LOSSLESS ACCELERATION OF FRACTAL IMAGE COMPRESSION BY FAST CONVOLUTION

Dietmar Saupe, Hannes Hartenstein

Universität Freiburg, Institut für Informatik, Am Flughafen 17, 79110 Freiburg, Germany

ABSTRACT

In fractal image compression the encoding step is computationally expensive. We present a new technique for reducing the computational complexity. It is lossless, i.e., it does not sacrifice any image quality for the sake of the speedup. It is based on a codebook coherence characteristic to fractal image compression and leads to a novel application of the fast Fourier transform-based convolution. The method provides a new conceptual view of fractal image compression. This paper focuses on the implementation issues and presents the first empirical experiments analyzing the performance benefits of the convolution approach to fractal image compression depending on image size, range size, and codebook size. The results show acceleration factors for large ranges up to 23 (larger factors possible), outperforming all other currently known lossless acceleration methods for such range sizes.

1. INTRODUCTION

In fractal image compression [1, 2] image blocks (ranges) have to be compared against a large codebook of blocks similar to vector quantization. For each such comparison a computationally expensive least-squares optimization is required. Typical codebooks consist of many thousands of blocks and the straightforward implementation of fractal image compression by "brute force" suffers from long encoding times. The existing methods to reduce the computational complexity are: discrete methods (classification and adaptive clustering), continuous methods (functionals or feature vectors), and dimensionality reduction methods. For a survey of these see [3]. In these methods suitable subsets of the codebook are eliminated from the search. Most of the techniques are *lossy* in the sense that they trade in a speedup for some loss in image fidelity. In contrast, with a *lossless* method the codebook block with the minimal (collage) error is obtained rather than an acceptable but suboptimal one. Three lossless methods are known:

(1) Rademacher labelings [4]. Speedup factor ≈ 1.5.

(2) Dimension reduction by partial distortion elimination [5]. Speedup factor < 4.

(3) Dimension reduction by image pyramids [6]. Speedup factor ≈ 4.

Due to the moderate speedup factors these techniques are used in conjunction with other (lossy) methods. An almost lossless acceleration method based on image pyramids with impressive results is given in [7]. Our new solution offered in [8] is the first one that takes advantage of the fact that the codebook blocks, taken from the image, are usually overlapping. The fast convolution — based on the convolution theorem and carried out in the frequency domain — is ideally suited to exploit this sort of codebook coherence. This paper focuses on implementation issues and presents the first computer experiments analyzing the performance benefits of the convolution approach to fractal image compression depending on image size, range size, and codebook size.

2. FRACTAL IMAGE COMPRESSION VIA CONVOLUTION

Let us consider the generic type of fractal image compression and remark later about generalizations. A partitioning of the image into disjoint image blocks (called *ranges*) is defined. A pool of (larger) image blocks (called *domains*) serves as a source of blocks from which ranges can be approximated as the sum of a DC component and a scaled copy of a domain block (collage). For a range block R the domain blocks are twice the linear size. The domain blocks are shrunken by pixel averaging to match the range block size. This gives a pool of *codebook blocks* $D_1, \ldots, D_{N_D}$. For range R and codebook block D we let

$$(s, o) = \arg \min_{s,o \in \mathbf{R}} \|R - (sD + o\mathbf{1})\|^2$$

where $\mathbf{1}$ is the constant block with unit intensity at every pixel. The scaling coefficient s is clamped to $[-1, 1]$ to ensure convergence in the decoding and then both s and o are uniformly quantized. The *collage error* for

range R is $E(D,R) = \|R - (sD + o\mathbf{1})\|^2$. The codebook block D_k with minimal collage error $E(D_k, R)$ yields the fractal code for range R consisting of the index k and the corresponding quantized scaling and offset parameters s and o. The decoding in a conventional fractal codec proceeds by iteration of the range approximations starting from an arbitrary initial image.

