International Journal of Image and Graphics
Vol. 11, No. 4 (2011) 571–587

DOI: 10.1142/S0219467811004251

IMPROVED FRACTAL IMAGE COMPRESSION BASED ON ROBUST FEATURE DESCRIPTORS

WILLIAM ROBSON SCHWARTZ* and HELIO PEDRINI†

Institute of Computing, University of Campinas
Campinas, SP, Brazil, 13083-852
**schwartz@ic.unicamp.br*
†*helio@ic.unicamp.br*

Received 8 July 2010
Revised 8 August 2011
Accepted 15 August 2011

Fractal image compression is one of the most promising techniques for image compression due to advantages such as resolution independence and fast decompression. It exploits the fact that natural scenes present self-similarity to remove redundancy and obtain high compression rates with smaller quality degradation compared to traditional compression methods. The main drawback of fractal compression is its computationally intensive encoding process, due to the need for searching regions with high similarity in the image. Several approaches have been developed to reduce the computational cost to locate similar regions. In this work, we propose a method based on robust feature descriptors to speed up the encoding time. The use of robust features provides more discriminative and representative information for regions of the image. When the regions are better represented, the search for similar parts of the image can be reduced to focus only on the most likely matching candidates, which leads to reduction on the computational time. Our experimental results show that the use of robust feature descriptors reduces the encoding time while keeping high compression rates and reconstruction quality.

Keywords: Fractal compression; feature analysis; robust feature descriptors; histograms of oriented gradients; partitioned-based methods.

1. Introduction

The use of efficient and robust data compression methods has become of fundamental importance in nowadays communication systems due to the large amount of information being transmitted and stored. Fractal image compression is one of the most promising techniques for image compression due to its resolution independence, fast decompression, and its ability of obtaining high compression rates with smaller quality degradation compared to traditional compression methods.

Fractal compression exploits the fact that natural scenes present self-similarity, then a significant amount of redundancy can be removed, providing high compression rates. Since it is difficult to detect self-similarity in a global scale, the image

is usually partitioned into square blocks and affine transforms are used to describe the similarity between blocks. The resulting transforms, called *fractal codes*, are stored and used afterwards to reconstruct the original image by means of iterative function evaluations starting from an arbitrary image. By storing the fractal codes rather than bitmaps, images can be decompressed to resolutions that are higher or lower than the original resolution without distortions.

The use of block decomposition and affine transforms has provided significant results in fractal compression,[1,2] However, the encoding process is computational expensive due to the search performed to locate pairs of blocks presenting high similarity. Therefore, several approaches have been developed to reduce the computational cost to match blocks.[3–6] The most common approaches are *classification search methods*, *local search*, and *partitioned-based methods*. Classification search methods cluster blocks sharing similar properties and the search considers only blocks belonging to the same cluster. Local search assumes that similar blocks are located in nearly regions. Finally, quadtree-based methods consider multiple block sizes such that larger blocks are used to encode regions with few details and small blocks encode regions with fine details.

An important dpart of matching blocks presenting similar properties is the use of a representative description for the regions under consideration. A good feature descriptor spans a feature space that provides high separability between samples possessing different properties but clusters together similar samples. Recent works on object detection[7–9] and recognition[10,11] have achieved improved results due to the use of robust local feature descriptors for object representation. One of the most successful local feature descriptors is the histograms of oriented gradients (HOG).[12] HOG looks at the spatial distribution of edge orientations by capturing edge and gradient structures that characterize the local shape.

This work focuses specifically on speeding up the search for matching blocks by combining the quadtree-based approach with classification search approach using robust feature descriptors. In order to reduce the search space, a clustering algorithm is applied to group blocks sharing similar properties, these described by HOG features, a robust feature descriptor method widely employed in tasks such as object detection and recognition. Experimental results show that the use of robust feature descriptors, such as HOG, reduces the encoding time while keeping high compression rates and reconstruction quality.

This paper is organized as follows. In Sec. 2, a brief description of fractal compression is presented, as well as a review of relevant related work. Sec. 3 describes the proposed fractal image encoding method. Experimental results and conclusions are presented in Secs. 4 and 5, respectively.

2. Fractal Image Compression

Fractal image compression has become an important lossy technique due to its capacity of achieving high compression ratio and high reconstructed image quality.

The basic idea of fractal image compression is to exploit self-similarities present in the image to reduce the amount of redundancy. On the other hand, the main disadvantage of fractal encoding is its high computational cost due to the search used to find self-similarities.[1]

Barnsley[13] was the first to suggest the use of a set of transformations to compactly store images. The self-similarities contained in the images are represented by iterated function systems (IFS).[14] An iterated function system is a finite set of contraction mappings on a complete metric space.

Jacquin[15] proposed to partition the image into square blocks, search for regions or blocks of the images that are self-similar according to a certain criterion and, once a match is found, compute the transformations. A special type of IFS, called partitioned iterated function system (PIFS), is used to represent image blocks.

In fractal encoding, the image is divided into range and domain blocks. The smaller non-overlapping range blocks, R, cover the entire image, whereas larger domain blocks, D, are usually constructed from a subset of the original image. For each range block, the set of domain blocks is searched for the best match. The size of the domain block collection has a significant impact on the matching efficiency of each range block to its domain block and on the reconstructed image quality.

A contractive transformation is used to map each range block to its matched domain block. The transformation aims at minimizing a metric distortion measure during the search process. Similarity transform parameters between the range block and the corresponding domain block are used to encode the range block. The encoded image can be reconstructed by iterative evaluation of the transformations until converging to an approximation of the original image.

Several efforts have focused on reducing the complexity of the matching process during the fractal image encoding. The most relevant strategies are described in the following sections.

2.1. *Classification search methods*

Classification search methods group blocks sharing similar properties into clusters, such that the search for matches for a range block considers only domain blocks belonging to a single cluster.

Fisher[1] proposed a classification method that divides the set of domain blocks into 72 classes according to certain combinations of average and variance computed in four quadrants of the block under consideration. The reduction of the searching space is obtained while preserving reconstruction quality.

