Improving Fractal Image Compression Scheme through Quantization Operation

Ghada K.Toama* **Hyder H.Razaak****

Date of acceptance 13/12/2006

Abstract

We explore the transform coefficients of fractal and exploit new method to improve the compression capabilities of these schemes. In most of the standard encoder/ decoder systems the quantization/ de-quantization managed as a separate step, here we introduce new way (method) to work (managed) simultaneously. Additional compression is achieved by this method with high image quality as you will see later.

Introduction

JPEG and Vector Quantization have certainly been the most commonly used methods for image compression, followed more recently by wavelet methods. However, over the past few years, there has been also much interest and development in fractal–based methods of image compression [1,2,3] accompanied by a number of significant developments [4]. Most of these schemes are centered around the method of Iterated Function System [1,2]. The common conception is that fractal–based schemes exploit geometric self-similarities that are inherent in images. However, amore realistic justification of these schemes is that they exploit the local scaling properties of irregularities, e.g. edges, across a range of resolutions, in turn reflected in the decay of associated wavelet coefficients [3].

Fractal-based schemes seek to approximate a target image function f(x,y) as a union of tiling of geometrically shrunken copies of subsets of f with suitably transformed gray level values. As such, the image f is approximated by the fixed point $\overline{f}$ of a contractive operator or fractal transform T. From Banach's Fixed Point Theorem, the approximation $\overline{f}$ may be generated by the iteration procedure $f_{n+1}=Tf_n$ (f_0 can be a blank screen, for example.). It is the transform T, in particular the coefficients of its geometric and gray-level transformations, that is stored in computer memory, often requiring much less storage than the original image f, resulting is significant (lossy) data compression.

In this study, we introduce new method of quantization and de-quantization for transform coefficients (scale, offset) to improve their compression capabilities, an additional compression is achieved by this methods of the quantized coefficients.

Basics of Fractal Compression

As started earlier, given a target image f, we seek to find a contractive fractal transform operator T with fixed point $\overline{f}$ approximates $\overline{f}$ to some suitable degree of accuracy, i.e. d($\overline{f}$,f)<€ where d denoted £2(RMS) metric. From an important corollary of Banch's Fixed Point Theorem, referred to in the IFS

*Remote Sensing Unit, Collage of Science,University of Baghdad
**Computer Science Department College of Science, AL-Mustansiriya University

literature as the collage theorem [1], this inverse problem reduces the following: Find an operator T such that d(f,T(f)) is suitably small. In other words, T should send the target close to itself.

Originally, IFS-type methods sought to express a target set or image as a union of shrunken copies of itself. However, most real-world objects are rarely so entirely self-similar. Instead, self-similar may be exhibit only locally, in the sense that sub regions of an image may be self-similar. The basis of the block encoding scheme which we briefly summarize below:

The primitive fractal encoder variable–block size (VBS) consist of the following steps:

1. Image partition (Create Range Pool): At first the image to be coded must be partitioned into non–overlapping blocks of different sizes, by using quadtree partitioning method [5]. The resulting blocks are called range blocks R_i.
2. Construction of Domain pool (Create Domain Pool) : by averaging method through the original image (i.e., each four pixel from the original image averaged into one pixel in the domain pool), this will lead to a creation of a list of overlapped or non-overlapped domain blocks from the image plane. The test for similarity between the domain and range blocks must be performed on the same block sizes. The goal of creating (generating) the domain pool is to approximate the pixel intensities of the range block with those of a domain block, good approximation are obtained when many domain blocks are allowed. When the list of domain blocks become large the searching time for the domain block will increases and become time consuming [6].
3. The search: For each range block R the optimal affine approximation (mapping) is given by,

$R_i = SD_i + O$ ……(1)

Where Ri is the range pixel value,

D_i is the corresponding domain pixel value,

S is the scaling coefficient,

O is the offset coefficient.

For mapping each range block, the search process implies that for each domain block (D_i), listed in the domain pool,

a) Compute the optimal approximation

$R = S D_i + O$ for the eight cases of symmetry operations, where the symmetry state include:

Identity.

Rotation through+ 90o.

Rotation through+ 180o.

Rotation through+ 270o.

Reflection about mid-vertical axis.

Reflection and Rotation through- 90o.

Reflection and Rotation through- 180o.

