Fractal Image Encoding Using a Constant Size Domain Pool

Vitor de Lima[1] William Robson Schwartz[1] Hélio Pedrini[1]

[1]University of Campinas
Institute of Computing
Campinas-SP, Brazil, 13084-971

Abstract

Image compression techniques play an important role in data reduction and transmission. This work proposes a fractal image encoder based on a small domain pool with constant size constructed from the neighborhood of each range block and an efficient spatial subdivision data structure. This method is compared to a searchless algorithm using well-known grayscale images. The proposed approach is capable of performing fast compression and decompression, while maintaining high visual fidelity and operating at low bit-rates.

1. Introduction

Most fractal image encoders are characterized by extremely slow encoding times (taking a few hours to encode a single image) and fast decompression [5]. This is caused by the fact that constructing a fractal to approximate the content of an image is much more complex than displaying it [10]. Unfortunately, they are not competitive with other encoders based on transform coding both in encoding speed and rate-distortion performance.

A fractal encoder outputs a transform called *collage* that, given a downsampled version of the original image, it can create an approximation of the signal in its original scale. The usual method to create this is based on numerous searches for similar blocks between the image in these two scales which can be accelerated by specially designed heuristics [5].

A fast method for constructing a collage, called *searchless fractal compression*, was proposed by Furao and Hasegawa [3]. Each block in the original scale (called *range block*) is associated to a single block in the higher scale (called *domain block*). If the similarity between these two blocks is acceptable, the range block is encoded, otherwise, it is partitioned and the process is applied to the resulting sub-blocks. The encoders based on this concept are faster by orders of magnitude when compared to other fractal encoders employing more complex fractal-based algorithms, taking less than one second to encode an image instead of hours.

This work proposes a fast method for encoding images that is not as restrictive as the searchless encoders while keeping the encoding speed approximately at the same order of magnitude and improving the image quality at similar bit rates.

The next sections are organized as follows. Section 2 presents a brief review of fractal coding and searchless methods. The proposed method is described in Section 3. Experimental results comparing the proposed method against a searchless encoder are presented in Section 4. Finally, the conclusions of the work are given in Section 5.

2. Background

This section presents a brief review of fractal image coding and the searchless fractal encoders.

2.1. Fractal Coding

Fractal image encoders [5] generate a transform as output, called *collage*, that approximates the original image given a downsampled version of it. This transform is created in such way that when applied several times to any image, it will converge to an approximation of the original signal. Based on this property, starting with an arbitrary image, the decoder only needs to apply the collage and downsample its results repeatedly until it converges to the desired output.

Most fractal encoders construct the collage through a subdivision of the image into blocks (range blocks) that are matched against higher scale blocks (domain blocks) in the downsampled image by means of costly searching algorithms. This process makes the encoding process extremely expensive, specially when compared to the decoding time.

The collage can apply any affine transform to map a domain block into a range one. The collage also changes the

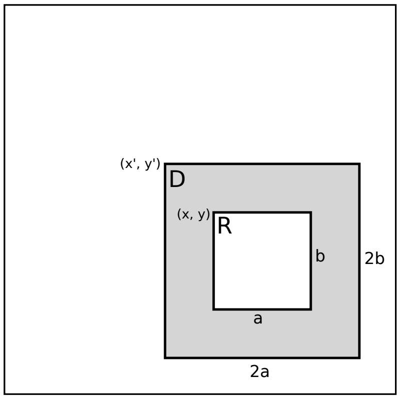


Figure 1. Relationship between range (R) and domain (D) blocks in a searchless fractal encoder.

intensities of the pixels contained in the domain blocks by using a transform such as the one proposed by Øien and Lepsøy [7], shown in Equation 1, where G is the resulting pixel intensity, D is the intensity in the domain block, $\bar{D}$ is the mean intensity of the pixels in the domain block, $\bar{r}$ is the mean intensity of the pixels in the range block and α is a parameter determined by a least squares estimation.

$$G(D) = \alpha(D - \bar{D}) + \bar{r} \tag{1}$$

2.2. Searchless Fractal Coding

Initially proposed by Furao and Hasegawa [3], the searchless fractal coding aims at quickly constructing a collage avoiding the complex search for the best match between range and domain blocks. Each range block at the coordinates (x, y) with dimensions $a \times b$ is matched against only one domain block with dimensions $2a \times 2b$ located at $(x - a/2, y - b/2)$, as shown in Figure 1. To encode a range block, only α and $\bar{r}$ must be transmitted, instead of the large number of parameters used in other state-of-the-art fractal encoders.

