NOVEL FRACTAL IMAGE COMPRESSION METHOD WITH NON-ITERATIVE DECODER

Chang-Su Kim, Rin-Chul Kim and Sang-Uk Lee

Dept. of Control and Instrumentation
Seoul National University
E-mail : cskim@Claudia.snu.ac.kr

ABSTRACT

In this paper, we propose a novel fractal image compression technique, which does not require iteration at the decoder. The main problem relating to the conventional non-iterative algorithm is that the smooth region cannot be coded efficiently, since the size of the range block is limited to be less than 8×8. We alleviate this problem by generating two codebooks from planarly approximated image. In other words, the first codebook is generated by the smoothing operator for large and smooth range blocks and the second codebook is generated by the spatial contraction operator for small and active range blocks, respectively. The computer simulation results on the real images demonstrate that the proposed algorithm provides much better performance than most other fractal-based coders, in terms of the subjective quality as well as the objective quality (PSNR). Moreover, the proposed algorithm is very fast in decoding, since it does not require iteration at the decoder.

1. INTRODUCTION

Fractal compression, which is based on the IFS (iterated function systems) proposed by Barnsley [1], is a new approach to image coding recently. Most fractal coders[2] attempt to find a contraction map whose unique attractor approximates the source image. In the decoder, the map is applied iteratively to an arbitrary image to obtain the attractor. If the map can be represented with less bits than the source image, a coding gain is obtained.

In these fractal coders, the iteration number at the decoder depends on the source image, and the degradation of the attractor cannot be exactly predicted in the encoding process. Lepsøy et al. solved these problems by introducing a sufficient condition for iteration-free decoding [3]. Though [3] is very fast in decoding, the main problem is that the smooth region cannot be coded efficiently, due to the fact that range block larger than 8×8 cannot be used. In [3], notice that the size of the domain block should be square of the size of the range block.

In this paper, attempts have been made to extend the Lepsøy's non-iterative algorithm, so that the proposed algorithm is capable of employing the range block as large as 16×16 or 32×32 by incorporating the smoothing operator, instead of the spatial contraction operator, into the domain-range mapping for encoding the large and smooth range block. From the point of view that fractal coding is a self-VQ scheme [4], the proposed algorithm generates two codebooks from planarly approximated image. More specifically, the first codebook is generated by the smoothing operator for large and smooth range block and the second codebook is generated by the spatial contraction operator for small and active range block. By selecting the codebook, according to the characteristics of the range block, the proposed algorithm provides better performance than the Lepsøy's algorithm.

2. BACKGROUND

At the fractal encoder, the source image is partitioned into the range blocks R_i ($0 \leq i < N$), and each R_i is approximated by one or more fixed-basis-blocks B_j's and a domain block $D_{n(i)}$ in the same image, which is larger than the range block. The approximation of the R_i is given by

$$R_i \simeq \mathcal{L}_i(D_{n(i)}) + \sum_{j=1}^{M} c_{ij} B_j , \qquad (1)$$

where $n(i)$ is the index or location of the optimal domain block and the c_{ij}'s are real coefficients, respectively. The linear operator $\mathcal{L}_i$ shrinks the $D_{n(i)}$ to the range block's size, shuffles the pixels, and scales the greytone by a real factor sequentially [2].

The image to image mapping are composed of these blockwise mappings for each R_i. In Jacquin's algorithm [2], the mapping should be contractive for the iterated images to converge, and the iteration number is dependent on the source image. Øien and Lepsøy alleviated these problems by modifying the linear operator $\mathcal{L}_i$ [5], in which the approximation of the range block R_i is defined as

$$R_i \simeq \alpha_i \cdot \mathcal{O} \circ \mathcal{I}_i \circ \mathcal{C}(D_{n(i)}) + \beta_i \cdot B, \qquad (2)$$