The essential part of this basic computation in fractal image compression is a certain convolution [8]. To see that denote by $\langle \cdot, \cdot \rangle$ the inner product in a Euclidean space of dimension n (= number of pixels in a range block). For a range block R and codebook block D the optimal coefficients are

$$s = \frac{n\langle D,R\rangle - \langle D,\mathbf{1}\rangle\langle R,\mathbf{1}\rangle}{n\langle D,D\rangle - \langle D,\mathbf{1}\rangle^2}, \quad o = \frac{1}{n}\left(\langle R,\mathbf{1}\rangle - s\langle D,\mathbf{1}\rangle\right).$$

For any (s,o) the error $E(D,R)$ can be regarded as a function of $\langle D,R\rangle, \langle D,D\rangle, \langle D,\mathbf{1}\rangle, \langle R,R\rangle$, and $\langle R,\mathbf{1}\rangle$,

$$E(D,R) = s(s\langle D,D\rangle + 2(o\langle D,\mathbf{1}\rangle - \langle R,D\rangle)) + \\ + o(on - 2\langle R,\mathbf{1}\rangle) + \langle R,R\rangle.$$

The evaluation of s, o, and $E(D,R)$ requires 22 floating point operations, the use of quantized s, o leads to an additional cost, e.g., 19 floating point operations in our implementation.

Typically, when all range blocks are of the same shape, the computations are organized in two nested loops:

- Global preprocessing: compute $\langle D,D\rangle, \langle D,\mathbf{1}\rangle$ for all codebook blocks D.
- For each range R do:
  - Local preprocessing: compute $\langle R,R\rangle, \langle R,\mathbf{1}\rangle$.
  - For all codebook blocks D do:
    - Compute $\langle D,R\rangle$ and $E(D,R)$.

The calculation of the inner products $\langle D,R\rangle$ dominates the computational cost in the encoding, e.g., direct computation of $\langle D,R\rangle$ with R containing $16 \cdot 16$ values requires 512 floating point operations which is twelve times the cost of computing the error. The codebook blocks D are typically defined by downfiltering the image to half its resolution. Any subblock in the downfiltered image, that has the same shape as the range, is a codebook block for that range. In this setting the inner products $\langle D,R\rangle$ are nothing but the *finite impulse response* (FIR) of the downfiltered image with respect to the range R. In other words, the convolution (or, more precisely, the cross-correlation) of the range R with the downfiltered image is required. This discrete two-dimensional convolution can be carried out more efficiently in the frequency domain when the range block is not too small (convolution theorem). This procedure takes the inner product calculation out of the inner loop and places it into the local preprocessing where the inner products $\langle D,R\rangle$ for *all* codebook blocks D are obtained in one batch by means of fast Fourier transform convolution. Clearly, the method is lossless.

	Image Size			
	256×256		512×512	
Range Size	std	conv	std	conv
4×4	524	929	2097	3837
8×8	2097	976	8388	4009
16×16	8388	1036	33554	4193
32×32	33554	1139	134217	4408

Table 1: Number of floating point operations (in thousands) to compute the correlations between a single range block and all possible domain blocks using the standard "brute force" scheme (std) and the convolution based approach (conv).

Moreover, the global preprocessing requires a substantial amount of time, but can be accelerated by the same convolution technique. The products $\langle D,\mathbf{1}\rangle$ are obtained by convolution of the downfiltered image with a range block where all intensities are set equal (called *range shape matrix*). The sum of the squares is computed in the same way where all intensities in the downfiltered image are squared before the convolution.

3. IMPLEMENTATION AND RESULTS

The cross-correlation between an image block and the downfiltered image of size $N/2 \times N/2$ can be computed in the frequency domain using the fast Fourier transform (FFT):

1. Compute the 2D-FFT of the downfiltered image.
2. Enlarge image block by zero padding to size $N/2 \times N/2$.
3. Compute the 2D-FFT of the enlarged image block.
4. Perform the complex conjugate of the last transform.
5. Do the complex multiplication of both transforms.
6. Do the inverse FFT of the result.

In our implementation we use the FFT code from [9]. The algorithm for the fractal image encoding with fast convolution is summarized in the following:

A. Input: Image of size $N \times N$, range size $B \times B$.
B. Global preprocessing.
   1. Compute the downfiltered image by pixel averaging.
   2. 2D-FFT of the downfiltered image.