The image is partitioned into equally sized blocks using a quadtree, which are encoded by the process previously described. For each block. if the obtained error is larger than a certain threshold, the block is recursively divided into four sub-blocks until a certain minimum size is reached.

The searchless encoder proposed by Furao and Hasegawa [3] was later refined in [16] by employing a more flexible data structure to subdivide the image, in which a range block can be split into half either in the horizontal or vertical directions. There is no limit to the size of the range blocks, but smaller regions always have their α parameter set to zero and $\bar{r}$ is encoded using fewer bits.

3. Proposed Method

This paper proposes a new method for encoding the relationship between the domain and range blocks in images that is not as restrictive as the searchless methods, but is not as complex as the classical brute force or heuristic algorithms [1].

A more flexible domain pool with a larger selection of possible domain blocks is created to avoid the necessity of the searchless encoders to split certain regions of the image several times until they reach the desired reconstruction quality. This selection avoids such excessive use of subdivisions and also can reduce the number of bits used to encode the image.

For each range block with dimensions $a \times b$ located at (x, y), there are 9 possible domain blocks that can be used to represent it. The position of these blocks can be expressed as

$$\begin{aligned} x' &= x - a + p_x \times a/2 \\ y' &= y - b + p_y \times b/2 \end{aligned} \tag{2}$$

where p_x and p_y must be equal to 0, 1 or 2. All possible domain blocks given by this equation are illustrated in Figure 2. These candidate domain blocks have dimensions equal to $2a \times 2b$ and only one of them is chosen as the definitive mapping by testing which one of them has the lowest error when compared to the original range block, after having its gray values properly transformed by Equation 1.

If the range block must be split to achieve a lower reconstruction error, then it is divided into two equally sized sub-blocks. However, to do so, it must decide in which direction this subdivision will be performed. The heuristic used to choose how to properly partition a block tries to encode both sub-blocks separately without actually subdividing them, but by evaluating the two possible directions that can be used to split the original block. The direction resulting in the lowest estimated error is chosen.

The encoder initially subdivides the image into a uniform grid of blocks with 64×64 pixels. The rate-distortion heuristic proposed by Saupe et al. [12] is used for rate control, the blocks are inserted into a priority queue to sort the range blocks according to their sum of squared differences (SSD) instead of the mean squared difference (MSE), this replacement was proposed by Fisher and Menlove [2] to improve both the PSNR and the perceptual quality of the resulting image. At each iteration of the decoder, the range block with the largest SSD is subdivided into two blocks, which are reinserted into the queue. The total number of iterations, N_{it}, is defined by the user.

After the last iteration, the range blocks in the queue are transmitted by a context-adaptive arithmetic encoder [11], where α parameters are quantized into one of the values in the set $\{0.25, 0.5, 0.75, 1.0\}$, as suggested by Tong and

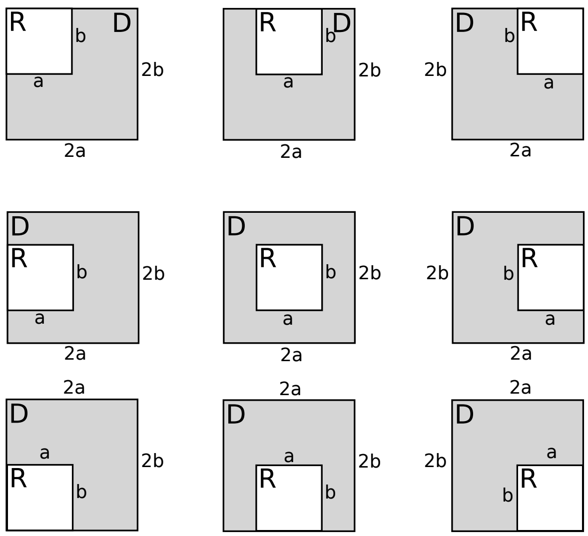


Figure 2. All possible relationships between the range (R) and the domain blocks (D) in the proposed encoder.

Pi [15], and encoded by an adaptive context chosen by the $\lfloor \log_2 A \rfloor$, where A is total area of the block, p_x and p_y values are encoded by their own adaptive contexts. Range blocks with only one or two pixels have their α parameter set to zero as suggested by Wu et al. [16].

