A Novel Fractal Image Compression Scheme with Block Classification and Sorting Based on Pearson's Correlation Coefficient

Jianji Wang, *Student Member, IEEE*, and Nanning Zheng, *Fellow, IEEE*

***Abstract*— Fractal image compression (FIC) is an image coding technology based on the local similarity of image structure. It is widely used in many fields such as image retrieval, image denoising, image authentication, and encryption. FIC, however, suffers from the high computational complexity in encoding. Although many schemes are published to speed up encoding, they do not easily satisfy the encoding time or the reconstructed image quality requirements. In this paper, a new FIC scheme is proposed based on the fact that the affine similarity between two blocks in FIC is equivalent to the absolute value of Pearson's correlation coefficient (APCC) between them. First, all blocks in the range and domain pools are chosen and classified using an APCC-based block classification method to increase the matching probability. Second, by sorting the domain blocks with respect to APCCs between these domain blocks and a preset block in each class, the matching domain block for a range block can be searched in the selected domain set in which these APCCs are closer to APCC between the range block and the preset block. Experimental results show that the proposed scheme can significantly speed up the encoding process in FIC while preserving the reconstructed image quality well.**

***Index Terms*— Fractal image compression, Pearson's correlation coefficient, block classification, block sorting.**

I. Introduction

FRACTAL image compression (FIC) is an image coding technology based on the local similarity of image structure. It was proposed by Michael F. Barnsley in 1988 [1], and improved by Arnaud E. Jacquin in 1992 [2]. Jacquin's FIC scheme is called the baseline fractal image compression (BFIC), which uses the partitioned iterated function system (PIFS) to search the matching block pairs without human-computer interactions. Since then, FIC has a quick development and numerous FIC schemes have been published.

Traditional image coding technologies encode an image by pixel-based and statistical methods, and FIC is an interesting attempt at the structure-based image coding. FIC has been used not only in image coding, but also in some interesting image problems [3]–[9] and pattern recognition problems such as facial recognition [10]. However, compared with traditional image coding technologies, fractal compression suffers from the high computational complexity in encoding.

Generally, one image is firstly partitioned into some square blocks in FIC, and then these square blocks compose a set called the pool. According to two kinds of size, an image is partitioned into two different pools. The pool composed by the blocks with larger size is called the domain pool, and the other pool is called the range pool. The cells in the range pool are the blocks to be encoded. The blocks in the domain pool are contracted to the same size as the range blocks, and then FIC takes the domain pool as a virtual codebook. The matching domain block for each range block needs to be searched. Exhaustive search of the matching block pairs costs too much time, which is one of the major difficulties in BFIC.

Lots of works had been conducted to speed up the process of searching the matching pairs in FIC in recent years [11]–[22]. Some schemes were proposed based on a selected part of the domain pool to reduce the search cost for a range block [12]–[14]. For example, based on the spatial correlation on the range and domain blocks, many fast FIC schemes were proposed by searching the matching domain block from the adjacent domain block of the current range block [13] or from the adjacent domain blocks of the domain block which is the matching block for the adjacent range block of the current range block [14]. Classification was also applied in some fast FIC schemes [15]–[18] to classify the domain blocks into some different classes, then the matching block for a range block can be searched from several classes closely associated with the range block. Fisher's fast FIC scheme proposed in 1992 [15] is a typical classification scheme. Some other strategies, such as synthesis with other technologies [19], and sorting with some features [20], [21], were applied to speed up encoding in FIC.

Although numerous schemes were proposed to speed up encoding in FIC, the encoding time is still too long so far. For example, the encoding time for a 512×512 image with 4×4 range blocks is more than 20 seconds in the DRDC scheme proposed by Riccardo Distasi *et al* in 2006 [18], and the encoding time for a 256×256 image is more than 2.8 seconds in the DUFC scheme proposed by Yi-Ming Zhou *et al* in 2009 [19]. Of course, it should be considered that the number of pairwise block-comparisons for a 512×512 image is 16 times more than the number of pairwise comparisons for a 256×256 image in BFIC with the same partition. Although these schemes should not be compared without the same computer environment, they still show that there

Manuscript received October 1, 2012; revised April 1, 2013; accepted May 25, 2013. Date of publication June 17, 2013; date of current version August 13, 2013. This work was supported in part by Program 973 under Grant 2010CB327902 and the National Natural Science Foundation of China under Grant 61231018. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Stefano Tubaro.

The authors are with the Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an 710049, China (e-mail: jianjiwang@foxmail.com; nnzheng@mail.xjtu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2013.2268977

is a lot of work to do for speeding up encoding in FIC. Moreover, some schemes, such as the variance-based block-sorting scheme proposed by He *et al* in 2004 [20] and the Fisher's 72 classes scheme [15], are more efficient in encoding, but the reconstructed image quality in these schemes cannot be well preserved. Hence, a fast FIC scheme is required to speed up encoding more efficiently and, to preserve the reconstructed image quality better.

In this paper, a novel fractal compression scheme is proposed to meet both the efficiency and the reconstructed image quality requirements. This scheme is based on the fact that the affine similarity between two image blocks is equivalent to the absolute value of Pearson's correlation coefficient (APCC) between them [22], [23]. Firstly, all the domain blocks are classified into 3 classes according to the classification method proposed by Fisher in 1992 [15]. We will prove the Fisher's 3 classes method is an APCC-based method in Section III in this paper. Secondly, the domain blocks are sorted in each class by APCCs between these domain blocks and a preset block. Then the matching domain block for a range block can be searched from the APCC interval in which these domain blocks have the closer APCCs with APCC the range block has (all the APCCs are computed with the preset block). Since both the steps in our scheme are based on APCC which is equivalent to the affine similarity in FIC, the reconstructed image quality is well preserved. Moreover, the encoding time is significantly reduced in our APCC-based FIC scheme.

The rest of this paper is structured as follows. The concept of fractal image compression and the relationship between affine similarity and APCC are briefly introduced in Section II. In Section III, the strategy of choosing and classifying the domain blocks is introduced. In Section IV, we focus on the APCC-based domain block sorting algorithm and the method of searching the block pairs. The preset blocks are trained offline in Section V. Several optimization methods involved in the proposed scheme are introduced in Section VI. The experiment results are shown in Section VII. Finally, the conclusions are drawn in Section VIII.

II. BFIC and Affine Similarity

Jacquin's FIC scheme is regarded as the baseline fractal image compression (BFIC), in which the self-similarity of an image is measured by the affine similarity in the partitioned iterated function system (PIFS).

A. The Baseline FIC

As introduced in Section I, an image is firstly partitioned into two pools according to two different sizes $n \times n$ and $2n \times 2n$ in BFIC, one is the range pool filled with the non-overlapping blocks with size $n \times n$, and the other is the domain pool. All domain cells are then contracted into size $n \times n$ by averaging four pixels to one pixel. Then the range cells are the blocks to be encoded, and FIC takes the domain pool as a virtual codebook. The domain pool is called the "virtual codebook" because it is only used during encoding but not during the decoding process.

To improve the reconstructed image quality, eight transformations are applied to all domain blocks to octuple the domain block number. These transformations are T_0, T_1, T_2, T_3, T_4, T_5, T_6, and T_7, respectively, as introduced below [24]:

T_0: Identity.
T_1: Orthogonal reflection about mid-vertical axis.
T_2: Orthogonal reflection about mid-horizontal axis.
T_3: Orthogonal reflection about first diagonal.
T_4: Orthogonal reflection about second diagonal.
T_5: Rotation around center of block, through $+90^0$.
T_6: Rotation around center of block, through $+180^0$.
T_7: Rotation around center of block, through $+270^0$.

Subsequently, for an arbitrary range block $\mathbf{R}$, a domain block $\mathbf{D}$ has to be searched so that one affine transformation $s\mathbf{D}+o\mathbf{1}$ exists to minimize the squared L^2 distance with $\mathbf{R}$, where s and o are the affine scalar parameters and $\mathbf{1}$ is a block with size $n \times n$ in which all pixels equal to 1, then

$$\mathrm{MSE}(\mathbf{R}, s\mathbf{D} + o\mathbf{1}) = \|\mathbf{R} - (s\mathbf{D} + o\mathbf{1})\|_2 \tag{1}$$

where $\|\cdot\|_2$ is the two-norm. Finally, the combination of (s, o, index of $\mathbf{D}$ in the domain pool) constructs the PIFS subsystem of $\mathbf{R}$ in BFIC, and all the subsystems of range cells group into the PIFS of the original image.

Minimizing MSE in Eq. (1) by the least-squares method, we can obtain the values of parameters s, o, and the simplified energy function H as follows [23]:

$$\begin{aligned} \max_{\mathbf{D}} \mathrm{H} &= \frac{\sigma_{\mathbf{RD}}^2}{\sigma_{\mathbf{D}}^2} \\ s &= \frac{\sigma_{\mathbf{RD}}}{\sigma_{\mathbf{D}}^2} \\ o &= \mu_{\mathbf{R}} - s\mu_{\mathbf{D}} \end{aligned} \tag{2}$$

where $\mu_{\mathbf{R}}$ and $\mu_{\mathbf{D}}$ are the mean intensity of the blocks $\mathbf{R}$ and $\mathbf{D}$, respectively, $\sigma_{\mathbf{D}}$ is the standard deviation of the block $\mathbf{D}$, and $\sigma_{\mathbf{RD}}$ is the covariance between the blocks $\mathbf{R}$ and $\mathbf{D}$. For the range block $\mathbf{R}$, maximizing H in the domain pool is equivalent to minimizing MSE in Eq. (1). The three sub-equations in Eq. (2) construct the expressions of the BFIC.

The reconstruction of the image is a process of iterative computation. It achieves the attractor of the PIFS, which is a fractal representation of the original image.

