International Scholarly Research Network
ISRN Signal Processing
Volume 2011, Article ID 359592, 9 pages
doi:10.5402/2011/359592

Research Article

Combining Fractal Coding and Orthogonal Linear Transforms

Michele Nappi and Daniel Riccio

Biometric and Image Processing Laboratory (BIPLab), University of Salerno, via Ponte don Melillo, 84084 Fisciano (SA), Italy

Correspondence should be addressed to Daniel Riccio, driccio@unisa.it

Received 2 February 2011; Accepted 8 March 2011

Academic Editors: M. Barkat and W. Liu

Many desirable properties make fractals a powerful mathematic model applied in several image processing and pattern recognition tasks: image coding, segmentation, feature extraction, and indexing, just to cite some of them. Unfortunately, they are based on a strong asymmetric scheme, consequently suffering from very high coding times. On the other side, linear transforms are quite time balanced, allowing them to be usefully exploited in realtime applications, but they do not provide comparable performances with respect to the image quality for high bit rates. In this paper, we investigate different levels of embedding orthogonal linear transforms in the fractal coding scheme. Experimental results show a clear improved quality for compression ratios up to 15 : 1.

1. Originality and Contribution

The literature about both linear transform-based image compression (discrete cosine transform—DCT, Discrete Wavelet Transform—DWT) and fractal image coding is sizeable. Nevertheless, there are still few contributions exploring possible fusions of the two approaches. From a theoretic point of view, the novelty of the proposed paper is just to investigate several ways in which the former can embed, or even be embedded in, the latter. As pointed out by the experimental results, fusing these methodologies allows to significantly speed up the fractal coding process, while retaining most of the objective quality of the decoded image, even at high compression rate. This represents a highly desirable feature for image coders in all those applications managing high-resolution images (e.g., GIS, satellite image databases, and cultural heritage).

2. Introduction

Fractal image compression is based on the self-similarity property of an image, and performs image compression by applying a series of transformations to the image. To perform image decompression, these transformations are applied iteratively until the system converges, a condition which may be assessed by using the Hutchinson metric. The main disadvantage of previous techniques for fractal image compression is the large amount of computation that they require. Several methods have been proposed in order to speed up fractal image coding [1, 2]. Speedup methods based on nearest neighbour search by feature vectors outperform all the others in terms of decoded image quality at a comparable compression rate [3], but they often suffer from the high dimensionality of the feature vector [4]; Saupe's operator represents a suitable example. To cope with this drawback, dimension reduction techniques are introduced. Saupe reduced the dimension of the feature vector by averaging pixels while in [5] discrete cosine transform (DCT) is used to cut out redundant information.

In the same way, linear transforms (LT) have been widely exploited to extract representative features or to codify groups of pixels in image indexing and compression applications. Indeed, Linear transforms form the basis of many compression systems as they decorrelate the image data and provide good energy compaction. For example, the discrete fourier transform (DFT) [6] is used in many image processing systems while DCT [6] is used in standards like JPEG, MPEG, and H.261. Still others are Walsh-Hadamard transforms (WHT) [6] and Haar Transforms (HT) [6]. There are many attempts in the literature for combining fractal coding with the discrete cosine transform [7–9] or with the discrete wavelet transform [10, 11], all of them facing the

speedup problem of the coding phase. In more detail, in [7], image blocks are classified as smooth, diagonal/subdiagonal edge, and horizontal/vertical edge. The classification operation is performed using only the lowest horizontal and vertical DCT coefficients of the given block. Since the scheme is simple, the overhead is minor compared to the complexity of the encoder, but the speedup it can achieve is no larger than 3, which is quite smaller than that achievable by nearest neighbour search based methods. Zhao and Yuan, previously proposed a similar approach [9], in which the most attention was paid for reducing the bit rate; indeed, the original image is first compressed by DCT domain fractal transform instead of spatial domain fractal transform, then the difference image between the original image and its fractal approximation is quantized and encoded by a Huffman code. In this case, image ranges and domains are only partitioned in low and high-activity blocks while full search is performed only for the latter. A Similar approach has been developed by Zhang and Po in [11] by using HT instead of DCT. Range blocks and corresponding domain blocks are classified into edge selective categories by the energy compacted wavelet coefficients. The searching is carried out between the same classes for range/domain comparisons which significantly reduces the computational complexity, so that the speedup achieved is up by 12-times over the conventional full search method. All these methods make use of the lowest horizontal and vertical DCT/HT coefficients of a given block, but not exploiting nearest coefficients that also have an outstanding hand in the total energy. Fractal coding in wavelet domain has been also investigated by using the wavelet zerotree as in [8, 10], where a fractal method is adaptively applied to the parts of an image that can be encoded more efficiently relative to an EZW (embedded zerotree wavelet) coder at a given rate. However, in these cases wavelet coefficients are not used to speed up the coding phase but to improve the decoded image quality at a given rate. Hence, this paper sets as its main goal of investigation the ways of embedding an orthogonal LT T into the standard PIFS- (partitioned iterated function systems-) based coding scheme. In more details, at first, linear properties of T are exploited to dramatically reduce the computational cost of the encoding phase, by arranging its coefficients in a suitable way, Figure 1 depicts the architecture of such a scheme. Range and domain blocks are extracted from the input image, and an orthogonal linear transform (LT) is applied to extract feature vectors. Domain feature vectors are organized in a tree data structure, while range feature vectors are used to search the best approximating domain in the tree during the coding phase, speeding up the range/domain matching process. The residual information is computed as the difference between LT transformed versions of the original range and the approximated range; residual coefficients are quantized and stored in the final bit stream. In Figure 2, the decoding process is shown. The fractal decoded image is obtained iteratively by applying affine transformations to an all-black image. The T transform is applied to fractal decoded ranges while residual coefficients are dequantized separately; transformed range coefficients are summed with residual information and then T^{-1} inverse transformed to obtain the output range. The hybrid approach significantly outperforms Saupe's method for relatively small compression ratios, while becoming comparable when the bit-rate increases.