The parameter $\bar{r}$ is encoded by a prediction method similar to the one proposed by Teuhola [14], in which the range blocks are scanned recursively from the left to the right and from top to bottom. Each visited range block predicts its mean value by averaging four mean values from neighboring blocks (as shown in Figure 3), quantizes the prediction and the real mean value using a uniform quantizer and an adaptive context chosen by the $\lfloor \log_2 A \rfloor$, as shown in Table 1 to encode the difference between them.

The spatial subdivision binary tree is transmitted by two different symbols: a binary flag for each node to mark it as either a leaf or an internal node, which is encoded by an adaptive context based on the $\lfloor \log_2 A \rfloor$ and another symbol to indicate the direction that node was split, which has two different contexts, one for blocks with b larger than a and another one for the remaining blocks.

The decoding process is accelerated by 3 different and complementary methods. The initial image that is used in the first iteration of the decoder is composed by filling each range block with its mean (for more details, see [6]). The used pixel intensity transform is the one proposed by Øien

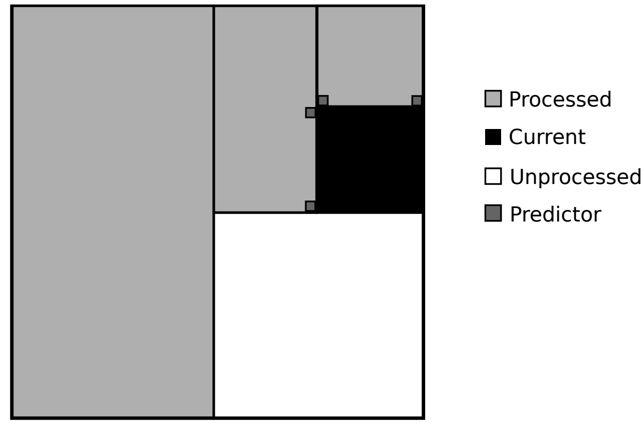


Figure 3. The prediction rule used to encode the mean values.

Decision Table for the Quantization of $\bar{r}$		
area	quantization step	number of used bits
1	16	4
2	16	4
4	8	5
8	8	5
16	4	6
32	4	6
64	2	7
≥ 128	1	8

Table 1. Quantizers applied according to the area of the range block.

and Lepsøy [7] with additional proofs and details given by Pi et al. [9]. Each iteration is applied according to the Gauss-Seidel inspired method proposed by Hamzaoui [4], which uses only one image during the iterations overwriting each range block with its updated contents. The use of these methods assures that the decoding process converges in 4 iterations or less, instead of the usual 8 to 10 iterations used by other fractal decoders.

Finally, the image is post-processed by the same deblocking filter proposed by Fisher and Menlove [2] that adapts itself according to the size of the range block, ignoring the ones with smaller areas and using more aggressive parameters in the larger ones.

Images	Searchless		Constant-Sized Domain Pool	
	encoding time (ms)	decoding time (ms)	encoding time (ms)	decoding time (ms)
Baboon	21.9	19.2	100.6	19.3
Barbara	21.0	19.0	92.3	19.2
Boat	20.3	18.8	94.5	19.2
Goldhill	21.6	19.0	98.0	19.2
Lena	21.6	19.2	99.3	19.5
Peppers	21.4	19.0	99.3	19.3

Table 2. Average encoding and decoding time of the proposed and the searchless method for the benchmark images.

4. Experimental Results

All experiments were conducted on an Intel Core 2 Duo E6750 processor, 2.66 GHz with 3GB of RAM running the Linux operating system. The method was implemented in C++ programming language using only integer values and running on a single thread.

The proposed approach is compared to a searchless encoder using the standard grayscale benchmark images *Baboon*, *Barbara*, *Boat*, *Goldhill*, *Lena*, and *Peppers*, with 512×512 pixels. The metric used to compare the original and the decoded images is the peak signal-to-noise ratio (PSNR), which can be calculated as:

$$\text{PSNR} = 10 \log_{10} \left(\frac{255^2}{\text{MSE}} \right) \tag{3}$$

where MSE is the mean squared error between the compared images. The resulting PSNR for each image using both methods at different bit-rates is shown in Figure 4.

The searchless encoder used in the comparisons is based on exactly the same base code from the constant size domain pool encoder, but assigning a fixed value for p_x and p_y of each range block, it is extremely similar, but not exactly equal to the one proposed by Wu et al. [16]. This allows a fair comparison between the two proposed domain-range mappings, avoiding possible differences in the implementation.