B. The Affine Similarity in BFIC

In Eq. (2), a hypothesis of $\sigma_{\mathbf{D}} \neq 0$ is made. In fact, if $\sigma_{\mathbf{D}} = 0$, the block $\mathbf{D}$ is directly proportional to the block $\mathbf{1}$. Then the scalar multiplier s in $s\mathbf{D} + o\mathbf{1}$ can be added to the offset o and set $s = 0$. Similarly, if $\sigma_{\mathbf{R}} = 0$, the block $\mathbf{R}$ is directly proportional to the block $\mathbf{1}$, and then the block $\mathbf{R}$ can be losslessly expressed by the block $\mathbf{1}$. Hence, the parameter s in $s\mathbf{D} + o\mathbf{1}$ can also be set to 0.

If $\sigma_{\mathbf{R}} \neq 0$ and $\sigma_{\mathbf{D}} \neq 0$, it has been proved in some papers [22], [23] that the affine similarity between two image blocks in FIC is equivalent to the absolute value of the Pearson's correlation coefficient (APCC) between them whether the image measurement is MSE or SSIM [25], [26]. In fact, as in Eq. (1), MSE between a range block $\mathbf{R}$ and a linear expression

of the domain block $\mathbf{D}$ can be computed as following [22]:

$$\begin{aligned} &\mathrm{MSE}^2(\mathbf{R}, s\mathbf{D} + o\mathbf{1}) = \|\mathbf{R} - (s\mathbf{D} + o\mathbf{1})\|_2^2 \\ &= m^2(\sigma_{\mathbf{R}}^2 - \tfrac{\sigma_{\mathbf{RD}}^2}{\sigma_{\mathbf{D}}^2}) = m^2\sigma_{\mathbf{R}}^2(1 - \rho_{\mathbf{RD}}^2) \end{aligned} \tag{3}$$

where $\sigma_{\mathbf{R}}$ is the standard deviation of the block $\mathbf{R}$, $\rho_{\mathbf{RD}} = \frac{\sigma_{\mathbf{RD}}}{\sigma_{\mathbf{R}}\sigma_{\mathbf{D}}}$ is the Pearson's correlation coefficient between $\mathbf{R}$ and $\mathbf{D}$, and $m = n \times n$ is the number of pixels in the block. In the process of searching a matching domain block for the range block $\mathbf{R}$, the values of m and $\sigma_{\mathbf{R}}^2$ are invariant. Hence, the most affine similar block for $\mathbf{R}$ is the block having the largest absolute value of Pearson's correlation coefficient with $\mathbf{R}$.

III. APCC-Based Block Classification

The absolute value of the Pearson's correlation coefficient (APCC) should be an efficient feature to speed up encoding because the affine similarity between two blocks in FIC is equivalent to APCC between them.

If the block numbers in the range pool and domain pool are $\mathbf{R}_{no}$ and $\mathbf{D}_{no}$, respectively, the number of pairwise comparisons is $\mathbf{R}_{no} \times \mathbf{D}_{no}$ in the exhausting search in BFIC.

In BFIC, the original domain blocks with size $2n \times 2n$ are firstly contracted as the same size as the range blocks. Then eight transformations are applied to them to octuple the block number in the domain pool. According to the above analysis, octupling the block number in the domain pool leads to octuple the computational complexity in searching the matching pairs. Hence, there is great potential to speed up encoding by reducing the scale of the domain pool.

In 1992, Fisher proposed a FIC classification scheme to speed up encoding, in which a method existed to cut short the domain pool with classifying the domain blocks into 3 classes [15]. We call it as the Fisher's 3 classes method in this paper. In Fisher's scheme, every range or domain block is firstly divided into four sub-blocks according to the mid-horizontal axis and the mid-vertical axis, then all the range and domain blocks are classified into 3 classes according to the means of luminance in the four sub-blocks, or 24 classes according to the order of the variances in four sub-blocks, or 72 classes by combining the 3 classes method and the 24 classes methods. In our scheme, only the Fisher's 3 classes method is used.

In Fisher's 3 classes method, for an arbitrary block in the range or domain pool, suppose the sums of luminance in its four sub-blocks are a_1, a_2, a_3, and a_4, from the upper left to the lower right, respectively. We have known that every block $\mathbf{D}$ in the domain pool is transformed to eight blocks, $\mathbf{D}$, $\mathbf{D}_1$, $\mathbf{D}_2$, $\mathbf{D}_3$, $\mathbf{D}_4$, $\mathbf{D}_5$, $\mathbf{D}_6$, and $\mathbf{D}_7$ by eight transformations T_0, T_1, T_2, T_3, T_4, T_5, T_6, and T_7 in BFIC. If a_1, a_2, a_3, and a_4 vary each other in $\mathbf{D}$, there is one and only one block derived form $\mathbf{D}$ meeting one of the three conditions in Eq. (4):

$$\begin{aligned} &a_1 \geq a_2 \geq a_3 \geq a_4 \\ &a_1 \geq a_2 \geq a_4 \geq a_3 \\ &a_1 \geq a_4 \geq a_2 \geq a_3. \end{aligned} \tag{4}$$

Then block $\mathbf{D}$ is classified into one of the three classes in accordance with the condition which one of its generated block meets in Eq. (4), and the corresponding transformation $T_{\mathbf{D}}$ which transforms $\mathbf{D}$ into this class can also be obtained. If some values are kept the same in a_1, a_2, a_3, and a_4, two or more of the generated blocks from $\mathbf{D}$ may meet two or all of the conditions in Eq. (4). In this case, the block can be classified into two or all classes.

Then we have the Lemma 1 as following:

Lemma 1: The Fisher's 3 classes method is an APCC-based block classification method.

Proof: For a block $\mathbf{D}$ in one of the three classes expressed in Eq. (4), now we prove that an APCC-based block classification method exists to classify $\mathbf{D}$ into the same class.

Construct three matrixes C_1, C_2, and C_3 as following:

$$C_1 = \begin{bmatrix} \mathbf{2} & \mathbf{1} \\ \mathbf{0} & \mathbf{-3} \end{bmatrix}, C_2 = \begin{bmatrix} \mathbf{2} & \mathbf{1} \\ \mathbf{-3} & \mathbf{0} \end{bmatrix}, C_3 = \begin{bmatrix} \mathbf{2} & \mathbf{0} \\ \mathbf{-3} & \mathbf{1} \end{bmatrix}$$

Each number in the matrixes C_1, C_2, and C_3 denotes a square sub-matrix filling with the number. Suppose the standard deviation is $\sigma_{\mathbf{C}}$ for C_1, C_2, and C_3, and the luminance sums for the four sub-blocks in the block $\mathbf{D}$ are a_1, a_2, a_3, and a_4, respectively, and the standard deviation of $\mathbf{D}$ is $\sigma_{\mathbf{D}}$.

Next we classify block $\mathbf{D}$ by the APCC-based block classification method in which $\mathbf{D}$ is classified into the class related to the matrix in C_1, C_2, and C_3 which has the maximum APCC with $\mathbf{D}$. Suppose the APCCs between $\mathbf{D}$ and three matrixes are ACC_1, ACC_2, and ACC_3, respectively, then we have

$$\begin{aligned} ACC_1 &= |2a_1 + a_2 - 3a_4|/(\sigma_{\mathbf{C}}\sigma_{\mathbf{D}}) \\ ACC_2 &= |2a_1 + a_2 - 3a_3|/(\sigma_{\mathbf{C}}\sigma_{\mathbf{D}}) \\ ACC_3 &= |2a_1 + a_4 - 3a_3|/(\sigma_{\mathbf{C}}\sigma_{\mathbf{D}}) \end{aligned}. \tag{5}$$

For $a_1 \geq a_2 \geq a_3 \geq a_4$, we have

$$ACC_1 - ACC_2 = 3(a_3 - a_4)/(\sigma_{\mathbf{C}}\sigma_{\mathbf{D}}) \geq 0. \tag{6}$$

If $2a_1 + a_4 - 3a_3 \geq 0$,

$$ACC_1 - ACC_3 = (a_2 + 3a_3 - 4a_4)/(\sigma_{\mathbf{C}}\sigma_{\mathbf{D}}) \geq 0. \tag{7}$$

If $2a_1 + a_4 - 3a_3 \leq 0$,

$$ACC_1 - ACC_3 = (4a_1 + a_2 - 3a_3 - 2a_4)/(\sigma_{\mathbf{C}}\sigma_{\mathbf{D}}) \geq 0. \tag{8}$$

Hence, for $a_1 \geq a_2 \geq a_3 \geq a_4$, whether $2a_1 + a_4 - 3a_3$ is greater than zero or not, we have

$$\begin{aligned} ACC_1 &\geq ACC_2 \\ ACC_1 &\geq ACC_3 \end{aligned}. \tag{9}$$

Similarly, for $a_1 \geq a_2 \geq a_4 \geq a_3$, whether $2a_1 + a_4 - 3a_3$ is greater than zero or not, we have

$$\begin{aligned} ACC_2 &\geq ACC_1 \\ ACC_2 &\geq ACC_3 \end{aligned}. \tag{10}$$

And, for $a_1 \geq a_4 \geq a_2 \geq a_3$, whether $2a_1 + a_2 - 3a_4$ is greater than zero or not, we have

$$\begin{aligned} ACC_3 &\geq ACC_1 \\ ACC_3 &\geq ACC_2 \end{aligned}. \tag{11}$$

Synthetic analysis bout the inequations (9), (10), and (11) with considering the conditions that the equality holds, the class of block $\mathbf{D}$ classified by the APCC-based block classification method related to C_1, C_2, and C_3 is kept the same as the class classified by the Fisher's 3 classes method.