3. Theoretical Concepts

Partitioned iterated function systems (PIFS) consist of a set of local affine contractive transformations, which exploits the image self-similarities to cut out redundancies, while extracting salient features. In more details, given an input image I, it is partitioned into a set $R = \{r_1, r_2, \ldots, r_{|R|}\}$ of disjoint square regions of size $|r| \times |r|$, named *ranges*. Another set $D = \{d_1, d_2, \ldots, d_{|D|}\}$ of larger regions is extracted from the same image I. These regions are called *domains* and can overlap. Their size is $|d| \times |d|$, where usually $|d| = 2|r|$. Since a domain is quadruple sized with respect to a range, it must be shrunk by a 2×2 average operation on its pixels. Due to the large number of domains, downsampling operations will introduce an overhead if performed for each range/domain comparison; the same result can be obtained in a more effective way by downsampling the whole image $I_C = c(I)$. The downsampled image I_C is a quarter of the input image I, so that downsampled domains are directly extracted from I_C with side-length $|r| \times |r|$.

The image I is encoded range by range: for each range r, it is necessary to find a domain d and two real numbers α and β such that the quadratic error is minimized with respect to the Euclidean norm. It is customary to impose that $|\alpha| \leq 1$ in order to ensure convergence in the decoding phase. To find the best approximating domain d, however, requires an exhaustive search over the whole set D, which is an impractical operation. Therefore, ranges and domains are classified by means of a feature vector in order to reduce the cost of searching the domain pool: if the range r is being encoded, only the domains having a feature vector close to that of r are considered [12].

On the other hand, a transformation T is called linear if it has two mathematical properties: (a) additivity ($T(x + y) = T(x) + T(y)$) and (b) homogeneity ($T(\alpha x) = \alpha T(x)$). The linear transform domain features are very effective when the patterns are characterized by their spectral properties; so, here, we investigate the feature extraction capability of the discrete fourier transform (DFT), the discrete cosine transform and the Haar transform (HT) [6].

4. Speeding up the Coding Phase

In order to reduce the computational cost of exhaustive search while still preserving a good image quality, we scan the domains in the pool to compute feature vectors that are organized in a K-dimensional-tree structure [13] (KD-tree) and will help us to choose the most promising candidate domains for encoding a given range.

Let r and d be a range and a domain block, respectively, and let T be a two-dimensional orthogonal linear transform (FFT, DCT or HT), the range block r can be rewritten in terms of d by applying an affine transformation $r = \alpha \cdot d + \beta \cdot o(1) + e$, where α and β are two scalar values, $o(1)$ is an all ones matrix of size $|r| \times |r|$, and e is the $|r| \times |r|$ matrix

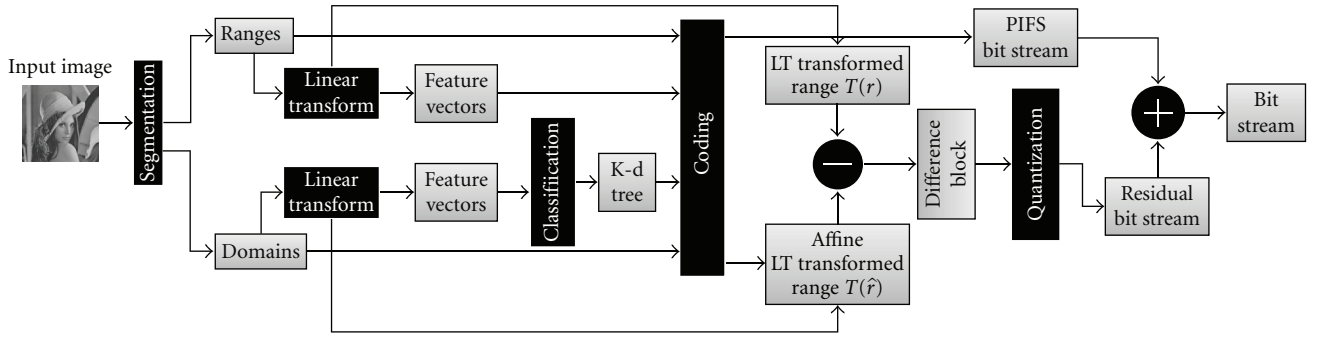

Figure 1: The architecture of the proposed fractal coder.

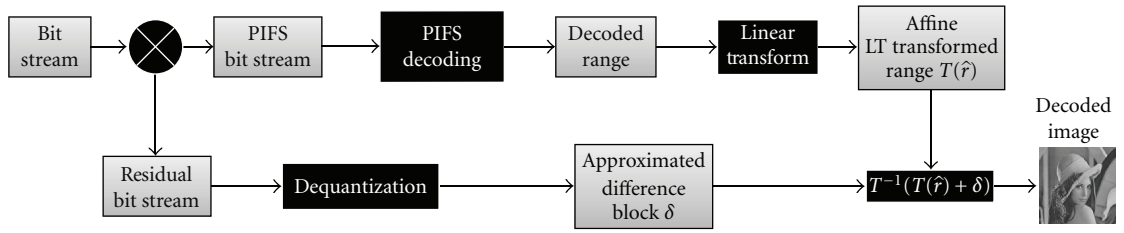

Figure 2: The architecture of the proposed fractal decoder.