Reflection and Rotation through- 270o.

i) Compute the scale (S) and offset (O) coefficients, using equations (2 and 3) respectively.

$$s = \frac{\left[n \cdot \left(\sum_{i=1}^{n} d_i r_i \right) - \left(\sum_{i=1}^{n} d_i \right) \left(\sum_{i=1}^{n} r_i \right) \right]}{\left[n \sum_{i=1}^{n} d_i^2 - \left(\sum_{i=1}^{n} d_i \right)^2 \right]} \quad \text{.....(2)}$$

$$o = \frac{\left[\sum_{i=1}^{n} r_i - s \cdot \sum_{i=1}^{n} d_i \right]}{n} \quad \text{.....(3)}$$

ii) Check these coefficients.

- If the coefficients of the scaling and offset not less than the minimum allowable scale/ offset value and not greater than the maximum allowable scale/ offset value then compute the root mean square error (drms) associated with the quantized

coefficients (s & o) by using equation (4).

$$R=\frac{1}{n}\left[\sum_{i=1}^{n}b_i^2+s(s\sum_{i=1}^{n}a_i^2-2\sum_{i=1}^{n}a_ib_i+2o\sum_{i=1}^{n}a_i)+o(no-2\sum_{i=1}^{n}b_i)\right] \quad \ldots(4)$$

- Else don't use these values.

iii) Compare the computed error with the minimum registered error (dmin), i.e.,

- If drms > dmin then jump to step (vi).

iv) Register

Sopt = Si, Oopt = Oi , dmin = drms, Sym = Symmetry index,

Xd =X coordinate of the current tested domain block,

Yd =Y coordinate of the current tested domain block.

v) Repeat the steps (i) to (v) for all symmetry orientations (8 cases) imposed on the same testing domain blocks.

vi) Repeat steps (i) to (vi) for all domain blocks listed in the domain pool.

vii) Output the set of parameters (Sopt, Oopt, Sym, Xd, and Yd) as a fractal coding parameters for the tested range block.

viii) Quantize these coefficients by using a uniform quantizer using equations (5 and 6 respectively). Quantization is simply the process of reducing the number of bits needed to store coefficient values by reducing its precision from float type to integer.

The quality of reconstructed image depends on the quantization process. The change of numbers from real to integer lead to lose more information. If this lost information is big, the quality of the reconstructed image will be worse. On the other hand the information is small. The quality of the image will be better and the compression ratio will be little, here we introduce (provide) new quantization operation also uniform that uses minimum number of bits and preserved the same quality.

$$quantizeScale=c\,\mathrm{int}\left(\frac{(s-\min)}{(\max-\min)}\times(2^{bit}-1)\right) \quad \ldots(5)$$

$$quantizeOffset=c\,\mathrm{int}\left(\frac{(o-\min)}{(\max-\min)}\times(2^{bit}-1)\right) \quad \ldots(6)$$

where

cint= Closest integer

bit=Number of bits allocated for quantization Scale and Offset

min=Minimum number assigned to 2bit (with minus sign)

max= Maximum number assigned to 2bit (with positive sign)

The Traditional-fractal encoding of an image implies the following output data:

1. The final partitioning (i.e., fixed, quadtree) tree of the image.
2. The scaling and offset value (si) and (oi) for each range block (i=1,2,….., No. of Range Blocks).
3. The address of the domains used to encode the ranges (i.e., the position (X, Y) for the domain block).
4. The index of symmetry operation (i.e., the orientation index) used to map the domain pixels to the range pixels.

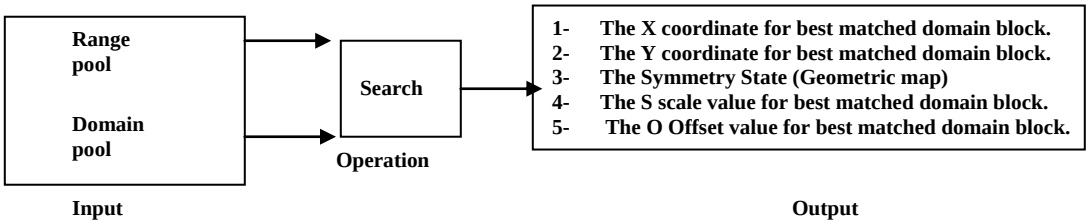

Figure (1) show the overall flow of the input/ output encoder operation

Ideally, the best results are obtained from an exhaustive search in which all domain block as well as all possible geometric maps (rotation and reflection) are tested: for each range block, the domain address(X,Y), the geometric map and gray level map coefficients (scale, offset) yielding the smallest £2 distance in equation number (5) determine the fractal code for this block.

Table (1) The encoding results of the Quadtree-fractal partitioning method applied on the Lena image with different number of bits for scale and offset.