Caso *et al.*[16] replaced the fixed ordering of variances of an image block by a vector of variances. Such variance vectors are strongly quantized resulting in a collection of classes, where each class has a neighborhood of classes that can be searched. Tong and Pi[17] and Wu *et al.*[18] used the standard deviation to classify blocks. Duh *et al.*[19] and Wang *et al.*[20] used edge information of the blocks to classify them into a number of classes, resulting in a speed-up ratio of three to four times.

Lepsøy and Øien[21] presented a codebook clustering algorithm for generating the set of classes adaptively, that is, dependent on the target images. Boss and Jacobs[22] introduced an archetype classification based on a set of training images.

Kovács[5] proposed a fractal image encoding method using image block classification based on two parameters to sort blocks into disjoint classes. The direction of the approximate first derivative and a normalized root mean square error of the fitting plane in the block under consideration are used to reduce the number of domain blocks searched for a range block, without significant loss of reconstruction fidelity.

2.2. *Local search methods*

A straightforward fractal encoding method performs an exhaustive search[14] over all domain blocks in the image to find the best matching domain block for each range block. Since such exhaustive search algorithm for finding the optimal matching requires a high computational cost, extensive research on fast fractal image encoding algorithms has been carried out. Local search methods assume that the best matching domain block is located near the range block under consideration, that is, the search for similar domain block can be limited to the vicinity of the range block.[23,24]

Lee and Lee[25] described a method for searching limited by using the variance of the block. A speed-up ratio of 1 to 10 times is achieved, depending on the image being considered. Lai *et al.*[26] proposed a fractal image coding based on a single kick-out condition and zero contrast prediction to avoid a large number of range-domain block matches when finding the best matched domain block.

Bani-Eqbal[27] proposed a tree-based search method for binding the domain block pixels and arranging the domain blocks in a tree structure to guide the search. Chung and Hsu[3] described a strategy for partitioning the original image into a set of approximately homogeneous blocks prior to the application of the search algorithm, achieving a significant speed-up compared to the exhaustive search method.

2.3. *Partitioned-based methods*

Fractal encoding is usually implemented with square blocks. Other partitioning approaches have also been proposed to adapt the coding to local image characteristics to increase performance. A disadvantage of the fixed-size range block is the lack of adaptivity to the image. Rectangular nonuniform partition schemes include quadtree and horizontal-vertical partitioning. Koli and Ali[4] investigated the effect of variable sizes of range and domain blocks in the fractal image coding. They observed that the larger the size of domain blocks is, larger will be the compression ratio with more deterioration in reconstruction quality.

In a quadtree partition, a square range block in the image is split into four equal-sized blocks. This process is repeated starting from the entire image and continuing until the squares are small enough to find a good matching domain block.

Larger blocks can be used to encode regions with few details, whereas smaller blocks encode regions with fine details. Several image fractal encoding methods based on quadtree partitioning have been proposed in literature.[1,28]

Horizontal-vertical partitioning[29] recursively divides the image either horizontally or vertically to form two new rectangles. As in the quadtree scheme, the partitioning repeats until a metric quality is satisfied. Since the position of the partition is variable, horizontal-vertical partitioning is more flexible.

Triangular partitioning[30,31] initially divides a rectangular image into two triangles. Each of these triangles is recursively subdivided into four triangles by splitting the triangle along lines that join three partitioning points along the three sides of the triangle. Such scheme has several potential advantages since triangles can cover the image at any orientation.

2.4. *Other methods*

Hartenstein and Saupe[32] and Lee and Ra[33] developed fractal image methods that transform the image blocks into the frequency domain and exploit redundancy based on cross-correlation analysis. Chung and Hsu[3] described a two-phase prediction-and-subblock-based fractal encoding algorithm for partitioning the image into a set of approximately homogeneous blocks prior to the application of the search algorithm.

A fractal encoding algorithm based on the self-organizing map (SOM) of Kohonen for codebook clustering was presented by Bogdan and Meadows.[34] Hamzaoui[35] combined the SOM method for clustering with the nearest neighbor approach to group adaptive image classes.

Endo *et al.*,[36] Iano *et al.*,[37] and Bellouata[38] obtained good reconstructed image quality using hybrid wavelet-fractal methods based on Fisher's classification scheme.[1]

Saupe[39] developed a fractal image compression to quantize image blocks using a quadtree partitioning, followed by a nearest neighbor search method. Tong and Wong[40] presented a formulation of approximate nearest neighbor search based on orthogonal projection and pre-quantization of the fractal transform parameters. Quadtree partitioning is used to adjust the compression ratio.

Uhl and Hammerle[41] and Hufnagl and Uhl[42] presented a parallel implementation of fractal image encoding on MIMD and SIMD architectures, respectively. Lee *et al.*[43] described a parallel architecture for quadtree-based fractal image coding.

Saupe and Ruhl,[44] Mohamed and Aoued,[45] and Vences and Rudomin[46] developed evolutionary and genetic methods for speeding up the fractal image compression.

3. Proposed Method

In this work, we employ the classification search approach based on a quadtree decomposition. For each level of the quadtree decomposition, domain blocks are