Both encoders were compared by varying N_{it} from 100 to 10000 iterations. The rate-distortion curves show that the proposed method maintains a lower distortion in the compressed images at the same rates even though there are fewer range blocks in the transmitted data, since each block uses a larger amount of bits.

Table 2 presents the average encoding and decoding times for each image. Although the proposed domain pool is nine times larger if compared to the searchless method, the total encoding time is only increased by a factor of five since there are fewer range blocks. The number of range blocks, as shown in Table 3, is smaller because they are subdivided less frequently and consequently they cover larger areas.

Images	Number of Range Blocks	
	searchless	proposed method
Baboon	6005	4223
Barbara	5807	4025
Boat	5708	4025
Goldhill	6005	4223
Lena	5708	4025
Peppers	5708	4124

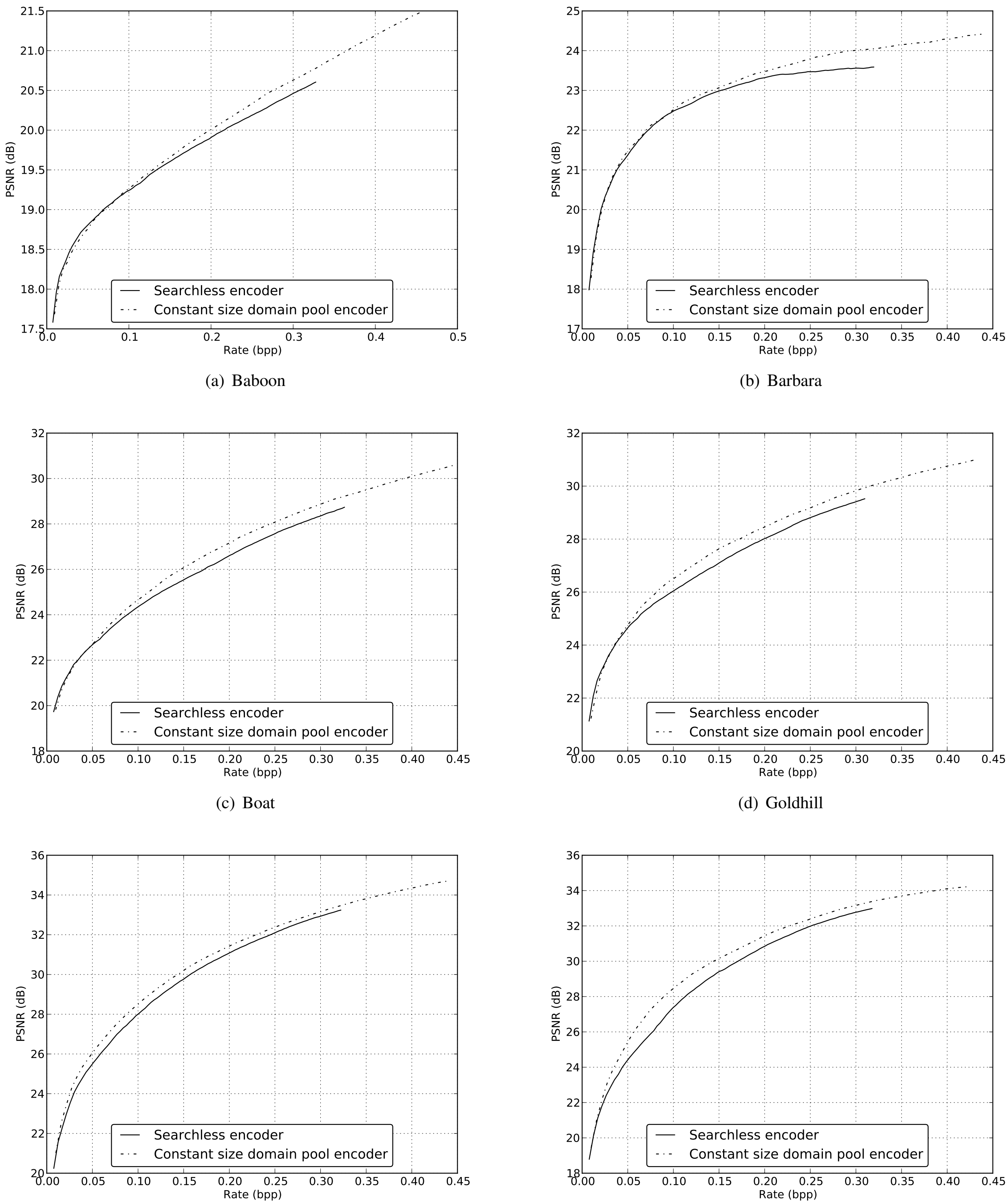
Table 3. Number of range blocks generated at 0.20 bpp for each tested image.

