Merge Operation Effect On Image Compression Using Fractal Technique

By

Ghada K.Toama* Luay K. Aboud**

Date of acceptance 13/12/2006

Abstract:

Fractal image compression gives some desirable properties like fast decoding image, and very good rate-distortion curves, but suffers from a high encoding time. In fractal image compression a partitioning of the image into ranges is required. In this work, we introduced good partitioning process by means of merge approach, since some ranges are connected to the others.

This paper presents a method to reduce the encoding time of this technique by reducing the number of range blocks based on the computing the statistical measures between them . Experimental results on standard images show that the proposed method yields minimize (decrease) the encoding time and remain the quality results passable visually.

1.. Introduction

In fractal image compression the image to be coded is partitioned into blocks called ranges. Each range is approximated by another part of the image called domain. Finding a partitioning that minimizes the approximation error is a hard problem in fractal image compression. Traditionally, hierarchical schemes like quadtree, rectangle (horizontal _vertical) partitioning have been used [1,2].

Here, we consider the merge process in which ranges are unions of small image blocks called atomic blocks based on the correlation between adjacent range blocks that produce significant gains in term of time and performance over classic (traditional) fixed, and hierarchical of quadtree partitioning process.

This paper leads to a significant speed up, additionally by vary the size of atomic blocks and the threshold of (correlation) between adjacent blocks which leading to minimize the encoding time.

2.. Basics of Fractal Image Compression

This paragraph reviews the standard type of fractal image encoding and introduces some basic notations used in this work. For a range block R we consider a pool of domain blocks twice the linear sizes. The domain blocks are shrunken by pixel averaging to match the range block size. This pool of domain (codebook) is enlarged by including all 8 isomeric versions (rotations and reflections (flips) of a block. This gives a pool of codebook (domain) blocks $D_1,\ldots,D_{ND}$. For a range R and codebook (domain) block D we let

$$\left[(s,o) = \arg\min_{s,o\in R} \left\| R - ({}_s D + o1 \right\|^2 \right]$$

* Collage of Science, Remote Sensing Unit, University of Baghdad.

**Collage of Science, Computer Science Department ,University of Baghdad.

Where 1 is the flat block with intensity 1 at every pixel. The parameters *s* and *o* are called scaling and offset, respectively. The coefficient s is clamped to $[-s_{max}, s_{max}]$ with $0< s_{max}<1$ to ensure convergence in the decoding and then both *s* and *o* are uniformly quantized yielding $\bar{s}$ and $\bar{o}$. The collage error for range *R* and codebook (domain) block *D* is

$$E(D,R)=\left\|R-(\bar{s}D+\bar{o}1)\right\|^2$$

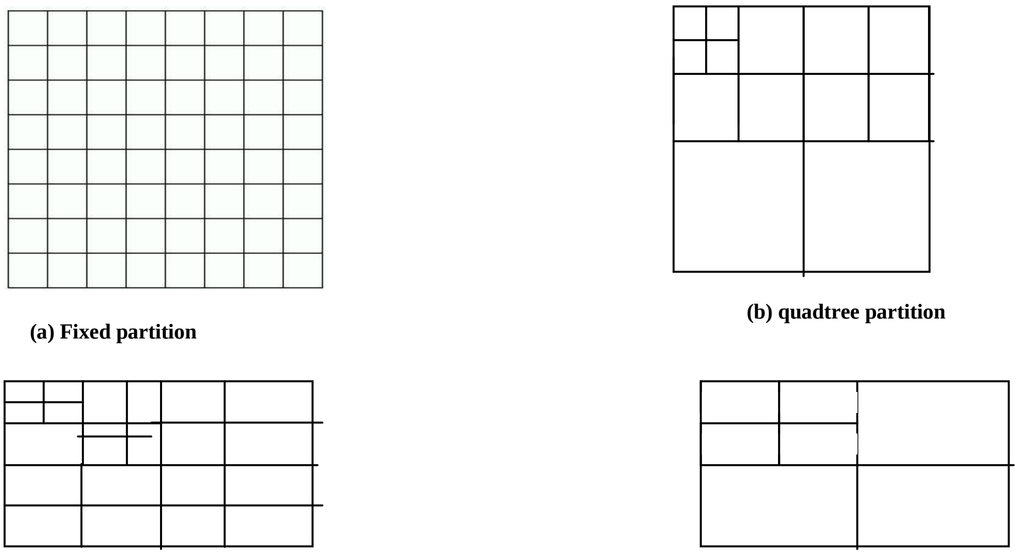
The fractal code for range R consist of the code book (domain pool coordinates (X,Y)) , Symmetry index (rotation, reflection) and the corresponding quantized scaling and offset parameters $\bar{s}$ and $\bar{o}$.