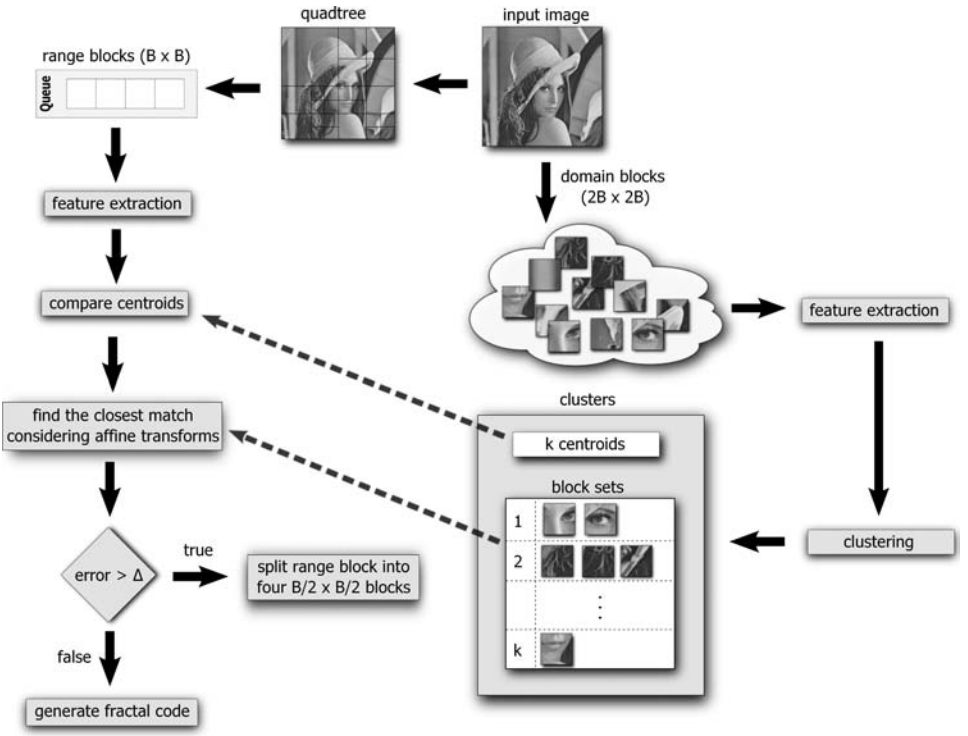


Fig. 1. Fractal image compression approach. For a given level of the quadtree decomposition where range blocks have $B \times B$ pixels are considered, domain blocks of $2B \times 2B$ pixels are sampled from the input image. Using feature vectors as descriptor for each block, a clustering algorithm is applied to partition the domain blocks in k clusters. After extracting the feature vector for a range block, the closest cluster, according to the distance between the cluster centroid and range block feature vector, is estimated, then only domain blocks belonging to the referred cluster are considered to find the best match. Once the best matching domain block is obtained, the reconstruction error is estimated. If it is smaller than the threshold, a fractal code is generated, otherwise, the range block is split into four sub-blocks of size $B/2 \times B/2$ pixels to be considered in the next level of the quadtree decomposition.

clustered and only those belonging to the closest cluster to a given range block are considered as candidates to a match. The procedure to encode an image using fractal compression is illustrated in Fig. 1 and described in details as follows.

As described in the previous section, the quadtree-based encoding method considers range blocks at different sizes to adapt the number of fractal codes to the amount of details contained in the image. Initially, the input image is partitioned into non-overlapping range blocks of size $B \times B$ pixels and possibly overlapping domain blocks with sizes of $2B \times 2B$ pixels. In order to obtain reconstruction for every region of the image, a matching domain block has to be obtained for each range block. If, among all domain blocks, the block with the best match provides a reconstruction error that is higher than a desirable error, the range block is further partitioned into four new blocks with $B/2 \times B/2$ pixels and set to be considered in

the next level of the quadtree decomposition (stored temporarily in a queue). The partitioning process finishes when a minimum block size is reached.

To achieve the goal of reducing the computational cost without degrading significantly the reconstruction quality, our approach is divided into two steps: *feature extraction* and *clustering*. The former is responsible for obtaining a better and more compact representation for blocks through the use of robust feature descriptors, while the latter reduces the set of domain block candidates to match a specific range block.

After sampling range and domain blocks (and rescaling the domain blocks to the same size of the range blocks), the feature extraction is performed using HOG descriptors, which captures edge or gradient structures that characterize the local shape.[12] To represent each block, a feature vector composed of 36 dimensions is extracted (HOG with four orientations and nine bins per histogram, normalized according to the L^2-norm). This vector is used to execute the clustering (in the case of domain blocks) and the matching (between domain and range blocks).

For each level of the quadtree decomposition, the K-means clustering algorithm is executed to partition the set of domain blocks into k clusters. Initially, all samples are standardized to have 0-mean and 1-standard deviation. After applying the K-means algorithm, the centroids for each cluster and its samples are stored to be used during the matching process.

Once the K-means has been executed for a given level of the quadtree decomposition, each range block, r_i, in the queue is compared to domain blocks aiming at finding the block with the smallest reconstruction error according to the root mean square error (RMSE)

$$e_i = \frac{1}{n}\sum_{j=1}^{n}[x_r(j) - (s_i x_d(j) + o_i)]^2, \tag{1}$$

where s_i is the contraction factor $0 < s_i < 1, o_i$ is an offset constant, $x_r(i)$ and $x_d(j)$ denote the pixel intensity for the range and domain blocks, respectively, and n is the number of pixels in the block.

To reduce the number of times that Eq. (1) is evaluated and parameters s_i and o_i need to be estimated, only a subset of the domain blocks is considered as candidates to match a range block r_i. Initially, we find the closest centroid, c_t, estimated by the K-means, to the feature vector extracted from the range block r_i; then, only domain blocks belonging to the cluster represented by c_t are considered as candidates to match block r_i. Therefore, if the feature vectors extracted from the blocks are well representative, the number of evaluations is reduced and only similar blocks will be considered as matching candidates, which speeds up the compression process. We show in the next section that when more clusters are considered (and, therefore, fewer evaluations are required), the quality of the reconstruction degrades lesser when HOG feature descriptors are used.

If the smallest reconstruction error, e_i, obtained by a pair of range and domain blocks is larger than a chosen threshold, the range block is split into four new range

blocks that will be considered in the next stage of the quadtree decomposition. Otherwise, if the error satisfies the threshold, the range block is encoded with the generation of a tuple, called *fractal code*, used in the reconstruction of the image. A fractal code for a range block r_i is composed of the tuple $(x_i, y_i, l_i, s_i, o_i)$, where (x_i, y_i) denotes the top-left corner of the best matching domain block and l_i has the decomposition level of the quadtree. More details regarding the fractal code and parameter estimation used in this work are given in the next section.