		Image Size					
		256 × 256			512 × 512		
Range Size		std sec	conv sec	speed up	std sec	conv sec	speed up
4×4	preprocessing	0.15	0.51	0.29	0.61	2.26	0.27
	search/range	0.18	0.25	0.72	0.74	1.07	0.69
	total	757.0	1006.8	0.75	12076.3	17566.1	0.69
8×8	preprocessing	0.56	0.52	1.04	2.27	2.29	0.99
	search/range	0.44	0.25	1.76	1.78	1.10	1.62
	total	447.3	259.8	1.72	7282.5	4503.8	1.62
16×16	preprocessing	2.22	0.52	4.27	8.97	2.29	3.92
	search/range	1.50	0.26	5.77	6.95	1.10	6.32
	total	386.6	67.5	5.72	7121.6	1126.5	6.32
32×32	preprocessing	9.01	0.54	16.68	35.81	2.35	15.24
	search/range	6.45	0.27	23.89	25.95	1.14	22.76
	total	422.0	18.00	23.44	6678.5	294.2	22.70

Table 2: Timing results (measured on an 132 MHz MIPS R4600 processor, cc-compiler option O1). The columns labeled "std" and "conv" list the cpu seconds consumed by the conventional "brute force" method and our convolution based approach.

3. Fast cross-correlation with the range shape matrix yielding the $\langle D, \mathbf{1}\rangle$ for all codebook blocks D.
4. Fast cross-correlation of the squared downfiltered image and the range shape matrix yielding the summed squares $\langle D, D\rangle$ for all codebook blocks D.

C. Searching. For each range R do steps C1 to C4.

1. Compute $\langle R, \mathbf{1}\rangle$ and $\langle R, R\rangle$.
2. Fast cross-correlation of the range block using the result of step B2 obtaining all products $\langle D, R\rangle$.
3. For each codebook block D compute the (quantized) coefficients s, o, and the collage error $E(D, R)$ using the results of steps B3, C1, and C2.
4. Extract the minimal collage error and output the fractal code for the range.

To compare this scheme to the standard "brute force" method, we check the number of floating point operations that are required to compute the inner products $\langle R, D\rangle$ between a fixed range block R and all domain blocks D. In the "brute force" case, an n-dimensional inner product takes $2n$ floating point operations. For an image of size $N \times N$ there are $N/2 \times N/2$ domain blocks (we include blocks that wrap around image borders), thus our total is $2n \cdot N/2 \cdot N/2$. With the convolution based scheme, we must taken into account the costs for transforming the enlarged range block R, the multiplication between this transform and the Fourier transform of the downfiltered image, and the backward transformation. The Fourier transform of a real $N/2 \times N/2$ array can be computed with $N/2$ complex 1-dimensional transformations of length $N/4$ and $N/4$ complex 1-dimensional transformations of length $N/2$ plus an additional $3N^2$ floating point operations. The same cost apply to the inverse transformation. The cost for a 1-dimensional Fourier transform of N complex values is $5N \log_2 N$. For the forward transform, performance is enhanced considerably because most entries in the zero padded range image are zero. This leads to numbers of floating point operations counted at run-time and given in Table 1.

To determine actual timing results, we use the gray scale image Lenna in sizes 256 × 256 and 512 × 512 pixels. The downfiltered images have resolutions of 128 × 128 and 256 × 256 yielding codebooks of size $128^2 = 16384$ and $256^2 = 65536$. We partition the image uniformly with square blocks of sizes ranging from 4 × 4 up to 32 × 32. The fractal encoding is run with and without using the fast convolution method, producing the same fractal codes. The resulting run times are collected in Table 2 where we differentiate between global preprocessing and the main loop of the algorithm. The result shows no or small speedup for ranges up to size 8 × 8. However, the speedup grows linearly with the number of pixels in the range. For the large 32 × 32 range size we get a speedup of about 23. Overall, this test shows that the performance of our method meets that of the other lossless methods for 16 × 16 ranges and outperforms the others for larger ones.

We continue this section with remarks concerning possible optimizations, different domain pool sizes and the use of square isometries.

1. **Optimization.** The method we have employed for computing the Fourier transform is suboptimal. One probably better uses the Fourier transform for real

valued sequences outlined in [10] and a split-radix decimation-in-frequency (decimation-in-time for the inverse transform) approach. The use of number theoretic transformations [11] appears to be limited because of the restrictions imposed by image size and maximal intensity value.