3.. Outline Of The Proposed Algorithm

Our proposed method starts with the partitioning image (i.e., after applying fixed or hierarchical quadtree) where we have atomic range blocks (i.e., 4x4 for fixed and quadtree, pixel blocks) then we merge neighboring ranges that not exceed the threshold to yield a partitioning by merge process with a decreasing number of ranges.

The ranges are merged as follows:

1. Startup by assuming that the whole input (initial) image is partitioned using traditional partitioning method(i.e., fixed and quadtree) as shown in fig(1 a,b).
2. Determine the threshold value that play as a controller of the merging process. Here the thresholding value represents the mean and standard deviation between each quarter.
3. At the end, the merging test is implemented, it is based on the merging function which is a function that take each 4 quarters then compute the mean and standard deviation of each of the quarter if it is less than or equal to the threshold of each of 4 quarter of these blocks are merged into 1 blocks as shown in figure(1 d,e)
4. Perform the encoding process using the merged process.



(a) Fixed partition

(b) quadtree partition

(d) After merging process the Fixed partition converted into quadtree partition (hierarchal) and number of range become smaller

(e) After merging process the quadtree partition range become smaller

Fig. (1) :illustrate the partitioning blocks before and after merging process

Table 1:illustrate the effect of merging process from fixed partition to quadtree partition

Lena image before merging process						Lena image after merging process					
Block Size	**Number Of Blocks**	**Comp. Ratio**	**PSNR (dB)**	**SNR (dB)**	**Encode Time (Sec)**	**Thresh old Value**	**Number Of Blocks**	**Comp. Ratio**	**PSNR (dB)**	**SNR (dB)**	**Encode Time (Sec)**
4x4	4096	4.92	30.6	23.4	37	10	3076	4.60	30.2	23.0	31
4x4	4096	4.92	30.6	23.4	37	30	2086	6.78	28.4	21.2	25
4x4	4096	4.92	30.6	23.4	37	50	1585	8.93	26.6	19.4	23
4x4	4096	4.92	30.6	23.4	37	70	1366	10.36	26.2	19.0	21
4x4	4096	4.92	30.6	23.4	37	100	1156	12.24	25.1	17.8	20
8x8	1024	13.82	24.4	17.1	21	10	877	16.13	24.3	17.0	20
8x8	1024	13.82	24.4	17.1	21	40	544	25.99	23.2	15.9	19
8x8	1024	13.82	24.4	17.1	21	60	409	34.56	22.2	14.9	18
8x8	1024	13.82	24.4	17.1	21	80	340	41.56	21.1	13.8	17
8x8	1024	13.82	24.4	17.1	21	100	295	41.55	20.0	12.7	16
Rose image before merging process						Rose image after merging process					
Block Size	**Number Of Blocks**	**Comp. Ratio**	**PSNR (dB)**	**SNR (dB)**	**Encode Time (Sec)**	**Thresh old Value**	**Number Of Blocks**	**Comp. Ratio**	**PSNR (dB)**	**SNR (dB)**	**Encode Time (Sec)**
4x4	4096	4.92	35.9	29.6	37	10	3376	4.196	35.7	29.4	32
4x4	4096	4.92	35.9	29.6	37	30	2221	6.377	33.6	27.3	29
4x4	4096	4.92	35.9	29.6	37	50	1582	8.951	31.7	25.4	27
4x4	4096	4.92	35.9	29.6	37	100	1147	12.34	29.7	23.4	25
8x8	1024	13.82	28.4	22.1	21	10	949	14.915	28.3	22.0	19
8x8	1024	13.82	28.4	22.1	21	40	649	21.8	26.9	20.6	15
8x8	1024	13.82	28.4	22.1	21	60	487	29.03	25.5	19.2	13
8x8	1024	13.82	28.4	22.1	21	80	385	36.70	24.2	17.8	12

The outlined algorithm can be implemented very efficiently. In fact, it can be speed up some traditional techniques from fractal coding, yielding a state-of-the-art program that is considerably faster than fractal coders which achieve a reasonable rate-distortion performance.