4. Experimental Results

In this section, we evaluate several aspects of our proposed approach. Initially, we describe the experimental setup used for this work. Then, we show compression results in a set composed of standard images and compare our method to related work published in literature.

4.1. *Experimental setup*

A fractal code for a range block r_i uses 16 bits to encode the location of the top-left corner of its matching domain block, 8 bits for the offset constant o_i, 2 bits to encode the range block size (saving only the decomposition level of the quadtree is enough to recover the range block location if they are stored in a sequential order), and 3 bits to store the contraction factor s_i. We limit the contraction factor to assume one of eight possible discrete values quantized uniformly from the interval $0 < s_i < 1$. Therefore, 29 bits are used for each fractal code.

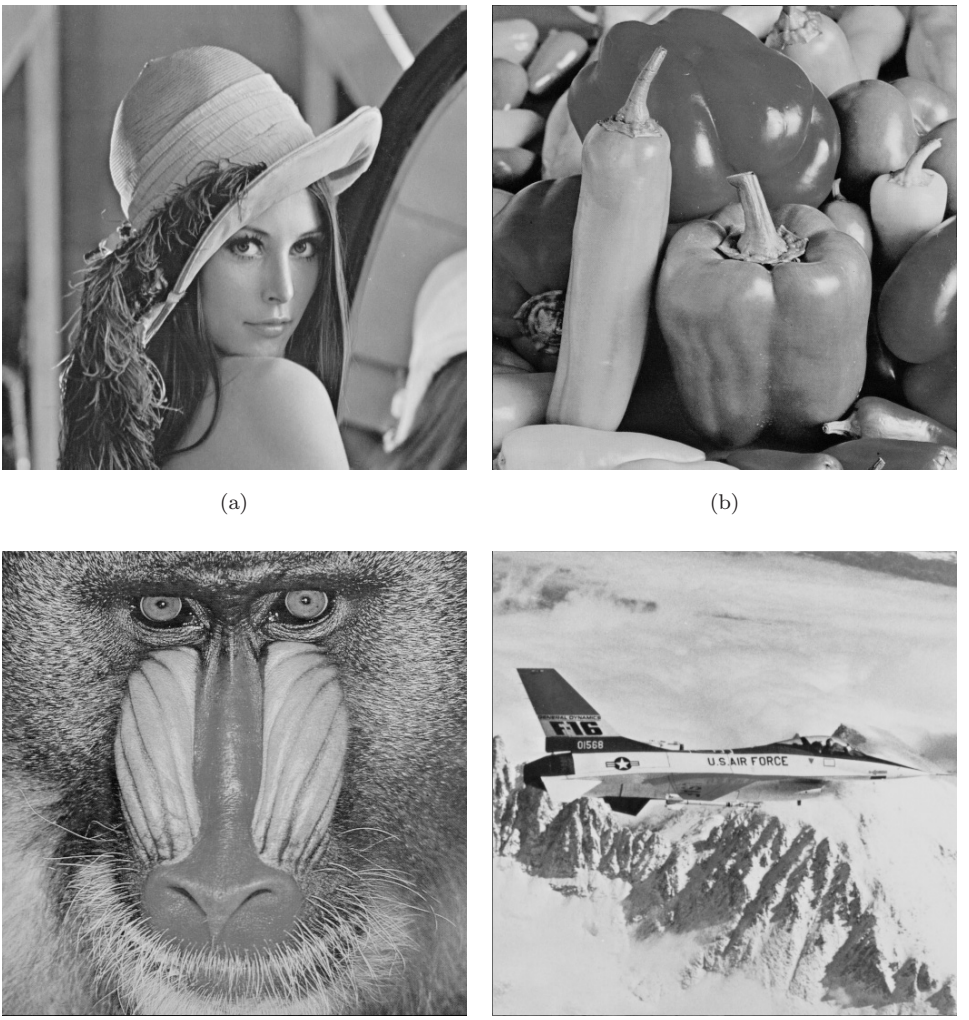
To avoid evaluating the reconstruction error (Eq. (1)) for multiple values of parameter s_i to find a matching between a pair (r_i, d_j) of range and domain blocks, we apply the least squares regression method to estimate the best values of s_i and o_i. The closest value among the eight possible is used as s_i. Only after this estimation, the reconstruction error is evaluated for the pair of blocks r_i and d_j.

For the quadtree, we use four levels of decomposition considering blocks of 32×32, 16×16, 8×8, and 4×4 pixels. The threshold for the reconstruction error at a decomposition level l is estimated by

$$th_l = v_0 + l\rho, \tag{2}$$

where v_0 is an initial threshold and ρ is the increment using at each level.

The baseline method used in our experiments considers simply the quadtree decomposition without performing the clustering for the domain blocks. By using this baseline, we are able to evaluate the speed-up obtained, how much the reconstruction quality degrades when fewer comparisons are performed, and the importance of the robust feature descriptors to reduce the reconstruction quality degradation. Four feature descriptors are used: HOG, mean, standard deviation, and entropy (the latter three are commonly used in fractal compression). Our experiments consider a standard set of testing images, shown in Fig. 2. All images are grayscale and have 512×512 pixels.



(a) (b)

(c) (d)

Fig. 2. Images used to evaluate the proposed image compression approach: (a) Lena, (b) peppers, (c) Baboon, and (d) f16.

All experiments were conducted on an Intel Core i7-860 processor, 2.8 GHz with 4 GB of RAM running Windows 7 operating system using a single processor core. The method was implemented using C++ programming language.