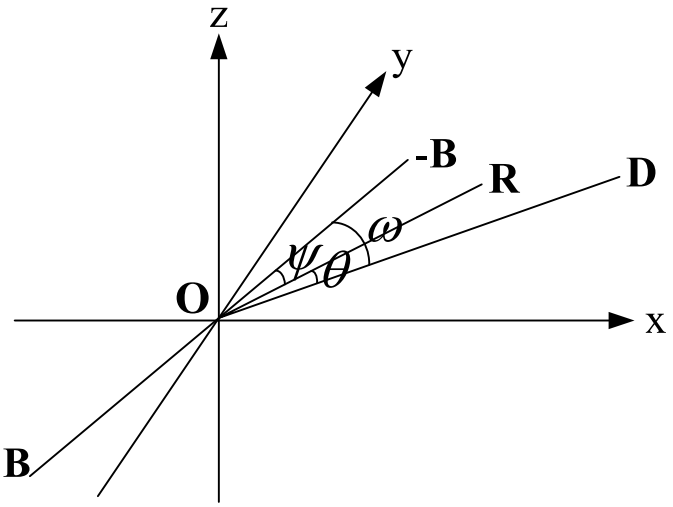


Fig. 1. The angles θ, ψ, and ϖ corresponding to Eq. (13).

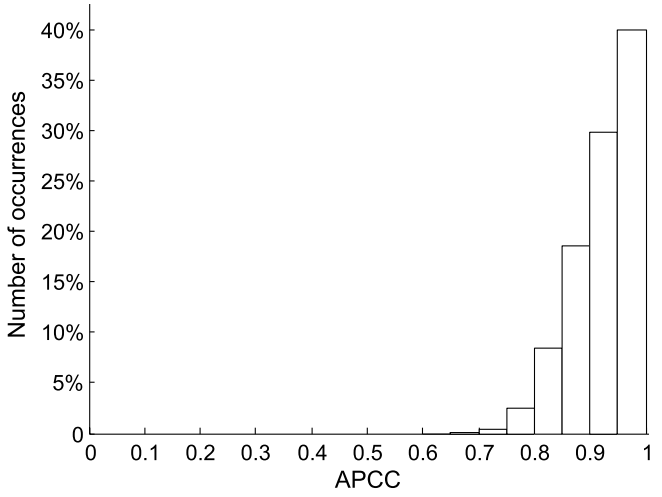


Fig. 2. The histogram of the APCCs in PIFSs of 10 test images with range block size 4×4. The 10 test images are baboon, boat, couple, F16, goldhill, Lena, man, milk drop, pepper, and sailboat, respectively.

Because Fisher's 24 classes method and 72 classes method are not APCC-based classification method, only the Fisher's 3 classes method is used in our scheme.

In this paper, the domain pool without enlarging with the eight transformations T_0, T_1, T_2, T_3, T_4, T_5, T_6, and T_7 is called the crude domain pool, and the domain pool enlarged with these transformations is called the BFIC domain pool. In addition, we make the codes generated in the encoder of the proposed scheme can be directly used in BFIC decoder.

In our scheme, for simplicity, every block is classified into only one class according to priority order of class 1, class 2, and class 3 expressed in Eq. (4) even if some sub-blocks of the block have the same sum of luminance. Then the index of the block **D** in the crude domain pool, the transformation on **D** to the class it belongs to, and three domain classes filled with the transformed domain blocks are retained to encode an image. Other conditions, such as the redundant generated blocks from the block **D**, are useless. Each range block is also classified with the same method, and its matching domain block is searched in the same class.

By this step, the total domain block number in the three domain classes is only one eighth of the domain block number in BFIC, which can greatly reduce the pairwise comparisons between the range blocks and domain blocks. Moreover, all the domain and range blocks are classified into 3 classes to reduce the pairwise comparisons further. More importantly, the reconstructed image quality is well preserved because the Fisher's 3 classes method is APCC-based.

IV. Sorting Domain Blocks With APCC

To further reduce the computational cost in FIC, a new method to sort the domain blocks with respect to APCC is discussed below. In this paper, $\mathbf{I^{n\times n}}$ is the symbol of $n \times n$ dimension image space. For example, for the grayscale images, $\mathbf{I^{n\times n}}$ is the discrete $n \times n$ dimension space with the integer pixels lying in [0, 255]. Then a lemma is given as following:

Lemma 2: Suppose $\mathbf{R}, \mathbf{D} \in \mathbf{I^{n\times n}}$. For an arbitrary non-zero variance vector $\mathbf{B} \in \mathbf{I^{n\times n}}$, if $|\rho(\mathbf{R}, \mathbf{D})| \to 1$, then $|\rho(\mathbf{R}, \mathbf{B})| \to |\rho(\mathbf{D}, \mathbf{B})|$. $\rho(\cdot, \cdot)$ is the Pearson's correlation coefficient.

Proof: If APCC of a pair of **R**, **D**, and **B** equals to 1, the lemma is obviously correct. If not, take **R**, **D**, and **B** as points in m-dimension space ($m = n \times n$), and suppose the vectors connecting the points **R**, **D**, and **B** with the origin are $\vec{r}$, $\vec{d}$ and $\vec{b}$, respectively. We have

$$|\rho(\mathbf{R}, \mathbf{D})| = |\rho(-\mathbf{R}, \mathbf{D})| = |\rho(\mathbf{R}, -\mathbf{D})|, \tag{12}$$

so $|\rho(\cdot, \cdot)|$ is an even function for its variables. Because $\rho(\mathbf{R}, \mathbf{D})$ equals to the cosine of the angle between $\vec{r}$ and $\vec{d}$, $|\rho(\mathbf{R}, \mathbf{D})|$ then equals to the cosine of the minimum angle θ in both the angles: one is the angle between $\vec{r}$ and $\vec{d}$, and the other is between $\vec{r}$ and $-\vec{d}$. And let ψ be the minimum angle in both the angles: one is the angle between $\vec{r}$ and $\vec{b}$, and the other is between $\vec{r}$ and $-\vec{b}$, and let ω be the minimum angle in both the angles: one is the angle between $\vec{d}$ and $\vec{b}$, and the other is between $\vec{d}$ and $-\vec{b}$, as shown as in Fig. 1. Then we have $\theta, \psi, \omega \in (0°, 90°]$ and

$$\begin{aligned} \cos\theta &= |\rho(\mathbf{R}, \mathbf{D})| \\ \cos\psi &= |\rho(\mathbf{R}, \mathbf{B})| \\ \cos\omega &= |\rho(\mathbf{D}, \mathbf{B})|. \end{aligned} \tag{13}$$

According to the relationship between θ, ψ and ω, we have

$$|\theta - \psi| \le \omega \le \min(90°, \theta + \psi). \tag{14}$$

For cosine is a monotonically decreasing function in $[0°, 90°]$,

$$\max(0, \cos(\theta + \psi)) \le \cos\omega \le \cos(\theta - \psi). \tag{15}$$

When $|\rho(\mathbf{R}, \mathbf{D})| \to 1$, we have $\cos\theta \to 1$, $\sin\theta \to 0$, then

$$\begin{aligned} \lim_{\cos\theta\to 1} \cos(\theta + \psi) &= \lim_{\cos\theta\to 1} (\cos\theta\cos\psi - \sin\theta\sin\psi) = \cos\psi \\ \lim_{\cos\theta\to 1} \cos(\theta - \psi) &= \lim_{\cos\theta\to 1} (\cos\theta\cos\psi + \sin\theta\sin\psi) = \cos\psi. \end{aligned} \tag{16}$$

Hence,

$$\lim_{|\rho(\mathbf{R},\mathbf{D})|\to 1} |\rho(\mathbf{R}, \mathbf{B})| = \lim_{|\rho(\mathbf{R},\mathbf{D})|\to 1} |\rho(\mathbf{D}, \mathbf{B})|. \tag{17}$$

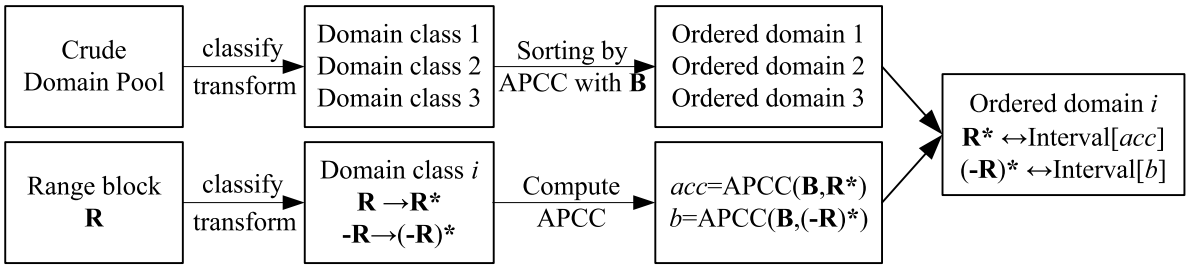


Fig. 3. The flowchart of the proposed FIC scheme with block classification and sorting based on APCC.

According to Lemma 2, if the domain block $\mathbf{D}$ is a high-quality matching block for the range block $\mathbf{R}$, for an arbitrary block $\mathbf{B}$ with non-zero variance, APCC between $\mathbf{R}$ and $\mathbf{B}$ is close to APCC between $\mathbf{D}$ and $\mathbf{B}$. Moreover, it can be seen from Fig. 2 that the APCCs of about 70% of the matching blocks are more than 0.9 in the PIFSs of 10 test images with range size 4×4, and the APCCs of about 88.7% of the matching blocks are more than 0.85, and the APCCs of only 2.9% of the matching blocks are less than 0.8. There is no occurrence with APCC less than 0.66 in Fig. 2. Based on Lemma 2 and Fig. 2, a new APCC-based domain block sorting method is proposed: a preset block $\mathbf{B}$ is used to sort the domain blocks by APCCs computed between each domain block and the preset block $\mathbf{B}$. Then for a range block $\mathbf{R}$, its corresponding domain block can be searched in a set of domain blocks in which the APCCs of these domain blocks are close to APCC between $\mathbf{R}$ and $\mathbf{B}$.

