda	23	1000	6701	5315

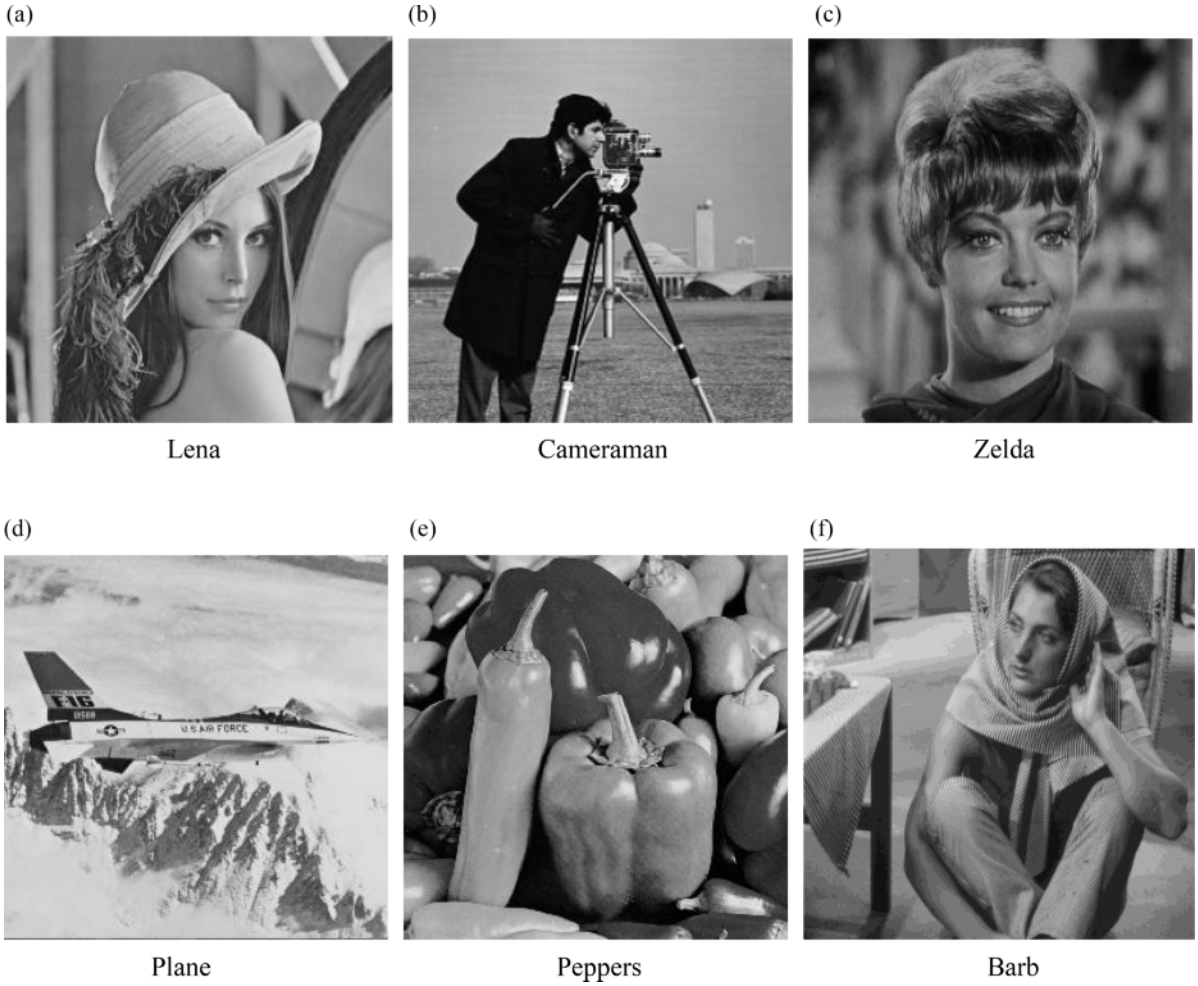

Fig. 7. Reconstructed images: (a) Lena, (b) Cameraman, (c) Zelda, (d) Plane, (e) Peppers, and (f) Barb.

Table 3. Comparison of the three algorithms.

	Full search method		No search method		Proposed method	
	Time	PSNR	Time	PSNR	Time	PSNR
Lena	1927.985	34.223732	0.624	30.643696	2.375	31.504866
Cameraman	1891.500	38.243977	0.564	29.775312	2.015	31.801004
Plane	1891.859	35.997983	0.628	27.619534	2.187	30.313623
Peppers	1945.750	32.599366	0.812	28.865073	2.453	31.240955
Barb	1888.297	37.549792	0.564	25.412168	2.687	25.546100
Zelda	1896.016	31.487453	0.624	34.567123	2.218	32.296183

The quality of the retrieved image is measured by the PSNR, which is given by

$$\mathrm{PSNR} = 10\log_{10}\left[\frac{255^2}{\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\big(f(i,j)-g(i,j)\big)^2}\right]. \quad (10)$$

Figure 6 illustrates the experimental results using the proposed method. According to Figure 6a – c and 6e, we can see that the number of R blocks of Level 1 and the compression ratio increase as T is growing, while the quality of the reconstructed images will decay. Meanwhile, the encoding time almost keeps stable. Different matching threshold value k produces various PSNR and consumes different encoding time, but it doesn't change the compression ratio. From Figure 6d and 6f, it reveals the fact that the encoding time and PSNR coincides with the k value. Table 2 shows the number of R blocks of different levels for various images.