In the initialization partitioning phase a fractal encoding of the image is sought for which all ranges are atomic blocks. This phase is computationally expensive. Each range block is compared with the same set of domain blocks. Therefore, we can use acceleration techniques that involve heavy preprocessing such as merging the nearest neighbor of blocks based on some statistical measure like mean and standard deviation.

4. RESULTS :

This section presents experimental results showing the efficiency of the proposed method. The performance tests carried out for a diverse set of well-known images of size 256x256 gray levels with 8bpp, using Visual C++6.0 programming language and the time is measured in seconds.

This study focuses on the implementation issues and presents the first empirical experiments analyzing the performance of benefits of merging approach to fractal image compression.

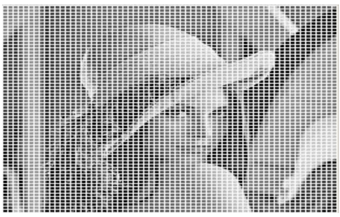
In our experiments, the threshold value of merging process varied depending on the properties (attributes) of the desired (decoded) image (i.e., we use big (large) threshold value if we want fast encode time and the decoded image may lose some precision and vice versa). Figure 2 shows the example of the merging process applied on the fixed partition with variable threshold value and variable block size, we see, varying the atomic block sizes depending on desired compression ratio improves the performance of our method (i.e., by merging process we can gain better compression ratio since many blocks merged and as a result minimum number of bits to represent the block).

In table1,2 we illustrate the performance of our method for fixed and hierarchical partitioning method respectively ,in this search, we have adapted the atomic block size for method depending on the compression rate.

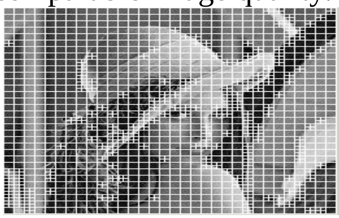
Table 1 and 2 compares the computing times PSNR, compression ratio between the traditional encoder and merging encoder.

These experiments show that our algorithm belongs to the best fractal coders regarding image fidelity, but cuts down the encoding time to a fraction of the running time of fractal coders that achieve comparable image quality.

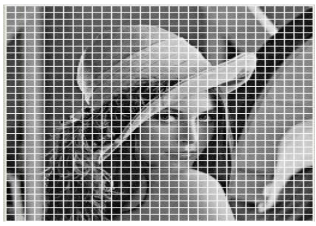
Lena Image

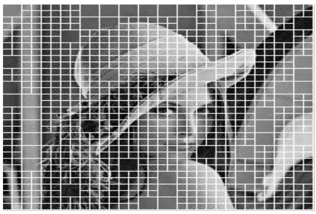


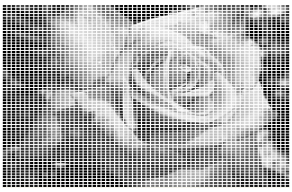
Fixed partition Block Size=4x4
Number of blocks=4096
PSNR=30.6 SNR=23.4
Encode Time=37
Comp. Ratio=4.92



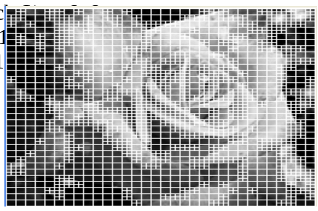
Fixed partition after merging process
Threshold=50
Number of blocks=1585
PSNR=26.6 SNR=19.4
Encode Time=23
Comp. Ratio=8.93







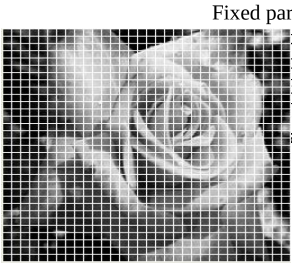
Fixed partition Bloc
Number of blocks=1
PSNR=24.4 SNR=1
Encode Time=21
Comp. Ratio=13.82