Since the domain pool has been simplified and classified into 3 classes by Fisher's 3 classes method, now we need only sort each domain class independently. Then for a range block $\mathbf{R}$, the class to which $\mathbf{R}$ belongs should be computed. Because APCC is an absolute value, both the blocks $\mathbf{R}$ and $-\mathbf{R}$ need to be taken into consideration. Suppose the block $\mathbf{R}$ is transformed into $\mathbf{R}^*$ to match one class defined in Eq. (4), and block $-\mathbf{R}$ is transformed into $(-\mathbf{R})^*$ to match the same class. In fact, $\mathbf{R}^*$ and $(-\mathbf{R})^*$ always match the same class, but the transformation from $\mathbf{R}$ to $\mathbf{R}^*$ and the transformation from $-\mathbf{R}$ to $(-\mathbf{R})^*$ are generally different.

Take $\mathbf{R}^*$ as an example, suppose acc is APCC between $\mathbf{R}^*$ and the preset block $\mathbf{B}$. Then a set of domain blocks is chosen for $\mathbf{R}^*$ to search its matching domain block. The set of domain blocks is chosen in the ordered domain class to make APCCs between the preset block $\mathbf{B}$ and these blocks in this set all around acc. That is to say, these blocks in this set are selected from an APCC interval centered by acc in this ordered domain class. Then the pairwise comparisons are performed between $\mathbf{R}^*$ and all the blocks in the set to search the corresponding domain block for $\mathbf{R}^*$.

If the matching block for $\mathbf{R}^*$ is $\mathbf{D}^*$ in this set and $T_{\mathbf{R}}(\mathbf{R}) = \mathbf{R}^*$, $T_{\mathbf{D}}(\mathbf{D}) = \mathbf{D}^*$ in which $\mathbf{D}$ is one block in the crude domain pool and $\mathbf{D}^*$ is derived from $\mathbf{D}$, then $(T_{\mathbf{R}})^{-1}T_{\mathbf{D}}(\mathbf{D})$ is the matching domain block for the block $\mathbf{R}$. Because the eight transformations are closed for the inverse operations and compound operations, $T_{\mathbf{R}}$, $T_{\mathbf{D}}$, $(T_{\mathbf{R}})^{-1}T_{\mathbf{D}} \in \{T_0, T_1, T_2, T_3, T_4, T_5, T_6, T_7\}$. If the index of the block $\mathbf{D}$ in the crude domain pool is t, the index of $(T_{\mathbf{R}})^{-1}T_{\mathbf{D}}(\mathbf{D})$ in the BFIC domain pool can be calculated. Suppose it is $index(t, T_{\mathbf{D}}(T_{\mathbf{R}})^{-1})$. Then the parameters s and o computed in Eq. (2), and $index(t, T_{\mathbf{D}}(T_{\mathbf{R}})^{-1})$ compose the PIFS subsystem for the block $\mathbf{R}$. That is, $(s, o, index(t, T_{\mathbf{D}}(T_{\mathbf{R}})^{-1}))$.

Because the PIFS subsystem in the proposed scheme has the same form with the PIFS subsystem of BFIC introduced in Subsection A in Section II, the proposed FIC scheme does not change the bit rates and compression ratios involved in the compression of each image compared with BFIC.

In the above analysis we take $\mathbf{R}^*$ as an example. Certainly, $(-\mathbf{R})^*$ also needs to be considered because APCC is an absolute value. The final matching block for $\mathbf{R}$ should be chosen from both the matching blocks for $\mathbf{R}^*$ and $(-\mathbf{R})^*$. The flowchart of this scheme is shown in Fig. 3.

For the block $\mathbf{R}^*$, a set of domain blocks is required to search its matching block in the corresponding domain class. In this paper, a simple strategy is used to obtain the set. For each block $\mathbf{R}^*$, we firstly find the block having the closest APCC with the APCC $\mathbf{R}^*$ has in the ordered domain class (all APCCs are computed with the preset block $\mathbf{B}$), then k blocks near the block are selected for the final comparison.

V. Training the Preset Block

In the APCC-based block sorting method proposed in this paper in Section IV, a preset block $\mathbf{B}$ is required to compute APCCs with other blocks. In the proposed scheme, the preset blocks are trained offline, and then they are directly applied in the encoding process as auxiliary blocks.

From Lemma 2 in Section IV, each block in $\mathbf{I}^{\mathbf{n}\times\mathbf{n}}$ with non-zero variance can be chosen as $\mathbf{B}$ in theory. However, experiment results indicate that it has a great impact on the visual quality of the reconstructed image to choose different blocks as the preset block $\mathbf{B}$. Because the block $\mathbf{D}$ satisfying $|\rho(\mathbf{R}, \mathbf{D})| \to 1$ is usually hard to search for $\mathbf{R}$, it is important to choose a proper block as the preset block $\mathbf{B}$ to search the best approximate $\mathbf{D}$. Moreover, the domain pool is classified into 3 classes and the search of the matching pairs in different classes is absolutely independent. Hence, we can choose different blocks as the preset blocks for different classes.

For a preset block $\mathbf{B}$, to search the matching block for a range block $\mathbf{R}$, we firstly find these blocks $\mathbf{D}$ meeting the Eq. (18) in this domain class:

$$|\rho(\mathbf{R}, \mathbf{B})| \approx |\rho(\mathbf{D}, \mathbf{B})| \tag{18}$$

$\mathbf{B}$ and $\mathbf{R}$ are known in the encoding process, then the set of $\mathbf{D}$ meeting Eq. (18) forms a structure C of $\mathbf{R}$ and $\mathbf{B}$ which is

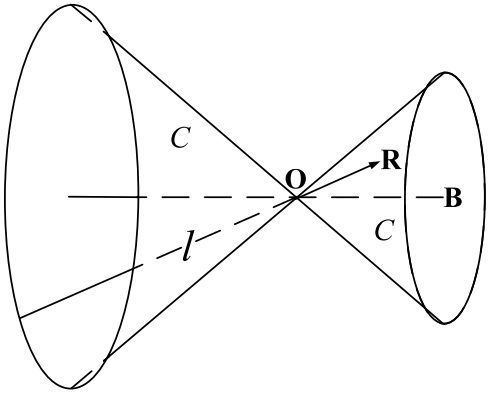


Fig. 4. The structure C constructed by blocks **D** which satisfy Eq. (18) and the structure l constructed by blocks **D** which satisfy Eq. (19).

a neighborhood of the surface of a double cone whose cone-axis is the line through the origin and the preset block **B**, and cone-angle is twice the arc cosine of APCC between the blocks **R** and **B**. However, the blocks **D** satisfying Eq. (19) only form a structure l of **R** and **B** which is a neighborhood of one element of the double cone through the origin and **R**. Fig. 4 shows the structures C and l constructed by Eqs (18) and (19), respectively.

$$|\rho(\mathbf{R}, \mathbf{D})| \to 1. \tag{19}$$

In Section IV, a set containing k domain blocks is selected to search the final matching block for block $\mathbf{R}^*$. The blocks in this set are likely to belong to the structure C of $\mathbf{R}^*$ and **B** because they are selected with APCCs close to APCC between $\mathbf{R}^*$ and **B**. Generally, because the set is selected with a few blocks, the precondition of one block to enter the selected small set is that this block belongs to C. However, the blocks in l are just what we need, and l is merely a part of C.

If the number of the domain blocks in l is invariable, the opportunity of these blocks in l to enter the selected domain set may reduce when the block number in C increases. Hence, it may need to enlarge the set to search the matching block in l, which brings high computational complexity. If the set is not enlarged, the searched matching block may not meet the needs. Since there are usually lots of blocks beside the center block or barycenter block of the domain class, which can lead to a large structure C if we use them as **B**, both the center or barycenter are not the appropriate blocks to be **B**, and **B** should even be far away from them.

From another view, for its difficulty to search the exact **D** to meet Eq. (19), the block **B** should be chosen to easy to find a not accuracy but acceptable **D** from C. In this case, $|\rho(\mathbf{R}, \mathbf{D})| < 1$ and there is a non-zero angle θ between the line through the origin and **R** and the line through the origin and **D**. If block **B** lies on the super-plane which passes through the origin to separate the blocks **R** and **D** on its two sides and has the same angle $\theta/2$ with the line through the origin and **R** and the line through the origin and **D**, the block **D** can be easily searched. Hence, statistically, block B should be chosen as the barycenter of the domain blocks from this view.

The above discussion draws two opposite conclusions. One asks for the preset block to get away from the barycenter of the domain class, and the other asks for taking the barycenter as the preset block. Finally, we have no any choices but to obtain the preset block **B** by training.

TABLE I
PROCEDURE OF TRAINING THE PRESET BLOCK

Step 1: Transform and classify all the blocks in S into 3 classes, S_1, S_2 and S_3, according to Eq. (4).

Step 2:
for every block s_i in S_1
  $\rho_i = 0$;
  for every block s_j in S_1
    $\rho_{ij} = |\rho(s_i, s_j)|$;
  endfor
  Sort S_1 with ρ_{ij} to an ordered sample set S_1^*;
  for every block s_j^* in S_1^*
    $\rho_{ij}^* = \left|\rho(s_i, s_j^*)\right|$;
    $Interval$ = {q+1 nearest blocks with ρ_{ij}^* in S_1^*}$\backslash s_j^*$;
    for every block in_l in $Interval$
      $\rho_{Interval,i}^l = |\rho(s_i, in_l)|$;
    endfor
    $\rho_i = \rho_i + \max_l \rho_{Interval,i}^l$;
  endfor
endfor
$t = \arg\max_i \rho_i$;
$\mathbf{B}_1 = s_t$;

The target of training **B** is obvious: Find a block **B** in a sample set S to maximize the sum of APCCs between each original block and its matching block searched in an interval generated by APCCs with **B** except the original block itself. Because the three classes in Eq. (4) are absolutely independent in the search process, their preset blocks can also be trained independently. Table I shows the procedure of training the preset block $\mathbf{B}_1$ for the first class.