where α_i, β_i are real coefficients, respectively, and the fixed-basis-block B is constant block where all pixel values are 1. The spatial contraction operator $\mathcal{C}$ shrinks the domain block to the size of the range block by decimation, and the isometry operator $\mathcal{I}_i$ shuffles the pixels. Notice that these two operators are same as the Jacquin's method. Let the $\mathcal{I}_i \circ \mathcal{C}(D_{n(i)})$ be A_i, then the orthogonalization operator $\mathcal{O}$ transforms A_i into A_i', by projecting it onto the orthogonal complement of $span\ \{B\}$. The introduction of this orthogonalization operator does not change the attractor, and the optimal α_i, β_i coefficients can be directly obtained by projection of R_i onto $span\ \{A_i'\}$ and $span\ \{B\}$ respectively, since A_i' and B are orthogonal to each other. Moreover, if $2^b \times 2^b$ and $2^d \times 2^d$ are the size of the range block and the domain block, respectively, then the iteration number K is fixed, regardless of the source image, given by

$$K = \lceil \frac{b}{d-b} \rceil, \qquad (3)$$

where $\lceil a \rceil$ denotes the smallest integer greater than a.

Lepsoy et al. proposed a non-iterative decoding algorithm [3], as a special case of the above method. In Eq.(3), if the size of the domain block is square of the range block's size ($d = 2b$), only one iteration is sufficient for the mapping to converge.

This non-iterative algorithm can be interpreted in the following way: Let the size of R_i and $D_{n(i)}$ be 4×4 and 16×16, respectively, and the $D_{n(i)}$ be composed of 16 consecutive range blocks, then the DC values for each R_i is contained in the β_i coefficient, and other AC informations are transmitted to the decoder via α_i coefficient, isometry operator $\mathcal{I}_i$, and the location of the domain block $n(i)$. Though the $D_{n(i)}$ is not known to the decoder, the spatially contracted domain block $\mathcal{C}(D_{n(i)})$ can be exactly reconstructed from the β coefficients of the 16 range blocks, which compose the $D_{n(i)}$, at the decoder. Therefore the attractor can be reconstructed without iteration. But, in this algorithm, the size of range block is limited to be less than 8×8. If the size of the range block is 16×16, the size of the domain block increases to 256×256, to ensure the details of the attractor. But notice that the size of the domain block is very large compared to that of the range block. Therefore there would be little chance of good domain-range mapping.

As in the self-VQ scheme, the Lepsøy's algorithm generates the codebook by spatially contracting the approximated image, in which each range block is approximated by a constant block. Generalizing this notion, if the source image is approximated by one or more fixed-basis-blocks and the codebook is generated from the approximated image, rather than the source image, then the attractor can be reconstructed without requiring iteration at the decoder.

3. PROPOSED ALGORITHM

As mentioned previously, the problem relating to the Lepsøy's non-iterative algorithm is that the smooth region cannot be encoded efficiently, since the size of the range block is limited. We alleviate this problem by generating the codebook which is suitable for large and smooth range blocks, as well as small and active range blocks. Let us describe the proposed algorithm subsequently.

3.1. Encoding Algorithm

First, we partition the source image into the range blocks of variable size by planar approximation. Secondly, two codebooks are generated from this planarly approximated image, one for active range blocks and the other for smooth range blocks. Then, each range block is encoded using these codebook by block matching algorithm.

STEP 1. Partitioning the source image and planar approximation : By employing the quadtree structure [6], the source image is partitioned into the range blocks of maximum size 32×32 and minimum size 4×4, according to the complexity. First, 32×32 block is approximated by plane, and if the approximation error is larger than the pre-specified threshold, then it is decomposed further into four smaller 16×16 blocks. This process is repeated until the planar approximation error becomes smaller than the threshold or 4×4 block is generated. Three fixed-basis-blocks X, Y, C are used for planar approximation in our approach. The pixel value for X, Y blocks increase linearly in x,y direction respectively, and the C block is constant block. If the block's size are $r \times r$, then these three block are defined as

$$\begin{aligned} X(m,n) &= 2m - r + 1, \\ Y(m,n) &= 2n - r + 1, \qquad (0 \le m, n < r), \\ C(m,n) &= 1, \end{aligned} \tag{4}$$

so that they are orthogonal to each other. By projecting each block R_i to the subspace spanned by these three orthogonal basis, the optimal planar approximation $\widetilde{R}$ is obtained as

$$\begin{aligned} \widetilde{R}_i &= \alpha_i \cdot X + \beta_i \cdot Y + \gamma_i \cdot C \\ &= \frac{< X, R_i >}{< X, X >} X + \frac{< Y, R_i >}{< Y, Y >} Y + \frac{< C, R_i >}{< C, C >} C, \end{aligned} \tag{5}$$

where $< A, B >$ denotes the inner product of A and B.