4.2. *Number of clusters*

We now evaluate the number of clusters used to partition the domain blocks into subsets. Figure 3 shows that while the number of bits per pixel (bpp) increases with the number of clusters, the peak signal-to-noise ratio (PSNR) reduces. Therefore,

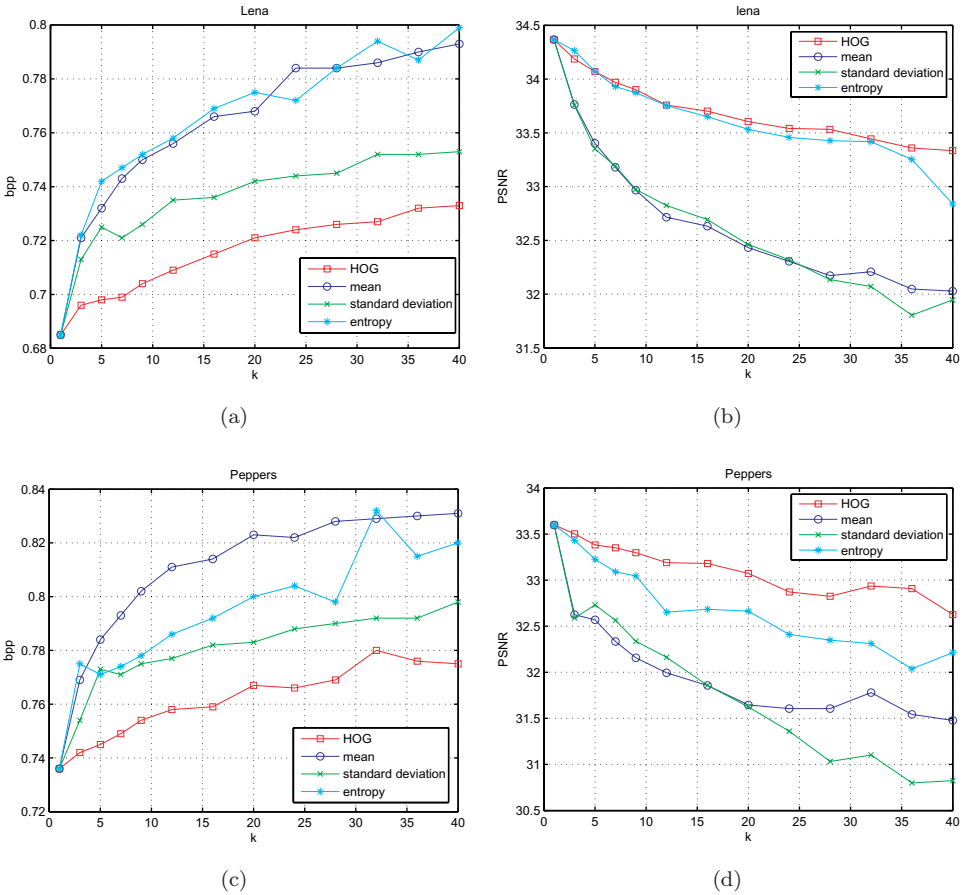
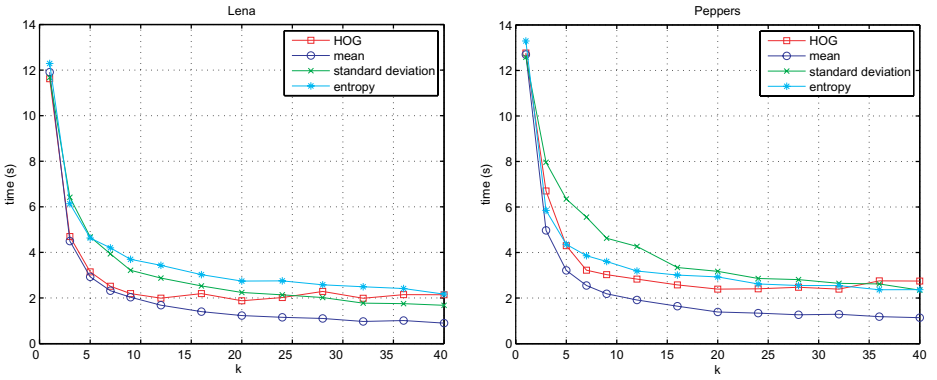


Fig. 3. Reconstruction quality (measured by bpp and PSNR) as a function of the number of clusters used by the K-means algorithm: (a) Lena, (b) Lena, (c) peppers, and (d) peppers.

the reconstruction quality degrades with the number of clusters. On the other hand, according to Fig. 4, the encoding time reduces significantly when the number of clusters increases. Furthermore, it is also possible to see that the compression time does not decrease significantly after $k = 20$ (due to overhead incurred by the K-means algorithm). Therefore, to obtain a fair reconstruction quality, we have used $k = 20$ for all the remaining experiments.