All the data in Table I are defined except for q. In our experiment, although it can generate different preset blocks with different q, these blocks have almost the same efficiency for encoding. The best preset block trained with $q = 200$ is also one of the best preset blocks with $q = 300$.

We use a sample set containing 36 633 sample blocks with size 4×4 and a sample set containing 77 805 sample blocks with size 8×8 to train the preset blocks. The preset blocks trained as in Table I for three classes expressed in Eq. (4) with different sizes 4×4 and 8×8 are shown in Fig. 5, respectively.

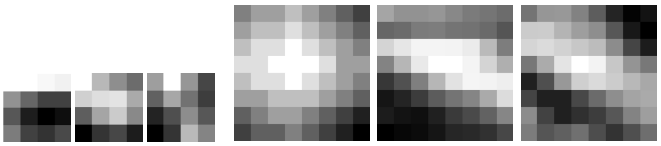
VI. ALGORITHM OPTIMIZATION

In Sections III and IV, the novel APCC-based FIC scheme is proposed by two steps. In this section, we introduce several optimization methods in the proposed scheme.

A. Optimization of the Energy Function

For a range block **R**, APCC between **R** and a domain block **D** can be computed as Eq. (20):

$$\left|\rho_{\mathbf{RD}}\right| = \left|\frac{\sigma_{\mathbf{RD}}}{\sigma_{\mathbf{R}}\sigma_{\mathbf{D}}}\right| \tag{20}$$



(a) 4×4 preset blocks (b) 8×8 preset blocks

Fig. 5. The preset blocks trained for the first class, second class, and third class from left to right, corresponding to the classes expressed in Eq. (4). (a) Three preset blocks trained for 4 × 4 image blocks. (b) Three preset blocks trained for 8 × 8 image blocks.

To search the corresponding domain block for block $\mathbf{R}$, these APCCs between $\mathbf{R}$ and some domain blocks need to be computed. In the process, $\sigma_{\mathbf{R}}$ is kept invariant. Hence, the energy function as Eq. (20) can be written as Eq. (21) to reduce the computational cost:

$$H(\mathbf{R},\mathbf{D}) = \frac{\sigma_{\mathbf{RD}}^2}{\sigma_{\mathbf{D}}^2} \tag{21}$$

If the domain block $\mathbf{D}$ is standardized to make its standard deviation equal to 1, then the maximization of APCC between $\mathbf{R}$ and $\mathbf{D}$ is just equivalent to maximizing the absolute value of the inner product between them. Here it is unnecessary to standardize the range block $\mathbf{R}$.

B. Optimization of Block Classification

In the Fisher's 3 classes method, each range or domain block needs to be classified into one of the three classes expressed in Eq. (4). For one given block, such as the range block $\mathbf{R}$, its corresponding class, and the transformation $T_{\mathbf{R}}$ which transforms $\mathbf{R}$ to $\mathbf{R}^*$, and the transformation $T_{-\mathbf{R}}$ which transforms $-\mathbf{R}$ to $(-\mathbf{R})^*$ need to be considered, in which $\mathbf{R}^*$ and $(-\mathbf{R})^*$ are kept the same meanings as in Section IV.

To find the class to which $\mathbf{R}$ belongs, and the transformations $T_{\mathbf{R}}$ and $T_{-\mathbf{R}}$, the general method is to test the eight transformations introduced in Subsection A in Section II on the block $\mathbf{R}$ one by one, if one transformation transforms $\mathbf{R}$ into one class expressed in Eq. (4), then this class and the transformation are just what we need. This general method is easy to implement but time-consuming to compute.

According to Eq. (4), the class $\mathbf{R}$ belonged to, the transformation $T_{\mathbf{R}}$, and the transformation $T_{-\mathbf{R}}$ are all determined by the luminance sums of the four sub-blocks of $\mathbf{R}$. In the proposed scheme, we compute them offline and then directly apply these results to classify blocks and find their corresponding transformations. As mentioned in Section IV, we find that $\mathbf{R}^*$ and $(-\mathbf{R})^*$ always match the same class. If the luminance sums of the four sub-blocks of $\mathbf{R}$ vary each other, $T_{\mathbf{R}}$ and $T_{-\mathbf{R}}$ are always different.

Suppose the luminance sums of the four sub-blocks of $\mathbf{R}$ are r_1, r_2, r_3, and r_4, respectively. Some examples about the class, the transformation $T_{\mathbf{R}}$, and the transformation $T_{-\mathbf{R}}$ related to different orders of r_1, r_2, r_3, and r_4 are listed in Table II, in which the class 1, class 2, and class 3 are defined according to Eq. (4), respectively. There are 24 cases in total according to the order of r_1, r_2, r_3, and r_4.

C. The Compound Transformations

In Section IV, if the matching block for $\mathbf{R}^*$ is $\mathbf{D}^*$ and $T_{\mathbf{R}}(\mathbf{R}) = \mathbf{R}^*$, $T_{\mathbf{D}}(\mathbf{D}) = \mathbf{D}^*$, then $(T_{\mathbf{R}})^{-1}T_{\mathbf{D}}(\mathbf{D})$ is the matching domain block for the block $\mathbf{R}$. The compound transformation $(T_{\mathbf{R}})^{-1}T_{\mathbf{D}} \in \{T_0, T_1, T_2, T_3, T_4, T_5, T_6, T_7\}$ can also be computed offline. Then the results are directly applied in the encoding process. There are 64 cases in total for $(T_{\mathbf{R}})^{-1}T_{\mathbf{D}}$, with 8 cases in $T_{\mathbf{R}}$ and 8 cases in $T_{\mathbf{D}}$.

TABLE II

SOME EXAMPLES ABOUT THE CLASS, THE TRANSFORMATIONS $T_{\mathbf{R}}$ AND $T_{-\mathbf{R}}$ RELATED TO DIFFERENT ORDERS OF r_1, r_2, r_3, AND r_4 IN BLOCK $\mathbf{R}$

order	class	$T_{\mathbf{R}}$	$T_{-\mathbf{R}}$
$r_1 \geqslant r_2 \geqslant r_3 \geqslant r_4$	class 1	T_0	T_6
$r_1 \geqslant r_4 \geqslant r_3 \geqslant r_2$	class 3	T_3	T_7
$r_2 \geqslant r_3 \geqslant r_1 \geqslant r_4$	class 3	T_1	T_6
$r_3 \geqslant r_4 \geqslant r_2 \geqslant r_1$	class 2	T_2	T_0
$r_4 \geqslant r_2 \geqslant r_1 \geqslant r_3$	class 2	T_4	T_5

VII. EXPERIMENT RESULTS AND ANALYSIS

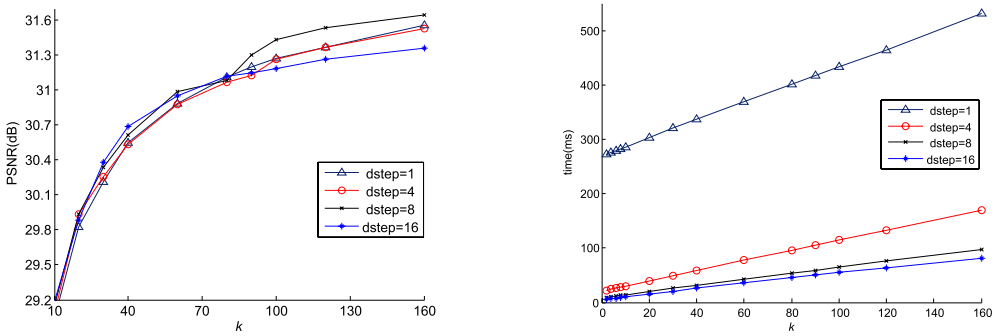
To verify the performance of the proposed APCC-based block classification and sorting FIC scheme, the experiments are performed on 16 standard test images with size 512 × 512, 2 standard test images with size 1024 × 1024, and a standard test image with size 256 × 256. The peak signal-to-noise ratio (PSNR) is used to measure the reconstructed image quality for different schemes with different partitions. All the demos in this paper are implemented using C++ language, and the experiments are run on the computer environment: CPU I3–2100 and windows XP OS.

A. Experiment on the Preset Blocks

In Section V, the training method for the preset block is introduced, and the preset blocks for three classes with different sizes are shown in Fig. 5. In our experiment, compared with the center blocks which are the average blocks in each domain class, and the barycenter blocks which are the blocks with the maximum first-moment of APCC in each domain class, and the farthest blocks from the barycenter blocks which have the minimum first-moment of APCC in each domain class, the trained preset blocks offer an overwhelming advantage in generating the reconstructed images with highest PSNR values. Even if some reconstructed images with highest PSNR values are not generated by the trained preset blocks, their PSNR values are very close to PSNR values of the reconstructed images generated by the trained preset blocks. The comparison with different preset blocks in several 512 × 512 images are shown in Table III, in which the range size is 8 × 8, and the domain step-length is 8 pixels, and $k = 100$.

B. Analysis of Computational Complexity

In the proposed scheme, the executing time in entire encoding process includes four parts. The first part is kept almost the same for natural images with the same size and image partition. It includes the time to construct the range pool and domain pool, and the time to compute the means and variances of domain blocks, and the time to standardize all the domain blocks. The second part is the time to classify and sort



(a) PSNR (dB), k, $dstep$. (b) time (ms), k, $dstep$.

Fig. 6. The curves with the reconstructed image quality computed by PSNR or the encoding time against the interval length k in four different image partitions with $dstep = 1$, 4, 8, and 16 for 512 × 512 image pepper. (a) The relation curves for PSNR, k, and $dstep$ for 512 × 512 image pepper. (b) The relation curves for encoding time, k, and $dstep$ for 512 × 512 image pepper.