of the residual error. As the PIFS decoder has no knowledge of the residual error e, it must be minimized during coding process, so that the $\hat{r} = \alpha \cdot d + \beta \cdot o(1)$ well approximates $r = \alpha \cdot d + \beta \cdot o(1) + e$. A feature vector u can be extracted from $\hat{r}$ and d by reorganizing the coefficients of the transformation T;

$$\begin{aligned} &\hat{r} = \alpha \cdot d + \beta \cdot o(1) \\ &\overset{\text{Applying } T}{\Longrightarrow} T(\hat{r}) = T(\alpha \cdot d + \beta \cdot o(1)) \\ &\overset{\text{Linearity of } T}{\Longrightarrow} T(\hat{r}) = \alpha \cdot T(d) + \beta \cdot T(o(1)) \\ &\overset{\text{Transforming } \overline{\beta}}{\Longrightarrow} T(\hat{r}) = \alpha \cdot T(d) + \beta \cdot O, \end{aligned} \tag{1}$$

where

$$O = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}. \tag{2}$$

Γ being the transformed domain $T(d)$, the transformed range can be rewritten as

$$T(r) = \begin{bmatrix} \alpha \cdot \Gamma^{00} + \beta & \alpha \cdot \Gamma^{01} & \cdots & \alpha \cdot \Gamma^{0n-1} \\ \alpha \cdot \Gamma^{10} & \alpha \cdot \Gamma^{11} & \cdots & \alpha \cdot \Gamma^{1n-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha \cdot \Gamma^{n-10} & \alpha \cdot \Gamma^{n-11} & \cdots & \alpha \cdot \Gamma^{n-1n-1} \end{bmatrix}. \tag{3}$$

Notice that only the first term of $T(r)$ is affected by β, and it represents the mean of r. As the main desired property of the feature vector is the independence from α and β, the first element of the $T(r)$ matrix is then discarded while the remaining ones are rearranged in a linear vector $u = \{\Gamma^{01}, \Gamma^{10}, \ldots, \Gamma^{n-1n-1}\}$ of dimension $n^2 - 1$ by means of a zig-zag scanning that starts from the position $(0, 1)$. In order to also cancel out effects of α on u, its elements are divided by the quantity $\overline{u}$, indeed,

$$\overline{u} = \frac{1}{n^2 - 1} \sum_{i=0}^{n^2-1} \alpha \cdot \Gamma_i = \alpha \cdot \frac{1}{n^2 - 1} \sum_{i=0}^{n^2-1} \Gamma_i = \alpha \cdot \overline{\Gamma}. \tag{4}$$

Finally, the real feature vector $\overline{u}$ is given by

$$\begin{aligned} \overline{u} &= \left\{ \frac{\alpha\Gamma^{01}}{\overline{u}}, \frac{\alpha\Gamma^{10}}{\overline{u}}, \ldots, \frac{\alpha\Gamma^{n-1n-1}}{\overline{u}} \right\} \\ &= \left\{ \frac{\alpha\Gamma^{01}}{\alpha\overline{\Gamma}}, \frac{\alpha\Gamma^{10}}{\alpha\overline{\Gamma}}, \ldots, \frac{\alpha\Gamma^{n-1n-1}}{\alpha\overline{\Gamma}} \right\} \\ &= \left\{ \frac{\Gamma^{01}}{\overline{\Gamma}}, \frac{\Gamma^{10}}{\overline{\Gamma}}, \ldots, \frac{\Gamma^{n-1n-1}}{\overline{\Gamma}} \right\}. \end{aligned} \tag{5}$$

5. Residual Information Coding

Linear transformations can be exploited within the fractal coding approach to a further extent, the residual information encoding. Indeed, in the proposed hybrid approach, the transform T is applied to each of the pool domains during indexing stage while it is applied to every range during encoding. Consequently, during encoding, $T(r)$ and $T(d)$ are

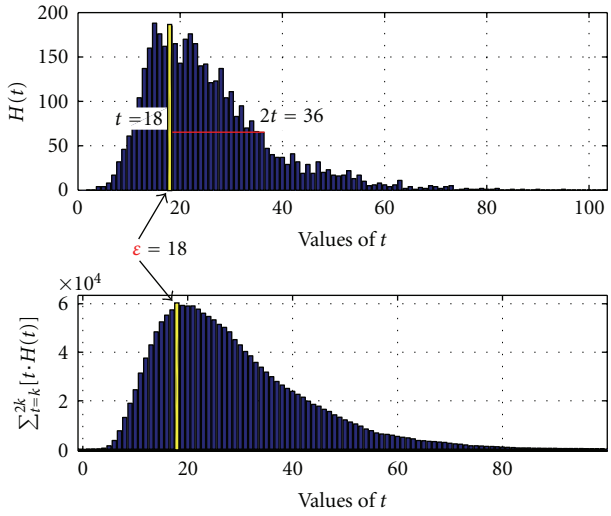

Figure 3: Graphical representation of the $H(t)$ histogram for the Lena image at a compression ratio of 16.5.