The smooth region is approximated by a large plane, and is partitioned into large range blocks, while active region (such as edge) is partitioned into small range blocks. In this paper, 4×4 or 8×8 range block is referred to as *active range block*, and 16×16 or 32×32 range block is referred to as *smooth range block*, respectively.

STEP 2. Generating the codebook : In STEP 1, the source image Λ is approximated blockwise by plane using the quadtree structure. Let us denote the planarly approximated image by $\widetilde{\Lambda}$. As mentioned previously, the $\widetilde{\Lambda}$ is transformed into codebooks Γ_1 and Γ_2. More specifically, the Γ_1 is a codebook for active range block, and the Γ_2 is a codebook for smooth range block, respectively.

Spatial contraction operator is used to transform the $\widetilde{\Lambda}$ into the Γ_1; *i.e.*, the pixel value for the Γ_1 is the average value of the consecutive 4×4 pixels in the $\widetilde{\Lambda}$. The spatially contracted image (codebook) Γ_1 is more detailed than the $\widetilde{\Lambda}$, making it suitable for the highly detailed active range blocks.

But, the planarly approximated image $\widetilde{\Lambda}$ is observed to be inappropriate to encode the smooth range blocks, since it yields much visible blocking artifact. Thus, it is transformed into the curved surface Γ_2 by a smoothing operator. In our approach, 9×9 moving average filter is used as the smoothing operator.

STEP 3. Block matching : Each range block R_i is now encoded using the codebook Γ_k (k=1 for active range block, k=2 for smooth range block). The range block R_i is approximated by the constant block C and a codebook vector D_j ($0 \le j < M_k$), which is the block in Γ_k, of the same size as the R_i.

For each j, D_j is first transformed into D'_j, which is orthogonal to the *span* $\{C\}$. The D'_j is obtained from

$$D'_j = D_j - \frac{< D_j, C >}{< C, C >} C. \tag{6}$$

Then, the approximated block $\widetilde{R}_{i,j}$ of R_i by D'_j is given by

$$\begin{aligned} \widetilde{R}_{i,j} &= \delta_{i,j} \cdot D'_j + \gamma_i \cdot C \\ &= \frac{< D'_j, R >}{< D'_j, D'_j >} D'_j + \gamma_i \cdot C. \end{aligned} \tag{7}$$

Then, among all possible j's, we need to find j_{min} which minimizes the approximation error $||R_i - \widetilde{R}_{i,j}||$. Let us denote the j_{min} for the R_i by $n(i)$, and $\delta_{i,j_{min}}$ by δ_i, for the sake of simplicity. Notice that the $n(i)$ is the index for the codebook and the δ_i is greytone scale factor.

Table 1: Bit Allocation

Planar Approximation	α_i	β_i	γ_i	total
	3	3	8	14 bits
Smooth Range Block	$n(i)$		δ_i	10 bits
	5		5	
Active Range Block	$n(i)$		δ_i	9 bits
	4		5	

Table 2: Comparison of Results

	Lepsøy's Algorithm		Proposed Algorithm	
	PSNR	bpp	PSNR	bpp
Lena	32.0	0.53	32.0	0.41
Pepper	31.5	0.53	32.2	0.38
Boats	31.2	0.59	31.1	0.54

3.2. Decoding Algorithm

The decoding algorithm is much simpler than the encoding algorithm. First, the planarly approximated image $\widetilde{\Lambda}$ is reconstructed with the quadtree information and the $\alpha_i, \beta_i, \gamma_i$. Secondly, two Γ_1, Γ_2 are generated from the $\widetilde{\Lambda}$ using the spatial contraction operator and the smoothing operator respectively. Lastly, each range block is reconstructed using the codebook vector in these codebooks. The codebook vector is indexed by $n(i)$ and transformed by Eqs.(6)-(7) using the coefficients δ_i, γ_i.

3.3. Parameter Quantization

For efficient transmission or storage, it is necessary to quantize the coefficients [7]. Let us describe the issue relating to the quantization in more detail.