TABLE III
PSNR OF THE RECONSTRUCTED IMAGES WITH DIFFERENT PRESET BLOCKS FOR IMAGES LENA, MILK DROP, GOLD HILL, PEPPER, AND F16

PSNR	center	barycenter	farthest	trained
gold hill	29.48	*29.50*	29.41	**29.69**
Lena	*31.54*	31.51	31.47	**31.81**
milk drop	34.40	*36.18*	35.97	**36.27**
pepper	30.93	30.88	*30.95*	**31.39**
F16	27.21	27.29	**27.54**	*27.48*

the domain blocks in each class. The third part for each range block is the time to compute its mean and variance, and the time of classification, and the time to select its domain set. The last part is the time to perform pairwise comparisons to search the final matching block for each range block.

As an example, the 512 × 512 image Lena with range size 4 × 4 and $dstep = 8$ is used to discuss the time involved in the encoding process.

The domain pool in the proposed scheme is the crude domain pool defined in Section III, which only has one eighth of the block number of the BFIC domain pool. Hence, the first part of encoding time related to domain blocks is about one eighth of the corresponding time in BFIC. The first part of encoding time related to range blocks is kept the same as BFIC because the range pool in the proposed scheme is as same as in BFIC. For the image Lena mentioned above, the first part of encoding time is about 2.5 ms.

Since the results of classifying blocks with different order of luminance sums in the sub-blocks have been obtained offline as in Table II, the classification of domain blocks needs only compute their luminance sums in the four sub-blocks for each block, which spends little time. In this paper, the quicksort method [27] is used to sort the domain blocks in each class. For the image Lena mentioned above, the second part of encoding time is about 1.5 ms.

The third part encoding time is kept the same for one range block whether how many blocks are chosen in its domain set. The reason is that we only find the beginning index and the end index in the ordered domain class to construct the domain set. For the image Lena mentioned above, this part of encoding time is about 6.9 ms, in which the runtime to construct the domain set by the binary search algorithm is about 3.3 ms.

The main encoding time is still appeared in the last part. We have to perform the pairwise comparisons to search the final matching domain block for a range block. The number of pairwise block-comparisons in which one range block involved is $2k$ with the selected domain set containing k blocks: k for $\mathbf{R}$ and the other k for $-\mathbf{R}$. Compared with BFIC in which one range block is compared with all the blocks in the BFIC domain pool, the proposed scheme can significantly reduce the computational complexity. For the image Lena mentioned above, the block number in the BFIC domain pool is 32 768. If we choose k equal to 20 in the proposed scheme, the number of pairwise block-comparisons in BFIC is as 32 768/40 = 819.2 times as that of the proposed scheme. If k is set to 100, the number of pairwise comparisons in BFIC is as 32768/200 = 163.84 times as that of the proposed scheme. The last part of encoding time for encoding the image Lena is 15.6 ms with $k = 20$, and 65.7 ms with $k = 100$.

C. The Relationship Between Different Parameters

In this paper, a domain set with k blocks is selected for each range block during encoding. Larger k enlarges the domain set, which can improve the reconstructed image quality, but the computational complexity for encoding an image is also increased. Hence, we must find a balance between the computational cost and the reconstructed image quality.

The domain step-length $dstep$ between two adjacent domain blocks in the image partition also has an important impact on the computational cost in the encoding process. In BFIC, it generates more domain blocks with a smaller domain step-length, which can improve the visual quality of the reconstructed images with increasing the computational cost in searching the matching pairs. However, in the proposed scheme, it can not be directly drawn the same conclusion.

The relationship between the reconstructed image quality, parameter k, and the domain step-length is shown in Fig. 6 (a). The curves shown in Fig. 6 (a) depict the PSNR values of the reconstructed image against the block number k in the domain set for image pepper in four different partitions with

PSNR=36.46 PSNR=35.17 PSNR=32.98

a) The reconstructed images with the encoding time equal to 26.5 ms

PSNR=38.06 PSNR=36.27 PSNR=34.56

b) The reconstructed images with the encoding time equal to 76.6 ms

Fig. 7. The reconstructed images for 512 × 512 images Lena, pepper, and man with $k = 20$ and $k = 100$ in the proposed scheme, respectively. The range blocks are partitioned with size 4 × 4 and $dstep = 8$. (a) The reconstructed images with the encoding time equal to 26.5 ms. (b) The reconstructed images with the encoding time equal to 76.6 ms.

$dstep = 1$, $dstep = 4$, $dstep = 8$, and $dstep = 16$, respectively. It can be seen that a positive correlation exists between the parameter k and the values of PSNR of the reconstructed images.

Moreover, an interesting trend can be obtained from Fig. 6(a): when k gets a small value, the reconstructed image quality with $dstep = 16$ is not worse than the reconstructed image quality with $dstep = 8$, and even better than the reconstructed image quality with $dstep = 1$ and $dstep = 4$. When k gets a little larger, the reconstructed image quality is better in the curve with $dstep = 8$. Further analysis indicates that the predominance of $dstep = 1$ or $dstep = 4$ just can be shown with a larger k. The reason can be found from Fig. 4: the blocks in the selected domain set are near the structure C, but the blocks we want just lie on or near l. If there are too many domain blocks with a smaller $dstep$, the blocks near l may be hard to enter the domain set containing only k blocks, thus the reconstructed image quality is likely to get worse. Certainly, for a large k, the scheme can also improve the image quality with increasing the number of the domain blocks.

The curves depicting the encoding time against the parameter k with different image partitions for image pepper are shown in Fig. 6(b). They correspond to $dstep$ equal to 1, 4, 8, and 16, respectively. From Fig. 6(b) we can see that an approximate linear relation exists between the parameter k and the encoding time. Moreover, the time difference becomes larger with larger parameter k for different $dstep$.

According to Fig. 6, PSNRs of the reconstructed images of pepper with $dstep = 16$ are very close to PSNRs of the reconstructed images with $dstep = 8$ when $k < 90$. Hence, if we use a small k to speed up encoding, we can let $dstep = 16$ to accelerate encoding further. Generally, it is a good idea to set $dstep$ as the side length or twice of the side length of the range block with a not large k in the proposed scheme.

For image partitions with different block sizes, the conclusions drawn above are also kept. For example, if the range blocks are partitioned with size 4 × 4, we can encode a 512 × 512 image with a small k and set $dstep$ as the side length or twice of the side length of the range block.

The reconstructed images for 512 × 512 images Lena, pepper, and man with range size 4 × 4 are shown in

Fig. 7 for $k = 20$ and $dstep = 8$, and for $k = 100$ and $dstep = 8$, respectively. We can see in Fig. 7 that the reconstructed image quality is hard to distinguish in human visual system (HVS) for different encoding time 26.5 ms and 76.6 ms.

D. Analysis About Classification in the Domain Pool

In Section III the domain pool is classified by Fisher's 3 classes method. It seems that the purpose of classification is only to speed up encoding. In fact, it not only speeds up encoding but also improves the reconstructed image quality when k is not large.

If the domain pool is not reduced and classified by the Fisher's 3 classes method, the number of blocks which meet the condition of **D** in Eq. (18) increases for the range block **R** and preset block **B**. It may reduce the probability of the blocks meeting Eq. (19) to enter the domain set containing only k blocks, which may lead to search a poor matching block for **R**. By classifying the domain pool using the Fisher's 3 classes method, the domain blocks are greatly reduced. It retains only one eighth of the blocks in the BFIC domain pool. Moreover, almost all the domain blocks which have higher probability to match with the transformed range blocks are remained according to Fisher's 3 classes method.

A double cone and a conic section perpendicular to the central axis of the double cone are shown in Fig. 8. The edge of the conic section is a circle and each element of the double cone goes through the circle by one point. We project all the domain blocks meeting Eq. (18) in BFIC through the lines connecting these blocks and the origin on the super-plane braced by the conic section. As in Fig. 8, the domain blocks meeting Eq. (18) in BFIC now all lie on a neighborhood of the circle on the same super-plane. According to the order of the luminance sums in the four sub-blocks, one domain blocks in BFIC can be classified into 24 classes. However, the transformed range blocks are definitely classified into the three classes expressed in Eq. (4), which are only one eighth of all the 24 classes. Hence, the domain blocks meeting Eq. (19) only lie on a small part near the circle, which means the other 21 domain classes are useless. If we only retain the domain blocks in the three classes in Eq. (4) for encoding, little effect could appear in the visual quality of the reconstructed image. Furthermore, the remained domain blocks are classified into three classes to increase the matching probability in each class. By contraries, if the domain blocks in other 21 classes are also retained, the blocks meeting Eq. (19) are hard to enter the domain set including only k blocks.

We name the scheme as the only-sorting scheme which directly sorts the domain pool according to Section IV without classifying the domain pool by the Fisher's 3 classes method. Compared with the only-sorting scheme, the block classification and sorting scheme is much more efficient and effective. The curve shown in Fig. 9 (a) depicts the relationship of the parameter k between the proposed block classification and sorting scheme and the proposed only-sorting scheme with range block size 8×8 and $dstep = 16$ for image pepper. Each point ($k1$, $k2$) on the curve means that the reconstructed

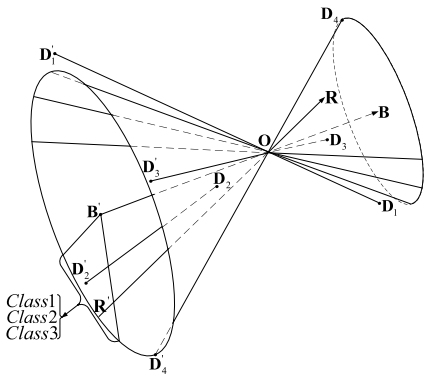


Fig. 8. The conic section of the cone meeting Eq. (18) for a range block **R** and the preset block **B**.

images have the same PSNR value for $k = k1$ in the block classification and sorting scheme and $k = k2$ in the only-sorting scheme. It is easy to see that the only-sorting scheme needs more blocks in the selected domain set to reach the same PSNR value, the reason is that it must enlarge the domain set to eliminate the impact of too many domain blocks in the only-sorting scheme. The curve for the comparison of the encoding time in two schemes is shown in Fig. 9 (b), we can see that the encoding process in the block classification and sorting scheme is much faster than in the only-sorting scheme to reach the same PSNR values.