both known while the transform $T(\hat{r})$ of approximate range $\hat{r}$ can be simply obtained as $T(\hat{r}) = T(\alpha \cdot d + \beta \cdot o(1))$ by exploiting T linearity (as discussed in previous sections) without any further computing cost. Coefficients of the T transformation can be further exploited to save part of the original energy lost through fractal encoding. More precisely, as $\hat{r}$ is calculated by minimizing the approximation error e for the range r, we can assume $\delta_r = T(r) - T(\hat{r})$ is represented by average small values which can be quantized by a reduced number of bits. To this aim, each $\delta_r(i, j)$ is rounded to the nearest integer. Moreover, during decoding, the approximate range $\hat{r}$ is obtained by means of fractal decoding, so, knowing δ_r and $T(\hat{r})$, we could recover $T(r)$ in terms of $T(\hat{r}) + \delta_r$. It is clear indeed that the core of following discussion is the proposal of a highly efficient strategy to store if not all, at least a fraction of information contained in δ_r while still preserving a useful quality/bit rate ratio. The compact representation of values $\delta_r(i, j)$ within the file is the most crucial aspect of the proposed method. Hereafter, this topic is presented in detail. The greater the module of a coefficient $\delta_r(i, j)$ the more relevant its encoding accuracy. The range of δ_r coefficients is not formally bounded, but we need to restrict this interval to a limited range in order to maximize the effectiveness of the proposed quantization strategy. Experimentally, we found that the most of the energy of δ_r coefficients could be isolated by selecting only those falling in a limited range $[\epsilon, 2 \cdot \epsilon]$. The constant ϵ is an integer value, which is computed so that modules of $\delta_r(i, j)$ coefficients near to $3/2\epsilon$ for any $r \in I$ are maximized. The $\delta_r(i, j)$ coefficients are approximated by $3/2\epsilon$ if their module falls in the range $[\epsilon, 2 \cdot \epsilon]$ and set to zero otherwise. To this regard it is possible to define a criterion to automatically set the ϵ value. For every range block r in I, $\delta_r(i, j)$ coefficients are computed and a global histogram H is generated, where $H(t)$ is the number of times the module $|\delta_r(i, j)|$ of some $\delta_r(i, j)$ equals $3/2t$. Thus, the value of ϵ is given by $\epsilon = \text{argmax}_k \sum_{t=k}^{2k}(tH(t))$. A graphical example of Lena's histogram together with the corresponding value for ϵ with a compression ratio of 16.5 is reported in Figure 3.

An even greater attention has to be paid for quantization and encoding of values $\delta_r(i, j)$. After ϵ has been computed, coefficients with $|\delta_r(i, j)|$ lower than ϵ are set to zero, so that only values greater than threshold ϵ can be found within each block. So if a block does not contain any $|\delta_r(i, j)| > 0$, this means that it has not useful residual information and it needs no further processing than a flag set to 0 in the bitstream, otherwise the flag is set to 1. Because many rows i of the block δ_r can be empty, a further bit-rate reduction can be achieved separating the rows i containing $\delta_r(i, j) \neq 0$ from those containing $\delta_r(i, j) = 0$, for all j. More precisely, each row i of block δ_r is associated to a bit b, where $b = 1$ if there exist $j \mid \delta_r(i, j) \neq 0$ otherwise $b = 0$, for a total of $|r|$ bits (where $|r|$ is the side length of the range block r). In the proposed approach, $\delta_r(i, j)$ coefficients are approximated to a constant value of $3/2\epsilon$ (the center of range $[\epsilon, 2\epsilon]$) rather than being singularly quantized; so that for each coefficient it is sufficient to store in the file just the sign and column info j. As this results in block δ in a sparse matrix, each element can be considered as an ordered couple (i, j); while due to the aforementioned representation (which distinguishes between empty and notempty rows), it is possible to further reduce the bit-rate. Considering that the values $\delta_r(i, j)$ are consecutively read from the file, one row after the other, it is possible to save in the file only the column j of each coefficient as for all the coefficients sharing the same row the columns are increasingly ordered. Consequently, while reading a column sequence, if the one just read is smaller than the previous one, this implies that the present column index belongs to a following row. Whereas the correct row to which the column sequence belongs can be retrieved as the first notempty row (previously flagged with a 1) and not necessarily be the very following row $i + 1$, because empty rows are frequent. In the end, for each block containing residual information, only $1 + |r| + 2 \cdot K \cdot \log_2(|r|)$ bits are saved, where K is the number of $\delta_r(i, j) \neq 0$, otherwise just a single bit set to 0 is saved. A graphical representation of the residual encoding process is provided in Figure 4.

Overall, the residual encoding framework can be resumed in the following lines.

Coding Process.

(a) Delta computation $\delta_r(i, j) = T(r) - T(\hat{r})$, $0 \leq i, j < |r|$,

(b) If $|\delta_r(i, j)| < \epsilon$ or $|\delta_r(i, j)| > 2 \cdot \epsilon$ set $\delta_r(i, j) = 0$,

(c) For all $i, j \mid \delta_r(i, j) > 0$ write $\delta_r(i, j)$ in compact form.

Decoding Process.

(a) Image decoding I through PIFS,

(b) Read δ_r from file,

(c) For each range $\hat{r}$ with $\delta_r \neq 0$,

(d) compute $T(\hat{r})$ and replace $\hat{r}$ with $T^{-1}(T(\hat{r}) + \delta)$ in image I.