The bit allocation is summarized in Table 1. The γ_i coefficients representing the DC values for each range block are uniformly quantized with 8 bits between 0 $\sim$ 255, and the $\alpha_i, \beta_i, \delta_i$ coefficients are Lloyd-Max quantized employing the probability distribution function obtained from test images. The index $n(i)$ is losslessly coded. Since 25 codebook vectors are searched for encoding smooth range block, 5 bits are allocated to the $n(i)$ of smooth range block. Similarly, 4 bits are allocated to the $n(i)$ of active range block for encoding 16 possible codebook vectors.

4. SIMULATION RESULTS

The proposed algorithm is tested on real grey level images. The planarly approximated image $\widetilde{\Lambda}$ for the 512 $\times$ 512 'Lena' image is presented in Fig 1 (quadtree structure is overlayed). In Fig 1, it is observed that the highly detailed region, such as hair or edge, is partitioned into small range blocks, while the smooth region is partitioned into large range blocks. The codebooks Γ_1, Γ_2 are shown in Fig 2. It is also observed that the Γ_1 is highly detailed, making it suitable for active range blocks. While the Γ_2 exhibits the form of curved surface, making it suitable for smooth range blocks. Fig 3 shows the decoded 'Lena' from these two codebooks. Though the region such as the shoulder is segmented into large range blocks (16 $\times$ 16 or 32 $\times$ 32), the blocking effect is almost invisible in those regions, since, in our approach, these areas are encoded with the smooth codebook Γ_2. It also appears that the degradation of edge is not severe, considering its low bit-rates.

Table 2 compares the performance of the proposed algorithm with that of Lepsøy's. It is seen that the proposed algorithm provides much better performance for 'Lena' and 'Pepper' images, since the smooth region in these images is coded efficiently with less bits. The 'Boats' image contains fine details, however, we can also see a slight improvement in this case.

Various simulation on other images also indicate that the proposed algorithm provides a better performance than most other fractal-based coders [2, 3], in terms of the subjective quality as well as the objective quality (PSNR).

5. CONCLUSION

In this paper, we proposed a novel fractal image compression technique, which does not require iteration at the decoder. We partitioned the source image with quadtree structure into the range blocks of variable size, by planar approximation. Then, two codebooks were generated from the planarly approximated image. The first codebook was generated by the spatial contraction operator for active range blocks, and the second codebook was generated by the smoothing operator for smooth range blocks, respectively.

It was demonstrated that the proposed algorithm provides an improvement in its performance, compared to the conventional fractal coders [2, 3]. Moreover, the proposed algorithm is very fast in decoding, since it does not require iteration at the decoder. The optimization process will make the proposed algorithm comparable to the JPEG standard for still image compression. Also the further research should be extended to the development of the efficient compression technique of moving image sequences.

6. REFERENCES

[1] Michael F. Barnsley. *Fractals Everywhere*. Academic Press, San Diego, 1988.

[2] Arnaud E. Jacquin. "Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations". *IEEE Trans. on Image Processing*, 1(1):18–30, January 1992.

[3] Skjalg Lepsøy, Geir E. Øien, and Tor A. Ramstad. "Attractor Image Compression with a Fast Non-iterative Algorithm". In *ICASSP*, volume 5, pages 337–340, 1993.

[4] Y. Fisher, T.P. Shen, and D. Rogovin. Fractal (self-VQ) encoding of video sequences. In *Proceedings of the SPIE: VCIP*, volume 2304-16, Chicago, IL, September 28-29 1994.

[5] Geir E. Øien and Skjalg Lepsøy. "A Class of Fractal Image Coders with Fast Decoder Convergence". In *Fractal Image Compression-Theory and Application*, chapter 8, pages 137–152. Springer-Verlag, New York, 1995.

[6] Eli Shusterman and Meir Feder. "Image Compression via Improved Quadtree Decomposition Algorithms". *IEEE Trans. Image Processing*, 3(2):207–215, March 1994.

[7] Geir E. Øien. "Parameter Quantization in Fractal Image Coding". In *ICIP*, volume 3, pages 142–146, 1994.