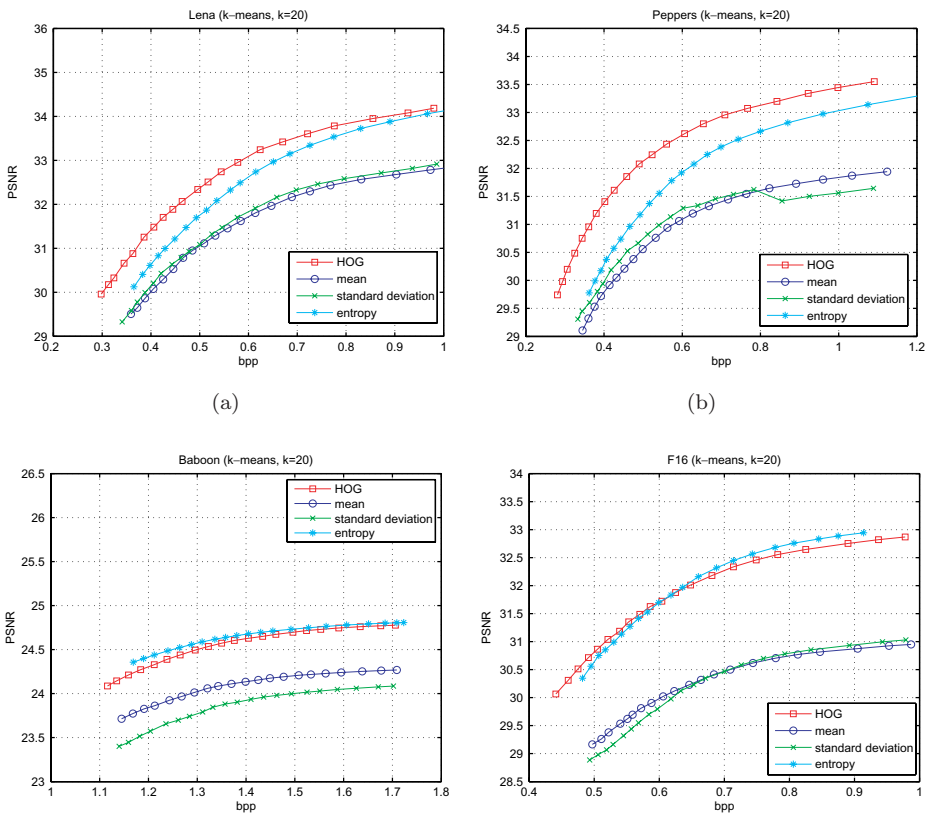
4.3. *Feature descriptor evaluation*

Figure 5 compares the reconstruction quality obtained with different feature descriptors. It is possible to see that, in general, HOG is the feature descriptor that provides better trade-off between bpp and PSNR, mainly when considered that its encoding time is lower than most of the other feature descriptors, as shown in Fig. 4. Therefore, the use of robust feature descriptors improves reconstruction



(a) (b)

Fig. 4. Computational time as a function of the number of clusters used by the K-means algorithm: (a) Lena, and (b) peppers.



(a) (b)

(c) (d)

Fig. 5. Tradeoff between PSNR and bpp for different feature descriptors: (a) Lena, (b) peppers, (c) Baboon, and (d) f16.

(a) (b)

(c) (d)

Fig. 6. Reconstructed Lena images compressed using different feature descriptors at a compression ratio of 0.4 bpp: (a) HOG (PSNR = 31.48 dB), (b) mean (PSNR = 30.07 dB), (c) standard deviation (PSNR = 30.20 dB), and (d) entropy (PSNR = 30.61 dB).

quality and reduces computational cost. In addition, Fig. 6 shows reconstructed images compressed with different feature descriptors at the same compression ratio. The compression based on HOG presents higher reconstruction quality.

4.4. *Baseline evaluation*

Using the HOG feature descriptors and $k = 20$ for the K-means, Fig. 7 compares the tradeoff between bpp and PSNR and the encoding time for each image. Although

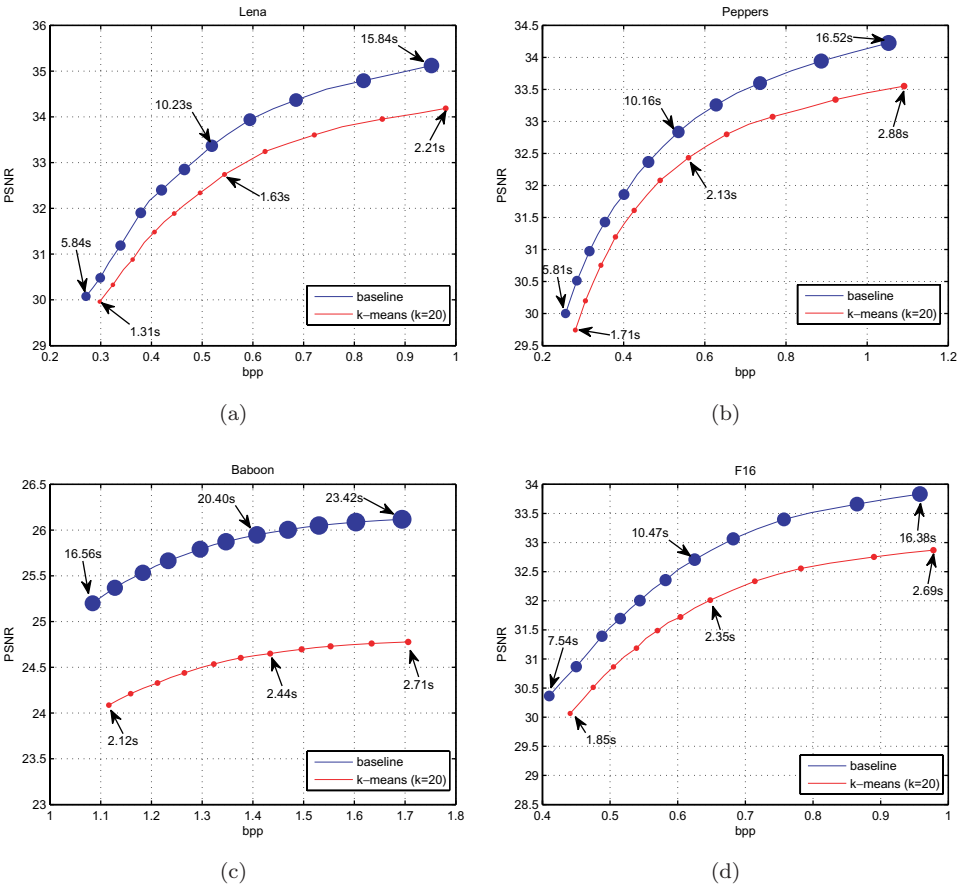


Fig. 7. Comparison between the clustering-based approach to the baseline method based only on quadtree decomposition. The area of the circles is proportional to the encoding time at each particular tradeoff bpp vs. PSNR: (a) Lena, (b) peppers, (c) Baboon, and (d) f16.

the encoding time has been reduced between 5 and 10 times when the clustering is considered, the reconstruction quality does not present significant degradation for a given bpp (PSNR is reduced by approximately 1 dB). Therefore, according to the results, the application of a clustering algorithm associated with robust feature descriptors provides further speed-up to the quadtree-based fractal compression approach.