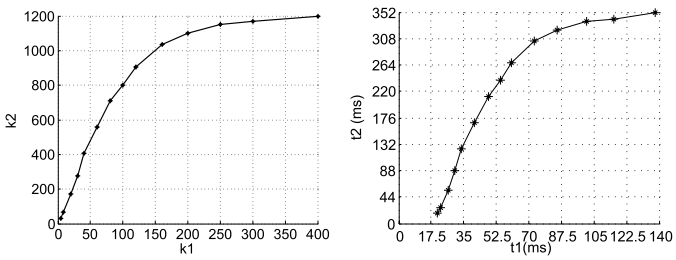
E. Comparison of Different Schemes

In this subsection, the proposed APCC-based classification and sorting scheme is compared with some other fast FIC schemes: the Fisher's 72 classes scheme (Fisher72) [15], the HOG-based clustering scheme (HOG-FIC) [16], Kovacs's classification scheme (Kovacs) [17], the variance-based sorting scheme (VBFC) [20], the one norm-based kick-out scheme (kickout) [21], and the only-sorting scheme proposed in Section IV in this paper (only-sorting). All these schemes, except kickout scheme, are implemented with the optimized energy function introduced in Subsection A of Section VI, which provides a fair comparison of performance between them.

For a natural image with size 512×512, if the range block is partitioned with size 8×8 and the domain step-length is chosen as 16 pixels, the encoding time is about 2.1 s in BFIC. If the range size is 4×4 and the domain step-length is 8 pixels, the encoding time is about 8.7 s in BFIC. However, MSE energy function is used to speed up encoding in the kickout scheme. Although the encoding time in the kickout scheme is less than the encoding time in BFIC with MSE energy function, the efficiency of the kickout scheme can not catch up with BFIC using the optimized energy function. For example, it needs 2.4 s and 16.4 s, respectively, to encode 512×512 image Lena with range size 4×4 and 8×8.

The comparison of these schemes for 512×512 image Lena with range block size 4×4 and $dstep = 8$ are shown in Fig. 10.

As shown in Fig. 10, VBFC has high efficiency in encoding, but the reconstructed image quality can not be well preserved.



(a) Comparison on parameter k. (b) Comparison on encoding time.

Fig. 9. Comparison between the proposed scheme and the only-sorting scheme for image pepper. The parameters $t1$ and $k1$ are the encoding time and the size of the selected interval in the proposed scheme, $t2$ and $k2$ correspond to the only-sorting scheme. In each point on the curves they have the same PSNR value of the reconstructed images in both two schemes. (a) Comparison on parameter k. (b) Comparison on encoding time.

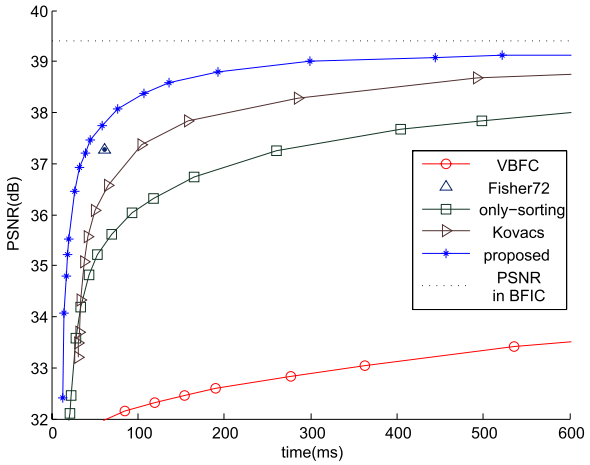


Fig. 10. Comparison with different schemes for 512 × 512 image Lena with range block size 4 × 4 and $dstep = 8$.

The Fisher72 scheme just lies a point in Fig. 10, which means it can not keep a balance between the reconstructed image quality and efficiency when we need. In addition, PSNR in Fisher72 scheme is 37.27 dB and the encoding time is 62.2 ms in Fig. 10. In the proposed scheme, PSNR of the decoded image is 37.32 dB with $k = 44$, and the encoding time is 41.8 ms. When the encoding time in the proposed scheme is about 61.8 ms which is near the encoding time in the Fisher72 scheme, the parameter k equals to 76 and PSNR of the reconstructed image is up to 37.82 dB. More experiments show that each point in the Fisher72 scheme is below the curve related to the proposed scheme for all the test images.

In Kovacs's scheme, the range and domain blocks are classified according to the polar angle and the normalized root mean square error (NMRS). Because both of the features are not equivalent to the affine similarity in FIC, the reconstructed image quality can not catch up with the proposed scheme when the encoding time is kept the same.

The reconstructed images in the only-sorting scheme have lower values of PSNR when the encoding time is kept the same as in the proposed block classification and sorting scheme.

The HOG-FIC scheme can not encode the image within 600 ms because it needs more time to extract HOG feature and classify blocks by K-means clustering algorithm online.

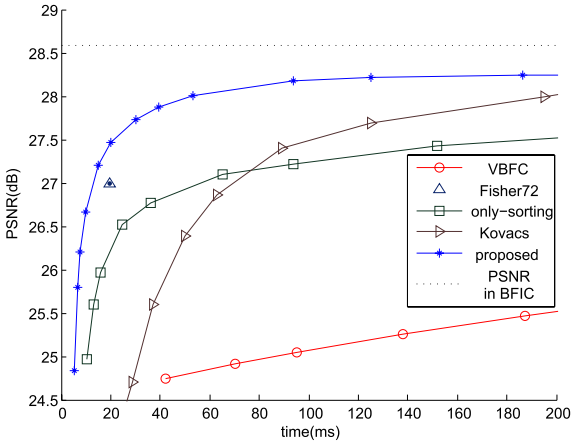


Fig. 11. The average time/PSNR comparison for different schemes on 16 512 × 512 images with range block size 8 × 8 and $dstep = 16$.

As shown in Fig. 10, the proposed scheme has a fastest accelerating rate to get close to PSNR in BFIC, and PSNRs in the proposed scheme are always on the top of these schemes when the encoding time is less than 0.6 s. Moreover, it is clear that the reconstructed image quality is very close to the reconstructed image quality in BFIC when the encoding time is about 0.3 s, about 1/29 of the encoding time in BFIC with only 0.3 dB image quality loss compared with BFIC.

The average time/PSNR comparison for different schemes on 16 images with range block size 8 × 8 and $dstep = 16$ is shown in Fig. 11. All the 16 images are with size 512 × 512, and they are aerial, baboon, barbara, Blonde, boat, bridge, couple, Elaine, F16, goldhill, house, Lena, man, milk drop, pepper, and sailboat, respectively. From Fig. 11 we can also see that the curve related to the proposed block classification and sorting scheme is always on the top of all the points or curves of these schemes.

The overall comparison with a larger time interval for image goldhill with range size 4 × 4 and $dstep = 8$ in different schemes is shown in Fig. 12. Obviously, the proposed scheme is the fastest speed climber from the beginning point. From the starting point to the end point, the curve corresponding to the proposed scheme is always on the top of these fast FIC schemes. The PSNR value of the end point on the curve of the proposed scheme is 37.21 dB. In Kovacas's scheme, when the classification number of polar angle and NMRS are set to 3 and 4, respectively, the encoding time is 1.41 s and PSNR value of the reconstructed image is 37.25 dB, which is the closest value to 37.21 dB. For a 512 × 512 image, the encoding time of 1.41 s is too long.

Furthermore, for these schemes, such as Fisher72, Kovacs, and VBFC, the encoding time increases greatly for images with larger size. Take the 512 × 512 and 1024 × 1024 images Elaine with range size 8 × 8 and $dstep = 16$ for example, the encoding time is 198.4 ms in the Fisher72 scheme to encode 1024 × 1024 image Elaine, which is about 10.8 times for it to encode 512 × 512 image Elaine (only 18.3 ms). It spends 4.07 s to encode the 1024 × 1024 image Elaine when both the classification numbers are set to 10 in Kovacs's scheme, which is as 25.6 times as the encoding time for it to encode

TABLE IV
COMPARISON OF THE DIFFERENT FIC SCHEMES FOR 1024 × 1024 IMAGES WITH RANGE SIZE 8 × 8

		VBFC	Fisher72	Kovacs (k_1, k_2)			Proposed					BFIC	kickout
		k=655		(35, 35)	(7, 8)	(4, 5)	k=10	k=44	k=492	k=1000	k=max		
Elaine	PSNR(dB)	35.11	36.07	36.08	37.00	37.15	35.11	36.07	37.00	37.09	37.13	37.34	37.34
	Time(s)	12.5	0.198	1.12	6.04	11.2	0.043	0.117	1.01	1.73	3.72	37.9	72.2
Man	PSNR(dB)	27.35	28.77	28.80	29.79	29.95	27.79	28.81	29.8	29.9	29.92	30.19	30.19
	Time(s)	3.60	0.204	1.17	6.33	11.7	0.043	0.118	1.00	1.82	3.52	37.9	71.8

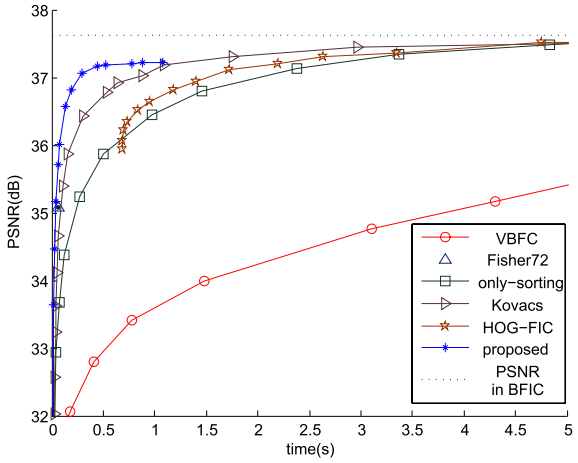


Fig. 12. The overall comparison with a larger time interval for image goldhill with range size 4 × 4 and $dstep = 8$.