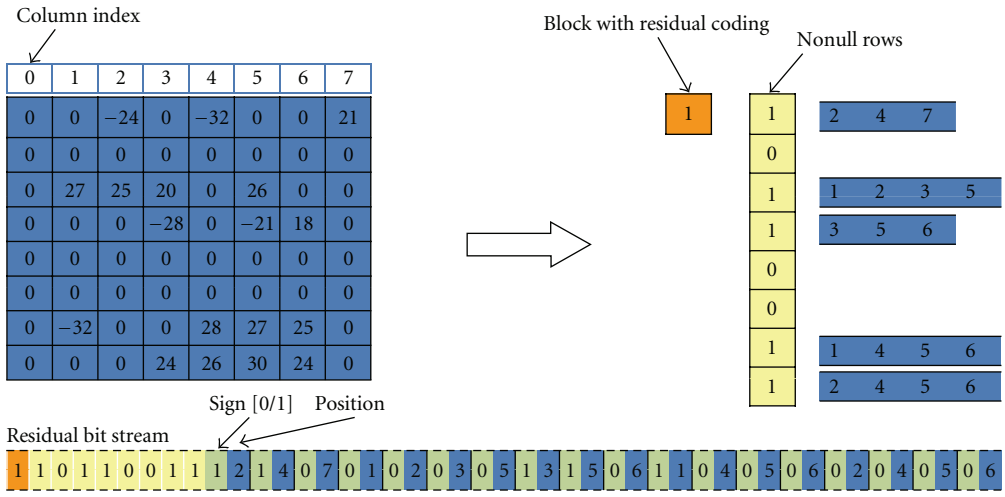

FIGURE 4: A real example of the residual coding with $\epsilon = 18$.

6. Experimental Results

Tests have been conducted on a dataset of twenty images, twelve of them coming from the Waterloo Bragzone Standard database [14], and the remaining eight from the web. A large variability in testing conditions has been ensured by selecting test images containing patterns, smooth regions, and details. They are all 8-bit grayscale images at a resolution of 512 by 512 pixels. The main aim of our tests is to assess the efficiency of the LT-based feature vector and the improvements provided by LT coding of residual information. The bit allocation for the range/domain approximation parameters is set as follows: 4 bits for α, 7 bits for β, $\log_2(|D|)$ bits for the domain coordinates (where $|D|$ represents the size of the domain pool), and 3 bits for the isometry. The compression ratio has been calculated as the ratio between the original image size and the encoded image size while the objective image quality has been measured in terms of peak signal-to-noise ratio (PSNR). In order to further assess the performance of the hybrid scheme, we also compared it with Saupe's algorithm [3].

A comparison with Saupe's algorithm, as shown in Figure 5, underlines the particular behavior of the hybrid scheme 3 variants (PIFS-LTD, PIFS-LTH, and PIFS-LTF). It obviously comes out that the FFT provides very poor performances, which represents a further confirmation that LT yielding real and imaginary coefficients are not effective at all when applied into the PIFS coding. Figure 5 also points out that DCT- and Haar-based feature vectors (in PIFS-LTD and PIFS-LTH, resp.) have almost quite comparable performances. Furthermore, they show better performances than the FFT (PIFS-LTF) and Saupe's vector. The main reason motivating the superiority of the PIFS-LTD and PIFS-LTH is that DCT and Haar transforms retain the most of the image information in its first coefficients, so when a shorter vector is obtained by truncating the original one to a little number of coefficients, more representative features are retained. On the contrary, this does not happen for Saupe's vector that is usually reduced by averaging its components.

In Figure 6, the results obtained through the integration of both domain classification method (PIFS-LTD and PIFS-LTH) and linear transform-based residual information encoding (in this case DCT) are shown. The advantage provided by residual information encoding is more relevant for small compression ratios. The reason for this can be found considering that for high-quality fractal decoded images the residual features relatively small values which are more easily encoded via the proposed technique. Figure 7 also confirm this thesis. The first image in its top row is the original image while the following two show the nonresidual coded block in blue at two different compression ratios, 4.8 and 11.5. Notice that residual coded pixels in the last column outnumber that in the second one, because of the larger distortion introduced by the fractal encoder. The second and third rows show a detail of the decoded image with (third row) and without (second row) residual coding, underlining that details added by residual information can damp blocking artifacts.

In Table 1, encoding times and corresponding PSNR values with and without residual coding are reported for both Saupe- and LT-based fractal encoders. Results refer to the 512×512 grayscale Lena image while the system runs on an Intel Core Duo 2.0 Ghz with 1 GB RAM. Results in Table 1 clearly underline the superiority of the LT-based classification scheme with respect to Saupe's approach. This is due to the ability of Linear Transformations (such as the DCT) to compact the most part of the signal energy in their first coefficient, turning in a more accurate indexing of the range/domain pool. The last column also points out that residual coding contribution is of about 0.5–0.8 dB to the final quality estimation of the decoded image.

The hybrid scheme proposed in this paper has also been compared with state-of-the-art classification approach such as [7]. The outcomes of this comparison are reported in Table 2. The first row shows time and PSNR of the full range/domain search for the 256×256 grayscale Lena image while the second row reports analogous information when