In the paper, the scale threshold value T is set to 32 and the matching threshold value k is 5. When $T_d = 8$ and $T_r = 4$, the reconstructed images using the proposed method are shown in Figure 7. In the full search and no search methods, the size of the R block is 4, that of the D block is 8, and the step length is 4. From Table 3, it reveals the fact that the full search algorithm consumes much more encoding time than the proposed method. Moreover, for the full search scheme, the compression ratio is smaller than the proposed approach, although the PSNR is better. For example, it needs 31 bits for the fractal code of each R block while using the full search method, the compression ratio is 4.129 for a Lena image of size 512×512. Compared to the proposed method, the encoding time for the no search method is slightly faster, but the quality of retrieved images is much worse. In short, take encoding time, PSNR, and compression ratio into consideration, the proposed approach is a better choice.

5. Conclusion

In this paper, we presented a novel fractal image compression algorithm based on domain blocks sorting and modified no search scheme, which improves the conventional fractal image coding using full search method significantly. To demonstrate the coding performance, we further compared our method with the no search method. Firstly, the original image is partitioned into R blocks of size 16×16. Secondly, if the size of the R blocks is larger than 4, a MNS scheme is adopted to search the best matching block; if the size of the R block is 4, one utilizes the proposed DBS method to seek the best D block, which reduces the searching space largely. The results show that this method can speed up the encoding phase greatly and obtains a good reconstructed quality.

Acknowledgements

This research is supported by the National Natural Science Foundation of China (Nos. 61370145, 61173183, and 60973152), the Doctoral Program Foundation of Institution of Higher Education of China (No. 20070141014), Program for Liaoning Excellent Talents in University (No. LR2012003), the National Natural Science Foundation of Liaoning province (No. 20082165), and the Fundamental Research Funds for the Central Universities (No. DUT12JB06).

[1] M. F. Barnsley, Fractal Everywhere, Academic Press, New York 1988.
[2] A. E. Jacquin, IEEE Transact. Image Proc. **1**, 18 (1992).
[3] Y. Fisher, Fractal Image Compression, Theory and Applications, Springer, New York 1994.
[4] D. J. Duh, J. H. Jeng, and S. Y. Chen, Image Vis. Comput. **23**, 1115 (2005).

[5] Y. L. Lin and M. S. Wu, Comput. Math. Appl. **62**, 310 (2011).
[6] X. W. Wu, D. Jackson, and H. C. Chen, Comput. Elect. Eng. **31**, 402 (2005).
[7] C. S. Tong and M. Pi, IEEE Trans. Image Proc. **10**, 1269 (2001).
[8] Y. M. Zhou, C. Zhang, and Z. K. Zhang, Chaos Solit. Fract. **39**, 1823 (2007).
[9] T. K. Truong, C. M. Kung, J. H. Jeng, and M. L. Hsieh, Chaos Solit. Fract. **22**, 1071 (2004).
[10] X. Y. Wang, F. P. Li, and S. G. Wang, J. Vis. Commun. Image Rep. **20**, 505 (2009).
[11] M. S. Wu, W. C. Teng, J. H. Jeng, and J. G. Hsieh, Chaos Solit. Fract. **28**, 497 (2006).
[12] S. Furao and O. Hasegawa, Sig. Proc. Image Commun. **19**, 393 (2004).
[13] X. Y. Wang and S. G. Wang, Comput. Graph. **32**, 445 (2008).
[14] X. Y. Wang, Y. X. Wang, and J. J. Yun, Image Vis. Comput. **28**, 1303 (2010).
[15] J. Lu, Z. Ye, and Y. Zou, IEEE Trans. Image Proc. **22**, 134 (2013).
[16] Y. Zhang, G. R. Liu, and X. X. Niu, Comput. Math. Appl. **51**, 1727 (2006).
[17] C. C. Tseng and J. G. Hsieh, Image Vis. Comput. **26**, 1154 (2008).
[18] Y. L. Lin and W. L. Chen, J. Inf. Sci. Eng. **28**, 17 (2012).
[19] K. Jaferzadeh, K. Kiani, and S. Mozaffari, IET Image Proc. **6**, 1024 (2012).
[20] Y. Iano, F. S. da Silva, and A. M. L. Cruz, IEEE Trans. Image Proc. **98**, 15 (2006).
[21] H. S. Chang and K. Kang, IEEE Trans. Image Proc. **14**, 145 (2005).
[22] J. Lu, Z. Ye, Y. Zou, and R. Ye, Chaos Solit. Fract. **38**, 1054 (2008).