4.5. *Comparisons*

Figure 8 compares our method to the approach proposed by Kovacs[5] using image Lena with size 512×512 pixels. His approach obtained significant reduction on computational cost compared to several fractal compression methods, including the quadtree-based method proposed by Fisher.[1] The results show that our method

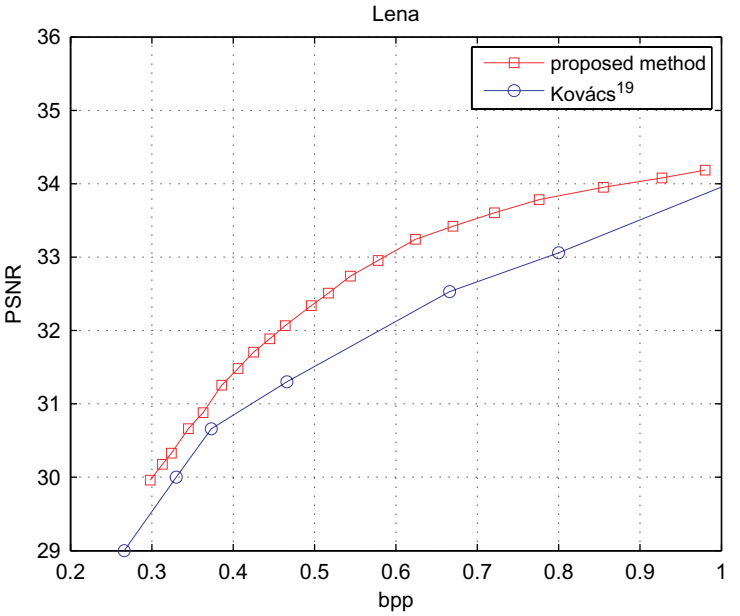


Fig. 8. Comparison showing reconstruction image quality versus compression ratio.

provides higher reconstruction quality at the same compression ratio. To follow the experiment designed by Kovacs,[5] the computational time for both methods is restricted to at most five seconds. In our case, however, the computational time is significantly smaller than 5 s, as shown in Fig. 7(a), which takes up to 2.21 s.

5. Conclusions

We have proposed an encoding method for reducing computational cost to find matching blocks for range blocks. The combination of a partition-based method using quadtree decomposition with robust feature descriptors allowed the search to consider only blocks sharing similar properties, providing significant speed-up. Experimental results have demonstrated that feature descriptors with higher discriminative power, such as HOG, provide higher compression rates and reconstruction quality compared to another published fractal encoding method.

Acknowledgments

The authors are thankful to FAPESP, CNPq, and CAPES for financial support. This research was partially supported by FAPESP grant 2010/10618-3.

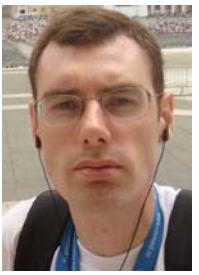
References

1. Y. Fisher, *Fractal Image Compression — Theory and Application* (Springer-Verlag, New York, 1995).

2. N. Koli and M. Ali, "A survey on fractal image compression key issues," *Information Technology Journal* **7**(8), 1085–1095 (2008).
3. K. C. C. Hsu, "Novel prediction- and subblock-based algorithm for fractal image compression," *Chaos, Solitons & Fractals* **29**(1), 215–222 (2006).
4. N. Koli and M. Ali, Lossy color image compression technique using fractal coding with different size of range and domain blocks, in *International Conference on Advanced Computing and Communications*, pp. 236–239 (Surathkal, India, December 2006).
5. T. Kovács, "A fast classification based method for fractal image encoding," *Image Vision Computing* **26**(8), 1129–1136 (2008).
6. V. de Lima, W. R. Schwartz and H. Pedrini, Fast low bit-rate 3D searchless fractal video encoding, in *Conference on Graphics, Patterns and Images* (Maceio, AL, Brazil, 2011).
7. S. Maji, A. Berg and J. Malik, Classification using intersection kernel support vector machines is efficient, in *IEEE Conference on Computer Vision and Pattern Recognition* (Anchorage, AK, USA, 2008).
8. W. R. Schwartz, A. Kembhavi, D. Harwood and L. S. Davis, Human detection using partial least squares analysis, in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 24–31 (Kyoto, Japan, 2009).
9. B. Wu and R. Nevatia, Optimizing discrimination-efficiency tradeoff in integrating heterogeneous local features for object detection, in *IEEE Conference on Computer Vision and Pattern Recognition* (Anchorage, AK, USA, 2008).
10. W. Schwartz and L. Davis, Learning discriminative appearance-based models using partial least squares, in *XXII Brazilian Symposium on Computer Graphics and Image Processing*, pp. 322–329 (Rio de Janeiro, RJ, Brazil, 2009).
11. X. Tan and B. Triggs, "Fusing gabor and LBP feature sets for kernel-based face recognition," *Analysis and Modeling of Faces and Gestures*, Springer Berlin/Heidelberg **78**, 235–249 (2007).
12. N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, in *International Conference on Computer Vision and Pattern Recognition*, pp. 886–893 (June 2005).
13. M. F. Barnsley, *Fractals Everywhere* (Academic Press, Boston, MA, USA, 1998).
14. M. F. Barnsley and L. P. Hurd, *Fractal Image Compression* (A. K. Peters, Natick, MA, USA, 1993).
15. A. Jaccquin, Image coding based on a fractal theory of iterated contractive image transformations, *IEEE Transactions on Image Processing* **1**, 18–30 (1992).
16. G. Caso, P. Obrador and C.-C. Kuo, Fast methods for fractal image encoding, in *SPIE Visual Communication and Image Processing*, Vol. 2501, pp. 583–594 (Taipei, Taiwan, May 1995).
17. C. Tong and M. Pi, Fast fractal image encoding based on adaptive search, *IEEE Transactions on Image Processing* **10**(9), 1269–1277 (2001).
18. X. Wu, D. Jackson and H. Chen, "A fast fractal image encoding method based on intelligent search of standard deviation," *Computers and Electrical Engineering* **31**(6), 402–421 (2005).
19. D. Duh and J. J. S. Chen, "DTC based simple classification scheme for fractal image compression," *Image Vision Computing* **23**, 1115–1121 (2005).
20. Z. Wang, D. Zhang and Y. Yu, "Hybrid image coding based on partial fractal mapping," *Signal Processing: Image Communication* **15**(9), 767–779 (2000).
21. S. Lepsøy and G. Øien, Fast attractor image encoding by adaptive codebook clustering, in *Fractal Image Compression: Theory and Application*, pp. 177–197 (Springer-Verlag, London, UK, 1995).