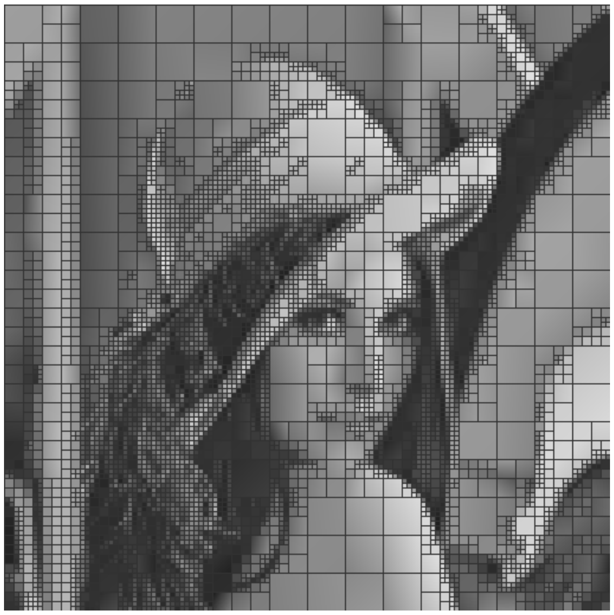
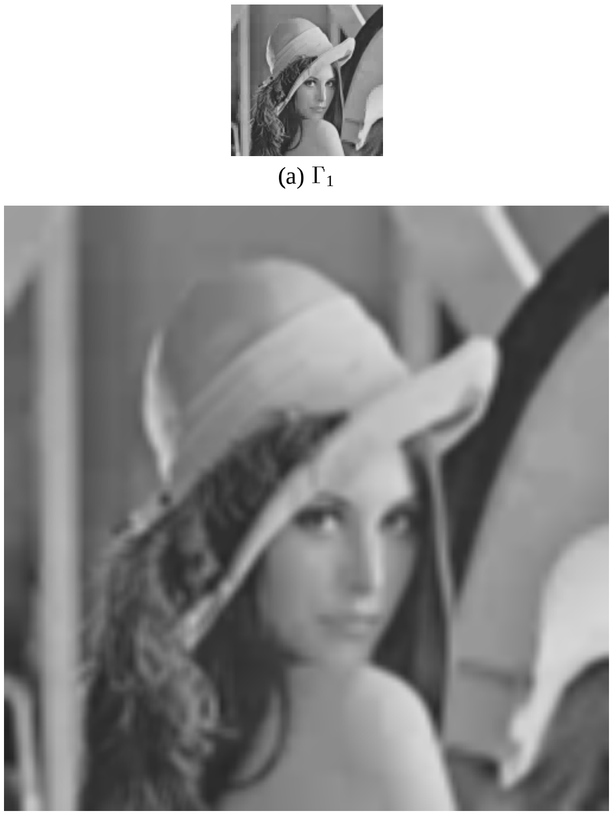


Figure 1: Planarly approximated ‘Lena’



(a) Γ_1

(b) Γ_2

Figure 2: Two codebook for ‘Lena’

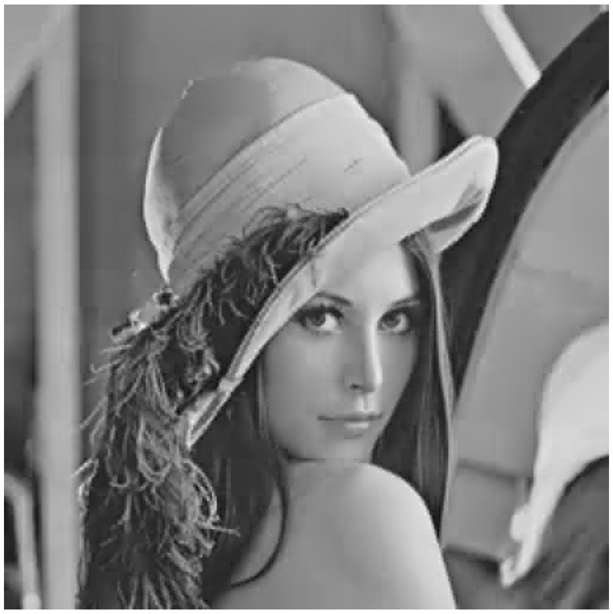


Figure 3: Decoded ‘Lena’, 0.41 bpp, 32.0 dB