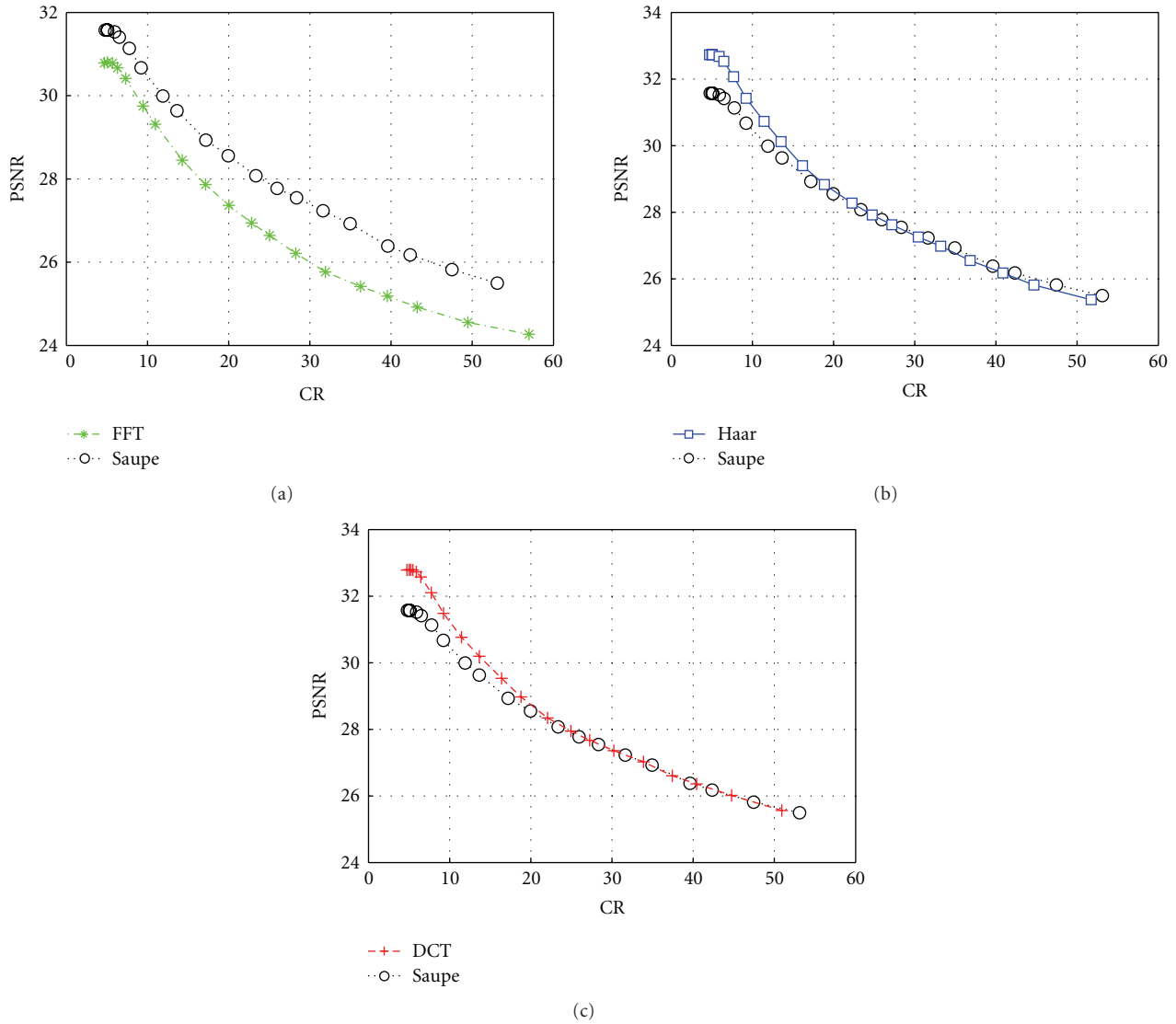

Figure 5: Average PSNR/CR comparison over all the test images for the hybrid approaches PIFS/LT and Saupe's method.

Table 1: Comparison among Saupe's operator and several LT-based feature vectors on the Lena image 512 × 512 for a compression ratio fixed at 16.5 without residual information coding (first two columns) and with residual information coding (last two columns).

Method	Without residual coding		With residual coding	
	Time	PSNR	Time	PSNR
Saupe	20.05 sec	32.90 dB	23.05 sec	33.44 dB
PIFS-LFT	21.35 sec	31.92 dB	24.67 sec	32.45 dB
PIFS-LTD	23.15 sec	33.94 dB	26.50 sec	34.60 dB
PIFS-LTH	22.45 sec	33.95 dB	25.24 sec	34.80 dB

Duh's speedup method is adopted. As it can be seen Duh's classification scheme produce a speedup of about a factor 3, confirming what the authors claim in [7]. On the contrary, the proposed speedup technique provides a speedup of about a factor 150 without the residual coding and 90 in the case the residual information is encoded too. This is attributable to the number of range/domain comparisons performed during the encoding process. Indeed, Duh's approach divides the total amount of range/domain comparisons by 3, while in the proposed scheme the each range is compared with no more than 50 domains with a feature vector similar to that of the range.

A further comparison with state of the art techniques is given by considering the hybrid fractal/wavelet approach proposed by Iano et al. [15] as shown in Table 3. This technique presents a certain affinity to the proposed method,