2. **Domain pool size.** In our method there is a *natural domain pool* obtained by shifting a mask of twice the range size over the original image with a step size of two pixels horizontally and vertically (with wraparound effect at image borders). To enlarge this domain pool we can consider several additional downfiltered images, corresponding to domains obtained by offsetting the natural ones. Separate convolutions for each one of them result in a linear time complexity; increasing the domain pool by some factor leads to same increase in computation time.
3. **Square isometries.** It is easy to incorporate square isometries of codebook blocks to enrich the codebook. In our method we can take advantage from direct methods to obtain the Fourier transforms for rotated and reflected images.

4. CONCLUSIONS AND FUTURE WORK

In this work we have shown how the fast convolution provides a new approach to reducing the time complexity of fractal image encoding. Our method is fundamentally different from all other acceleration schemes. It is interesting not only with regard to its performance but also from a conceptual point of view providing a new interpretation of fractal image compression in general [8]. It features the following properties:

- It is lossless, thus, equivalent to full search.
- Constant cost per search for all range sizes.
- Thus, it is fast for large ranges.
- The method is linear in the domain pool size.
- Memory requirements are very moderate.
- Well suited for parallel implementation or in hardware.

Clearly, our method is applicable also to quadtree partitionings and will lead to compound acceleration factors which need to be compared to those obtained by state-of-the-art classification methods taking into account the lossless character of our method. Moreover, arbitrary range shapes can be accommodated simply by zero padding the pixels that are not in the range. Thus, the method has a strong potential in applications where an adaptive image partition provides for large irregularly shaped ranges and a fractal code is sought, as, e.g., in [12]. In this case other (lossy) acceleration methods are infeasible or may be more expensive. Another interesting aspect is the combination of our method with other (lossy) acceleration methods.

5. REFERENCES

[1] Fisher, Y., *Fractal Image Compression — Theory and Application,* Springer-Verlag, New York, 1994.

[2] Jacquin, A. E., *Image coding based on a fractal theory of iterated contractive image transformations,* IEEE Trans. Image Proc. 1 (1992) 18–30.

[3] Saupe, D., Hamzaoui, R., *Complexity reduction methods for fractal image compression,* in: I.M.A. Conf. Proc. on *Image Processing; Mathematical Methods and Applications,* Sept. 1994, J. M. Blackledge (ed.), Oxford University Press, 1996, to appear.

[4] Bedford, T., Dekking, F. M., Breewer, M., Keane, M. S., van Schooneveld, D., *Fractal coding of monochrome images,* Signal Processing 6 (1994) 405–419.

[5] Caso, G., Obrador, P., Kuo, C.-C. J., *Fast methods for fractal image encoding,* Proc. SPIE Visual Communication and Image Processing '95, Vol. 2501 (1995) 583–594.

[6] Dekking, M., *Fractal image coding: some mathematical remarks on its limits and its prospects,* in: Conf. Proc. NATO ASI *Fractal Image Encoding and Analysis,* Trondheim, July 1995, Y. Fisher (ed.), to appear in Springer-Verlag, New York, 1996. Also: Technical Report 95-95 of the Faculty of Technical Mathematics and Informatics, Delft University of Technology, 1995.

[7] Lin, H., Venetsanopoulos, A. N., *A pyramid algorithm for fast fractal image compression,* Proc. 1995 IEEE Intern. Conf. on Image Processing (ICIP), Washington, Oct. 1995.

[8] Saupe, D., *A new view of fractal image compression as convolution transform coding,* IEEE Signal Processing Letters 3,1996.

[9] Press, W. H., Teukolsky, S. A., Vetterling, W. T., Flannery, B. P., *Numerical Recipes in C,* Second Edition, Cambridge University Press, 1992.

[10] Sorensen, H., Jones, D., Heidman. M., Burrus, S., *Real-Valued Fast Fourier Transform Algorithm,* IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-35, June 1987, 849–863.

[11] Nussbaumer, H. J., *Fast Fourier Transform and Convolution Algorithms,* Second Edition, Springer-Verlag, 1990.

[12] Saupe, D., Ruhl, M., *Evolutionary Fractal Image Compression,* Proc. 1996 IEEE Intern. Conf. on Image Processing (ICIP), Lausanne, Sept. 1996.