22. R. Boss and E. Jacobs, Archetype classification in an iterated transformation image compression algorithm, *Fractal Image Compression: Theory and Application*, pp. 79–90 (Springer-Verlag, London, UK, 1995).
23. D. Monro and F. Dudbridge, Fractal approximation of image blocks, in *IEEE International Conference on Acoustics, Speech, Signal Processing*, Vol. 3, pp. 4585–4588 (San Francisco, CA, USA, March 1992).
24. T. Truong, C. Kung, J. Jeng and M. Hsieh, "Fast fractal image compression using spatial correlation," *Chaos, Solitons & Fractals* **22**(5), 1071–1076 (2004).
25. C. Lee and W. Lee, "Fast fractal image block coding based on local variances," *IEEE Transactions on Image Processing* **7**(6), 888–891 (1998).
26. C. Lai, K. Lam and W. Siu, "A fast fractal image coding based on kick-out and zero contrast conditions," *IEEE Transactions on Image Processing* **12**(11), 1398–1403 (2003).
27. B. Bani-Eqbal, "Enhancing the speed of fractal image compression," *Optical Engineering* **34**(6), 1705–1710 (1995).
28. D. Saupe and S. Jacobs, "Variance-based quadtrees in fractal image compression," *Electronic Letters* **33**(1), 46–48 (1997).
29. Y. Fisher and S. Menlove, Fractal encoding with HV partitions, in *Fractal Image Compression: Theory and Application*, pp. 119–136 (Springer-Verlag, London, UK, 1995).
30. F. Davoine, M. Antonini, J. Chassey and M. Barlaud, "Fractal image compression based on delaunay triangulation and vector quantization," *IEEE Transactions on Image Processing* **5**(2), 338–346 (1996).
31. D. Hebert and E. Soundararajan, Fast fractal image compression with triangular multiresolution block matching, in *International Conference on Image Processing*, pp. 747–750 (Chicago, IL, USA, 1998).
32. H. Hartenstein and D. Saupe, "Lossless acceleration of fractal image encoding via the fast fourier transform," *Signal Processing: Image Communication* **16**(4), 383–394 (2000).
33. S. Lee and S. Ra, "An analysis of isometry transforms in frequency domain for the fast fractal coding," *IEEE Signal Processing Letters* **6**(5), 100–102 (1999).
34. A. Bogdan and H. Meadows, "Kohonen neural network for image coding based on iteration transformation theory," *SPIE Neural and Stochastic Methods in Image and Signal Processing* **1766**, 425–436 (1992).
35. R. Hamzaoui, "Codebook clustering by self-organizing maps for fractal image compression," *Fractals* **5**, 27–38 (1997).
36. D. Endo, T. Hiyane, K. Atsuta and S. Kondo, Fractal image compression by the classification in the wavelet transform domain, in *International Conference on Image Processing*, Vol. 981, pp. 788–792 (Chicago, IL, USA, 1998).
37. Y. Iano, F. Silva and A. Cruz, "A fast and efficient hybrid fractal-wavelet image coder," *IEEE Transactions on Image Processing* **15**(1), 98–105 (2006).
38. K. Bellouata, "Fast fractal coding of subbands using a non-iterative block clustering," *Journal of Visual Communication and Image Representation* **16**(1), 55–67 (2005).
39. D. Saupe, Fractal image compression via nearest neighbor search, in *NATO ASI Conference on Fractal Image Encoding and Analysis*, pp. 95–108 (Trondheim, Norway, July 1996).
40. C. Tong and M. Wong, "Adaptive approximate nearest neighbor search for fractal image compression," *IEEE Transactions on Image Processing* **11**(6), 605–615 (1992).
41. A. Uhl and J. Hammerle, "Fractal image compression on MIMD architectures I: Basic algorithms," *Parallel Algorithms Applications* **11**, 187–204 (1997).

42. C. Hufnagl and A. Uhl, "Algorithms for fractal image compression on massively parallel SIMD arrays," *Real-Time Imaging* **6**(4), 267–281 (2000).
43. S. Lee, S. Omachi and A. Hirotomo, A parallel architecture for quadtree-based fractal image coding, in *International Conference on Parallel Processing*, pp. 15–22 (Washington, DC, USA, August 2000).
44. D. Saupe and M. Ruhl, Evolutionary fractal image compression, in *IEEE International Conference on Image Processing*, pp. 129–132 (Lausanne, Switzerland, September 1996).
45. F. Mohamed and B. Aoued, "Speeding up fractal image compression by genetic algorithms," *Multidimensional Systems and Signal Processing* **16**(2), 217–236 (2005).
46. L. Vences and I. Rudomin, Genetic algorithms for fractal image and image sequence compression, in *Computation Visual*, pp. 35–44 (Monterrey, Mexico, 1997).



William Robson Schwartz received his Ph.D. degree in computer science from University of Maryland, College Park, USA. He received his B.Sc. and M.Sc. degrees in computer science from the Federal University of Parana, Curitiba, Brazil. He is currently a postdoctoral researcher in the Institute of Computing at the University of Campinas, Brazil. His research interests include computer vision, pattern recognition, and image processing.

Helio Pedrini received his Ph.D. degree in electrical and computer engineering from Rensselaer Polytechnic Institute, Troy, NY, USA. He received his M.Sc. degree in electrical engineering and his B.Sc. in computer science, both from the University of Campinas, Brazil. He is currently a professor in the Institute of Computing at the University of Campinas, Brazil. His research interests include image processing, computer vision, pattern recognition, computer graphics, and computational geometry.