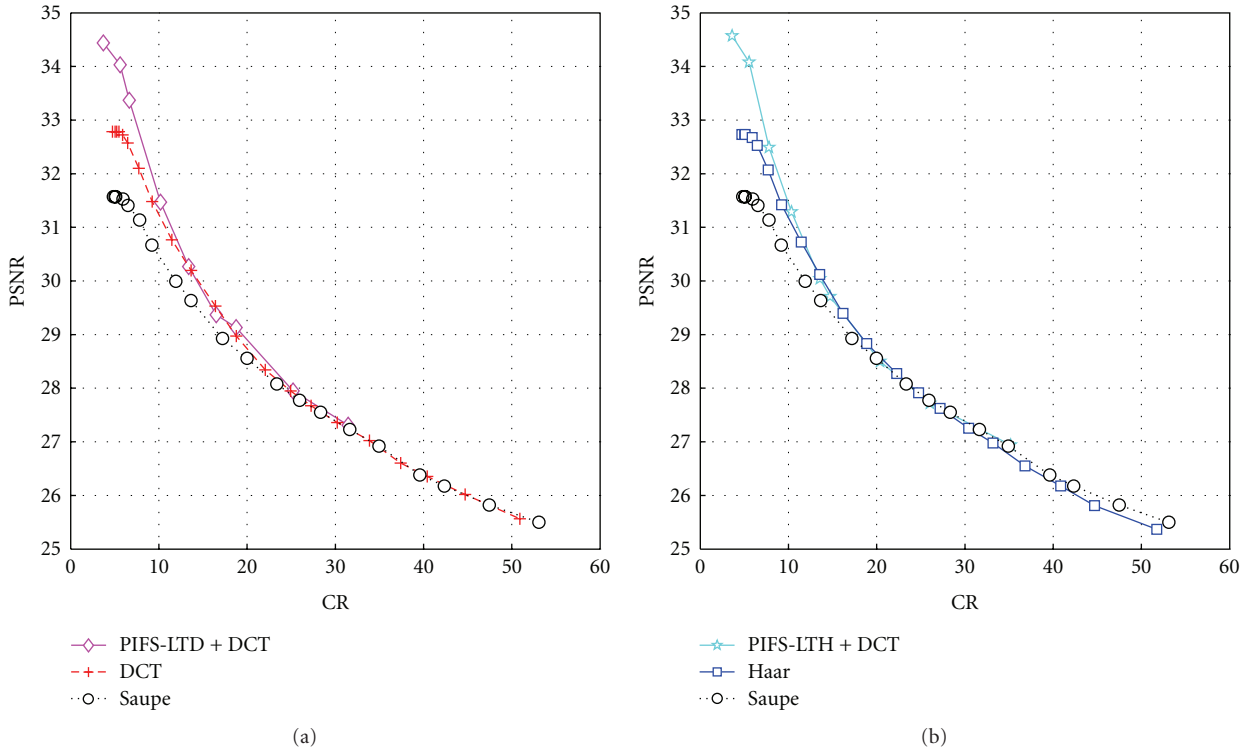

Figure 6: Average PSNR/CR comparison over all the test images for the hybrid approaches PIFS/LT with/without DCT residual coding and Saupe's method.

Table 2: Comparison among Duh's approach and several LT-based feature vectors on the Lena image 256 × 256 for a compression ratio fixed at 18.28 without residual information coding (first two columns) and with residual information coding (last two columns).

Method	Without residual coding		With residual coding	
	Time	PSNR	Time	PSNR
Full search	437.94 sec	28.90 dB	—	—
Duh	144.55 sec	28.40 dB	—	—
Saupe	2.87 sec	27.10 dB	4.54 sec	27.90 dB
PIFS-LFT	2.65 sec	25.80 dB	3.80 sec	26.10 dB
PIFS-LTD	2.81 sec	27.90 dB	4.10 sec	28.10 dB
PIFS-LTH	2.98 sec	27.80 dB	4.01 sec	28.02 dB

Table 3: Comparison among the set partitioning in hierarchical trees (SPIHT), Iano's and Saupe's approach, and several LT-based feature vectors with residual information coding on the 512 × 512 Lena image, for a bpp fixed at about 0.48.

Method	Bit rate	PSNR
SPIHT	0.48 bpp	32.59 dB
Iano	0.49 bpp	32.81 dB
Saupe	0.48 bpp	33.41 dB
PIFS-LFT	0.45 bpp	32.38 dB
PIFS-LTD	0.42 bpp	34.50 dB
PIFS-LTH	0.46 bpp	34.73 dB

as they both code base information through fractals, then exploiting additional processing to code the residual (detail) information.

7. Conclusions

In this paper, we presented two different ways to exploit linear transformations for a PIFS-based fractal encoding strategy. Indeed, linear transformations have been used to define a range/domain classification vector which drastically reduces the computational weight of encoding stage, by avoiding most of least significant range/domain comparison. Furthermore, linear transform properties have been exploited for residual information encoding, resulting in a decoded image quality far superior than in classic fractal approaches. A comparison with the Saupe fractal approach clearly shows the improvements achieved through the proposed method while a comparison with three among the most known linear transforms (FFT in PIFS-LTF, DCT in PIFS-LTD and Haar in PIFS-LTH) confirms how all of these differently contribute to enhance the basic scheme.