5. Conclusions and Future Work

This paper proposed a novel domain pool for fractal coding which is an intermediary between the searchless encoders and the usual methods employing brute force or heuristics in a large domain pool.

Due to the smaller number of encoded range blocks, the proposed method outperforms the searchless encoder even though the latter uses less bits to encode each block. The additional complexity of the larger domain pool has a significant performance impact, however, the encoder still takes approximately 100 ms to encode a 512×512 image compared to several minutes of the original fractal encoders based on large domain pools.

It is important to note that p_x and p_y parameters are transmitted without any form of optimized compression and



(a) Baboon (b) Barbara

(c) Boat (d) Goldhill

(e) Lena (f) Peppers

Figure 4. Rate-distortion curve for a number of tested images.

the domain block is chosen by exhaustive search, which causes a major decrease in the rate-distortion performance and an increase in the encoding time of the proposed approach. Future experiments will analyze possible heuristics to choose the proper domain block and compress its relative position.

Other proposals for future work include the use of a pyramidal algorithm during the image decoding [13], a more efficient method for the quantization and coding of $\bar{r}$ parameter, improvements on the heuristics and data structures used in spatial subdivision, and the use of rate-distortion optimization methods to choose between different domain pools and quantizers [8].

Acknowledgements

The authors are thankful to FAPESP, CNPq and CAPES for the financial support.

References

[1] Y. Fisher. *Fractal Image Compression: Theory and Application*. Springer-Verlag, London, UK, 1995.

[2] Y. Fisher and S. Menlove. Fractal Encoding with HV Partitions. In Y. Fisher, editor, *Fractal Image Compression - Theory and Application*, pages 119–126. Springer-Verlag, London, UK, 1995.

[3] S. Furao and O. Hasegawa. A Fast No Search Fractal Image Coding Method. *Signal Processing: Image Communication*, 19(5):393–404, 2004.

[4] R. Hamzaoui. Fast Iterative Methods for Fractal Image Compression. *Journal of Mathematical Imaging and Vision*, 11:147–159, October 1999.

[5] A. Jacquin. Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations. *IEEE Transactions on Image Processing*, 1(1):18–30, Jan. 1992.

[6] Y. H. Moon, H. S. Kim, and J. H. Kim. A Fast Fractal Decoding Algorithm based on the Selection of an Initial Image. *IEEE Transactions on Image Processing*, 9(5):941–945, May 2000.

[7] G. Øien and S. Lepsøy. *A Class of Fractal Image Coders with Fast Decoder Convergence*, chapter Fractal Image Compression, pages 153–175. Springer-Verlag, London, UK, 1995.

[8] A. Ortega and R. K. Rate-distortion methods for image and video compression. *IEEE Signal Processing Magazine*, 15(6):23–50, 1998.

[9] M. Pi, A. Basu, and M. Mandal. A New Decoding Algorithm based on Range Block Mean and Contrast Scaling. In *International Conference on Image Processing*, volume 2, pages II – 271–4 vol.3, Sept. 2003.

[10] M. Ruhl and H. Hartenstein. Optimal Fractal Coding is NP-Hard. In *Data Compression Conference*, pages 261–270, Mar. 1997.

[11] A. Said. *Lossless Compression Handbook*, chapter Arithmetic Coding. Communications, Networking, and Multimedia. Academic Press, 2003.

[12] D. Saupe, M. Ruhl, R. Hamzaoui, L. Grandi, and D. Marini. Optimal Hierarchical Partitions for Fractal Image Compression. In *IEEE International Conference on Image Processing*, 1998.

[13] I. Sutskover and D. Malah. Hierarchical Fast Decoding of Fractal Image Representation using Quadtree Partitioning. In *International Conference on Image Processing and Its Applications*, volume 2, pages 581–585, 1999.

[14] J. Teuhola. Fast Image Compression by Quadtree Prediction. *Real-Time Imaging*, 4:299–308, Aug. 1998.

[15] C. Tong and M. Pi. Fast Fractal Image Encoding based on Adaptive Search. *IEEE Transactions on Image Processing*, 10(9):1269–1277, Sept. 2001.

[16] X. Wu, D. Jackson, and H. Chen. Novel Fractal Image-Encoding Algorithm Based on a Full-Binary-Tree Searchless Iterated Function System. *Optical Engineering*, 44(10), 2005.