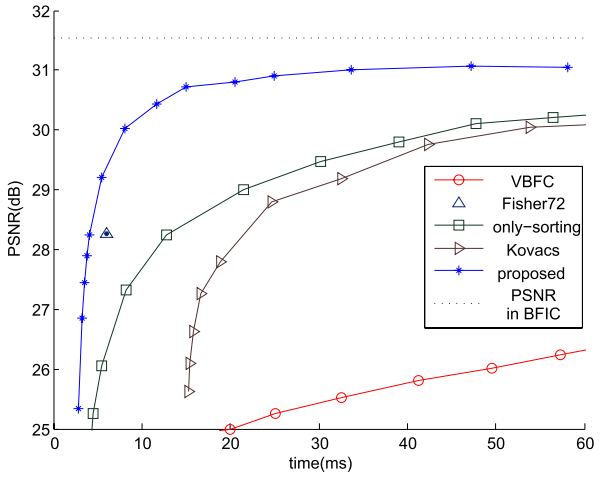


Fig. 13. Comparison with different schemes for 256 × 256 image cameraman with range block size 4 × 4 and $dstep = 8$.

the same image with the same classification numbers (only 159 ms). It spends 3.77 s when $k = 200$ in VBFC to encode 1024 × 1024 image Elaine, which is 62.5 times more than the encoding time when $k = 200$ for 512 × 512 image Elaine (only 60.4 ms). In the proposed scheme, the encoding time for $k = 10$ is 43.3 ms to encode 1024 × 1024 image Elaine, which is as 4.4 times as 9.9 ms to encode 512 × 512 image Elaine, and when $k = 44$ it is 117 ms to encode 1024 × 1024 image Elaine, which is as 4.3 times as 27.0 ms to encode 512 × 512 image Elaine.

It shows PSNRs and the encoding time for images Elaine and man with size 1024 × 1024 in different schemes in Table IV, from which we can see that PSNR values are still higher in the proposed scheme than other schemes when they have the same encoding time for the two images with size 1024 × 1024.

The curves with PSNR values of the reconstructed images against the encoding time for the 256 × 256 image cameraman with range block size 4 × 4 and $dstep = 8$ in different schemes are shown in Fig. 13.

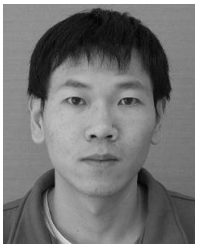
VIII. CONCLUSION

The over-long encoding time in fractal image compression is one of the major difficulties for its application. Although many researches were published to speed up encoding of FIC, the encoding time is still long, and in some schemes the reconstructed image quality becomes unacceptable. Based on the conclusion that the affine similarity between two image blocks in FIC is equivalent to the absolute value of Pearson's correlation coefficient between them, a new FIC scheme is proposed in this paper. Firstly, the Fisher's 3 classes method, which has been proved to be an APCC-based block classification method in this paper, is used to greatly reduce the number of blocks in the domain pool and classify the remaining domain blocks into 3 classes. Secondly, the domain blocks are sorted by APCCs between each domain block and the preset block in each class, and then the matching block for a range block is searched in a domain set selected by APCC with the preset block. Experiment results show that the proposed scheme can greatly reduce the encoding time with preserving the reconstructed image quality well.

REFERENCES

[1] M. F. Barnsley and A. E. Jacquin, "Application of recurrent iterated function systems to images," *Proc. SPIE*, vol. 1001, pp. 122–131, Nov. 1988.
[2] A. E. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations," *IEEE Trans. Image Process.*, vol. 1, no. 1, pp. 18–30, Jan. 1992.
[3] M. Pi, M. K. Mandal, and A. Basu, "Image retrieval based on histogram of fractal parameters," *IEEE Trans. Multimedia*, vol. 7, no. 4, pp. 597–605, Aug. 2005.
[4] J. H. Jeng, C. C. Tseng, and J. G. Hsieh, "Study on Huber fractal image compression," *IEEE Trans. Image Process.*, vol. 18, no. 5, pp. 995–1003, May 2009.
[5] M. Ghazel, G. H. Freeman, and E. R. Vrscay, "Fractal image denoising," *IEEE Trans. Image Process.*, vol. 12, no. 12, pp. 1560–1578, Dec. 2003.
[6] M. Ghazel, G. H. Freeman, and E. R. Vrscay, "Fractal-wavelet image denoising revisited," *IEEE Trans. Image Process.*, vol. 15, no. 9, pp. 2669–2675, Sep. 2006.
[7] S. S. Wang and S. L. Tsai, "Automatic image authentication and recovery using fractal code embedding and image inpainting," *Pattern Recognit.*, vol. 41, no. 2, pp. 701–712, 2008.

[8] S. G. Lian, "Image authentication based on fractal features," *Fractals*, vol. 16, no. 4, pp. 287–297, 2008.
[9] K. T. Lin and S. L. Yeh, "Encrypting image by assembling the fractal-image addition method and the binary encoding method," *Opt. Commun.*, vol. 285, no. 9, pp. 2335–2342, 2012.
[10] X. Tang and C. Qu, "Facial image recognition based on fractal image encoding," *Bell Labs Tech. J.*, vol. 15, no. 1, pp. 209–214, 2010.
[11] B. Wohlberg and G. de Jager, "A review of the fractal image coding literature," *IEEE Trans. Image Process.*, vol. 8, no. 12, pp. 1716–1729, Dec. 1999.
[12] H. Hartenstein, M. Ruhl, and D. Saupe, "Region-based fractal image compression," *IEEE Trans. Image Process.*, vol. 9, no. 7, pp. 1171–1184, Jul. 2000.
[13] T. K. Truong, C. M. Kung, J. H. Jeng, and M. L. Hsieh, "Fast fractal image compression using spatial correlation," *Chaos Solitons Fractals*, vol. 22, no. 5, pp. 1071–1076, 2004.
[14] X. Wang, Y. Wang, and J. Yun, "An improved fast fractal image compression using spatial texture correlation," *Chin. Phys. B*, vol. 20, no. 10, pp. 104202-1–104202-11, 2011.
[15] Y. Fisher, E. W. Jacobs, and R. D. Boss, "Fractal image compression using iterated transforms," *Image Text Compress.*, vol. 176, pp. 35–61, May 1992.
[16] W. R. Schwartz and H. Pedrini, "Improved fractal image compression based on robust feature descriptors," *Int. J. Image Graph.*, vol. 11, no. 4, pp. 571-587, 2011.
[17] T. Kovacs, "A fast classification based method for fractal image encoding," *Image and Vision Computing*, vol. 26, no. 8, pp. 1129–1136, 2008.
[18] R. Distasi, M. Nappi, and D. Riccio, "A range/domain approximation error-based approach for fractal image compression," *IEEE Trans. Image Process.*, vol. 15, no. 1, pp. 89–97, Jan. 2006.
[19] Y. Zhou, C. Zhang, and Z. Zhang, "An efficient fractal image coding algorithm using unified feature and DCT," *Chaos Solitons Fractals*, vol. 39, no. 4, pp. 1823–1830, 2009.
[20] C. He, S. Yang, and X. Huang, "Variance-based accelerating scheme for fractal image encoding," *Electron. Lett.*, vol. 40, no. 2, pp. 1052–1053, 2004.
[21] H. N. Chen, K. L. Chung, and J. E. Hung, "Novel fractal image encoding algorithm using normalized one-norm and kick-out condition," *Image Vis. Comput.*, vol. 28, no. 3, pp. 518–525, 2010.
[22] C. He, X. Xu, and G. Li, "Improvement of fast algorithm based on correlation coefficients for fractal image encoding," *Comput. Simul.*, vol. 12, no. 4, pp. 60-63, 2005.
[23] J. Wang, Y. Liu, P. Wei, Z. Tian, Y. Li, and N. Zheng, "Fractal image coding using SSIM," in *Proc. 18th ICIP*, Sep. 2011, pp. 245–248.
[24] A. E. Jacquin, "Fractal image coding: A review," *Proc. IEEE*, vol. 81, no. 10, pp. 1451–1465, Oct. 1993.
[25] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Process. Lett.*, vol. 9, no. 3, pp. 81–84, Mar. 2002.
[26] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? A new look at signal fidelity measures," *IEEE Signal Process. Mag.*, vol. 26, no. 1, pp. 98–117, Jan. 2009.
[27] C. A. R. Hoare, "Quicksort," *Comput. J.*, vol. 5, no. 1, pp. 10–16, 1962.



Jianji Wang (S'11) received the B.S. degree in applied mathematics from the School of Science, Xi'an Jiaotong University, Xi'an, China, in 2003, and the M.E. degree in computer science and engineering from the School of Electronic and Information Engineering, Xi'an Jiaotong University, in 2007, where he is currently pursuing the Ph.D. degree with the Institute of Artificial Intelligence and Robotics. His current research interests include fractal image coding and image processing.

Nanning Zheng (F'06) received the B.E degree from the Department of Electrical Engineering, Xi'an Jiaotong University, Xi'an, China, in 1975, the M.E. degree in information and control engineering from Xi'an Jiaotong University, in 1981, and the Ph.D. degree in electrical engineering from Keio University, Tokyo, Japan, in 1985. He is currently a Professor and the Director of the Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University. His current research interests include computer vision, pattern recognition, computational intelligence, image processing, and hardware implementation of intelligent systems. He has served as the General Chair for the International Symposium on Information Theory and its Applications in 2002 and the General Co-Chair for the International Symposium on Nonlinear Theory and its Applications in 2002. Since 2000, he has been the Chinese representative on the Governing Board of the International Association for Pattern Recognition. He currently serves as an Executive Editor of the Chinese Science Bulletin. He was a member of the Chinese Academy Engineering in 1999.