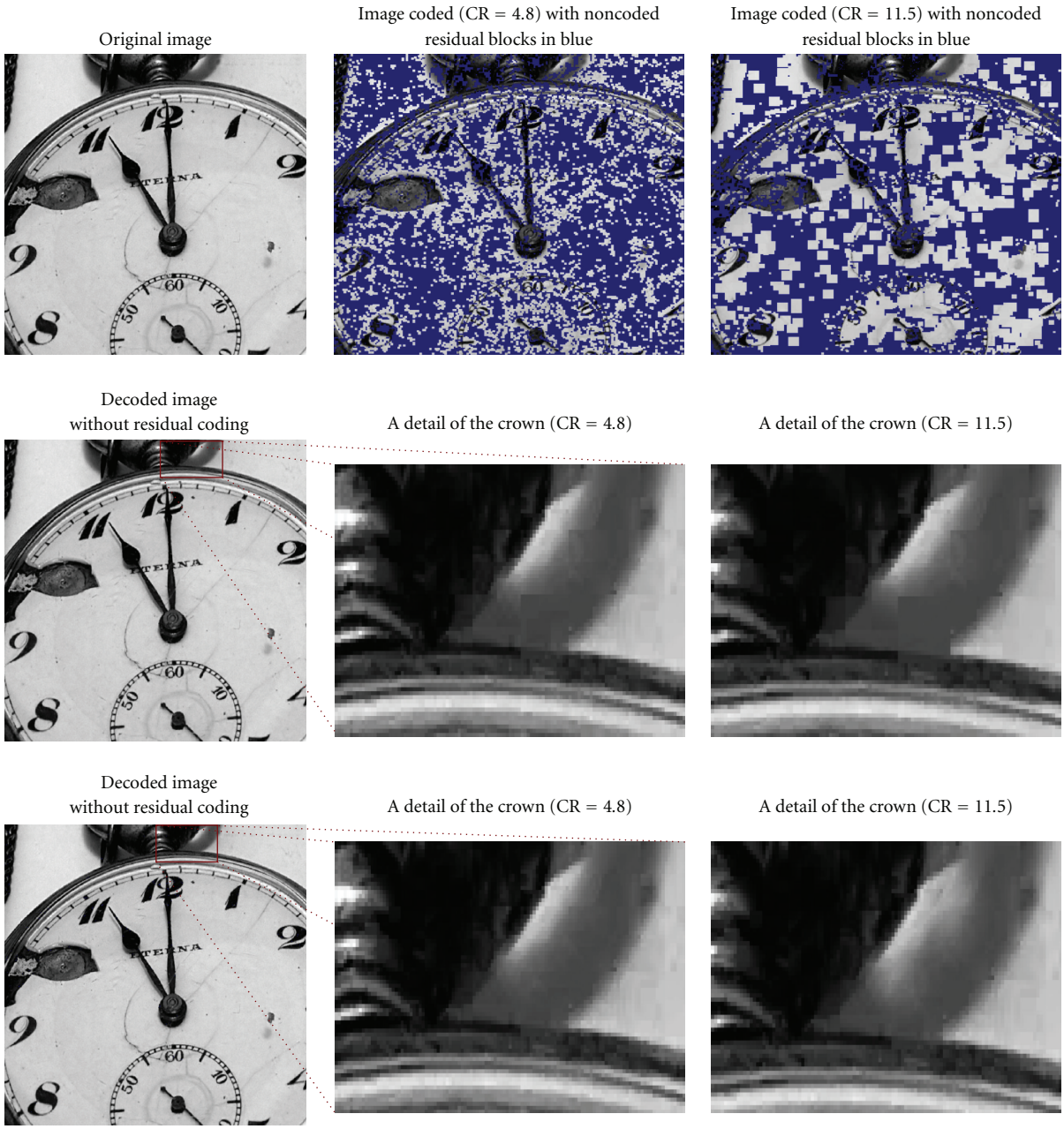

Figure 7: Residual coding for the image *clock*. On the first row: the original image, coded image at CR of 4.8 with noncoded residual blocks in blue, coded image at CR of 11.5 with noncoded residual blocks in blue. On the second row: coded image at a CR of 4.8 without residual coding, a detail of the crown, the same detail at a CR of 11.5. On the third row: coded image at a CR of 4.8 with residual coding, a detail of the crown, the same detail at a CR of 11.5.

The proposed hybrid approach has also been compared with a state-of-the-art classification scheme underlying that it can obtain higher speedup factors but at a comparable decoded image quality.

References

[1] C. He, S. X. Yang, and X. Huang, "Variance-based accelerating scheme for fractal image encoding," *Electronics Letters*, vol. 40, no. 2, pp. 115–116, 2004.

[2] C. M. Lai, K. M. Lam, and W. C. Siu, "Improved searching scheme for fractal image coding," *Electronics Letters*, vol. 38, no. 25, pp. 1653–1654, 2002.

[3] M. Polvere and M. Nappi, "Speed-up in fractal image coding: comparison of methods," *IEEE Transactions on Image Processing*, vol. 9, no. 6, pp. 1002–1009, 2000.

[4] C. Aggarwal, "On the effects of dimensionality reduction on high dimensional search," in *Proceedings of the ACMPODS Conference*, pp. 1–11, IBM Thomas J. Watson Research Center, Yorktown, NY, USA, 2001.

[5] B. E. Wohlberg and G. de Jager, "Fast image domain fractal compression by DCT domain block matching," *Electronics Letters*, vol. 31, no. 11, pp. 869–870, 1995.

[6] J. L. Wu and W. J. Duh, "Feature extraction capability of some discrete transforms," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 5, pp. 2649–2652, June 1991.

[7] D. J. Duh, J. H. Jeng, and S. Y. Chen, "DCT based simple classification scheme for fractal image compression," *Image and Vision Computing*, vol. 23, no. 13, pp. 1115–1121, 2005.

[8] J. M. Mas Ribés, B. Simon, and B. Macq, "Combined Kohonen neural networks and discrete cosine transform method for iterated transformation theory," *Signal Processing: Image Communication*, vol. 16, no. 7, pp. 643–656, 2001.

[9] Y. Zhao and B. Yuan, "A hybrid image compression scheme combining block-based fractal coding and DCT," *Signal Processing: Image Communication*, vol. 8, no. 2, pp. 73–78, 1996.

[10] T. Kim, R. E. Van Dyck, and D. J. Miller, "Hybrid fractal zerotree wavelet image coding," *Signal Processing: Image Communication*, vol. 17, no. 4, pp. 347–360, 2002.

[11] Y. Zhang and L. M. Po, "Speeding up fractal image encoding by wavelet-based block classification," *Electronics Letters*, vol. 32, no. 23, pp. 2140–2141, 1996.

[12] R. Distasi, M. Nappi, and D. Riccio, "A range/domain approximation error-based approach for fractal image compression," *IEEE Transactions on Image Processing*, vol. 15, no. 1, pp. 89–97, 2006.

[13] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[14] J. Kominek, "Waterloo BragZone and Fractals Repository," January 2007, http://links.uwaterloo.ca/Repository.html.

[15] Y. Iano, F. S. da Silva, and A. L. M. Cruz, "A fast and efficient hybrid fractal-wavelet image coder," *IEEE Transactions on Image Processing*, vol. 15, no. 1, pp. 98–105, 2006.