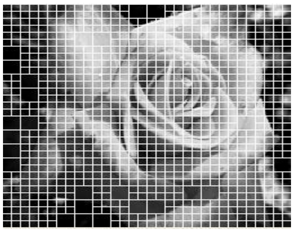


Fixed partition after merging process
Threshold=10
Number of blocks=877
PSNR=24.3 SNR=17.0
Encode Time=20
Comp. Ratio=16.13

rtition Block Size=4x4
of blocks=4096
5.9 SNR=29.6
Time=37
atio=4.92

Fixed partition after merging process
l=30
of blocks=2221
3.6 SNR=27.3
ime=29
atio=6.377





partition Block Size=8x8
er of blocks=1024
=28.4 SNR=22.1
e Time=21
. Ratio=13.82

Fixed partition after merging process
Threshold=10
Number of blocks=949
PSNR=28.3 SNR=22.0
Encode Time=19
Comp. Ratio=14.915

illustrate the effect of merging process on the fixed partition using Lena and Rose

Before Merge — After Merge

Max Block Size	Min Block Size	Number Of Blocks	Comp. Ratio	PSNR (dB)	SNR (dB)	Encode Time (Sec)	Threshold Value	Number of Blocks After Merge	Comp. Ratio	PSNR (dB)	SNR (dB)	Encode Time (Sec)
16	2	3154	4.49	28.5	21.2	28	30	2869	4.79	26.7	19.6	25
16	2	3154	4.49	28.5	21.2	28	50	2002	5.21	25.2	18.0	20
16	2	315[illegible]	[illegible]	[illegible]5	21.2	28	60	[illegible]	[illegible]1	24.0	17.1	17
Rose Image												
Max Block Size	**Min Block Size**	**Number Of Blocks**	**Comp. Ratio**	**PSNR (dB)**	**SNR (dB)**	**Encode Time (Sec)**	**Threshold Value**	**Number of Blocks After Merge**	**Comp. Ratio**	**PSNR (dB)**	**SNR (dB)**	**Encode Time (Sec)**
16	2	2688	5.27	31.95	25.66	40	10	2120	7.12	30.1	24.0	35
16	2	2688	5.27	31.95	25.66	40	40	1844	9.01	28.5	22.3	30
16	2	2688	5.27	31.95	25.66	40	60	1004	11.05	26.8	21.0	22

Table 2:illustrate the effect of merging process on the quadtree partitioning method

REFERENCES:

1. Saupe, D., and Ruhl, M., 1996, *Evolution fractal image compression*, Proceedings IEEE ICIP, Lausanne, pp. 129-132.

2. Thomas, L., and Deravi, F., 1995, *Region-based fractal image compression using heuristic search*, IEEE Transactions on Image Processing , pp.832-838.

3. Ghada, K., 2001, Adaptive Fractal Image Compression, M. Sc. thesis, National Computer Center/ Higher Education of computer and Informatics.

4. Haider, H.R., 2000, Adaptive Fractal Image Compression, M. Sc. thesis, College of Science, AL-Mustansiriya University.

5. Jamila, H., 2001, Fractal Image Compression, Ph.D. thesis, Collage of science, university of Baghdad .

تأثير عملية الدمج على ضغط الصور باستخدام تقنية الكسوريات

غادة كاظم طعمة* **لؤي كاظم عبود****

*مدرس مساعد/ كلية العلوم /وحدة الاستثمار والتحسس النائي

**استاذ مساعد/كلية العلوم/ قسم علوم الحاسبات

الخلاصـة:

ضـغط الصـور باسـتخدام الكسـوريات قـد أعطـى بعـض المميـزات المرغوبـة و التـي منهـا اسـترجاع الصـورة المضغوطـة بأقل وقت ممكن وكذلك أفضل تمثيل بياني للعلاقة rate-distortion curves للت تعاني مـن مشكلة طول وقت التشفير (المدة الزمنية اللازمة للتشفير).ضـغط الصـور باستخدام الكسريات يتطلب تقطيع الصـورة إلـى مجالات. في هذا العمل تم تقديم طريقة جيدة باستخدام طريقة الدمج، لانة هناك بعض المجـالات مرتبطـة مـع بعضـها البعض الآخر.

هذا البحث يقدم طريقة لتقليل وقت التشفير وذلك بتقليل عدد البلوكـات الخاصـة بمجـال الصـورة و ذلـك عـن طريـق حسـابات إحصـائية بـين هـذة المجـالات.النتـائج العلميـة علـى الصـور القياسـية بينـت ان هـذة التقنيـة تـؤدي إلـى تقليـل (تقليص) في زمن التشفير و المحافظة على نوعية مقبولة للصورة المرئية.