Maximum Block Size =4 *Minimum Block Size*=2 *Inclusion Factor*=0.5 *Ratio*=0.1 *Number Of Blocks*=7114			*Minimum Scale* =-1 *Maximum Scale* =1 *Minimum Offset* = -256 *Maximum Offset* =256 *Encode Time*=**44**		
Total Number of bits	**No. of Bit Scale**	**No. of Bit Offset**	**Comp. Ratio**	**PSNR**	**SNR**
7	2	5	3.070	16.1	7.2
12	5	7	2.540	29.5	22.1
14	7	7	2.377	32.1	24.8
18	9	10	2.046	33.9	26.8
20	10	10	1.99	34.0	26.9

Number Bit Scale=2
Number Bit Offset=5
CR=3.070
PSNR=16.1
SNR=7.2

Number Bit Scale=5
Number Bit Offset=7
CR=2.540
PSNR=29.5
SNR=22.1

Number Bit Scale=7
Number Bit Offset=9
CR=2.232
PSNR=33.4
SNR=26.2

Number Bit Scale=10
Number Bit Offset=10
CR=1.99
PSNR=34.0
SNR=26.9

Fig. (2) The Lena image (256× 256) with different Number of Bits for Scale and Offset.

It can easily be deduced that when the number of bits of scale and offset is increased (large), the quality of the reconstructed image will increase and the compression ratio and bit rate will be decreased since in this case we have many (so many) range blocks and each of them required (need) more bits to represent each of them so we have low compression ratio .

New (Adaptive) Quantization / De-quantization of Fractal Coefficients

In order to improve compression (factor), we explored new (adaptive) method (way) to quantize / de-quantize the fractal coefficients (scale, offset).The suggested (proposed) method depend the positive values of scale and offset (i.e., the range of scale between 0 to 1 [0,1], for offset between 0 to 255 [0,255]).

The reason of scaling parameters , we restricted to be positive (to reduce bit rate) and less than 1 (to ensure connectivity), has been shown to be uniformly distributed except for a sharp peak at the maximum value due to clipping [7].

The proposed quantization (method) for the scaling is also (again) uniform, this method depend on the difference between successive values of

the S_i is constant and equals to 2-b2, where b2 is the number of bits allocated for the quantization of scaling.

For example, if two bits are allocated for the quantization of scaling, then the four quantized scaling should be {0.25,0.50,0.75,1.0}.

To understand how we determine (select) these values, after we determine the number of bits needed to quantize scale values (as above example 2 bits), then we have the number of possibilities equal to $2^{\text{Number of bits}}=2^2=4$, in this method we compute the amount of difference (delta)=1/Number of possibilities =1/4=0.25

So we have these values

First Possible Value=0.25

Second Possible Value=0. 50

Third Possible Value=0.75

Fourth Possible Value=1.00

Each value differ from the above by the delta value.

To quantize the Scale value into 2 bit , we have $2^2=4$ possibilities and the (amount of difference)delta=1/No. of possibilities =1/4=0.25

index	S_i
0	0.25
1	0.50
2	0.75
3	1.0

```
int CNewpartDlg::index2 (float scale)
{
float diff[4];
int flag,ci;
for(int i=0; i<4;i++){diff[i]=0.0;}
  float Sq[4];
 Sq[0]=0.25;     Sq[1]=0.50;
Sq[2]=0.75;   Sq[3]=1.00;
float min=0.0;
   ci=0;
    for( i=0; i< 4;i++)  {diff[i]=fabs(scale-
Sq[i]);}
     min=diff[0];  flag=0;
    for(i=1; i< 4;i++)
  { if(diff[i] < min) {min=diff[i];flag=i;}}
return (flag);
}
```

The question appear here, who we can (use) benefit from this method describe above, after computed scale value in traditional method for the first range block with all domain block, suppose the scale value is equal to S=0.32 and we quantize into two bits, then we apply this method of successive difference , we found that S_i=0.25 and the index=0 is closest to this value, instead of saving (storing) this value we save (store) only the index, then

SQuantized= Index of the closest value to the Scale……..(7)

In the de-quantization operation, we depend on this rule

SDequantized=(Index of the closest value to the Scale+1)*delta…….(8)

By applying this rule on the example above ,we reach Sdequantized =(0+1)*0.25=0.25

Figure (3) illustrate the index and S_i possible values for three and four bit respectively.

To quantize the Scale value into 3 bit , we have $2^3=8$ possibilities and the (amount of difference)delta=1/No. of possibilities =1/8=01.25

index	Si
0	0.125
1	0.25
2	0.375
3	0.5
4	0.625
5	0.75
6	0.825
7	1.0

To quantize the Scale value into 4 bit , we have 24=16 possibilities and the (amount of difference)delta=1/ No. of possibilities =1/16=01.25

index	Si
0	0.0626
1	0.125
2	0.1875
3	0.250
4	0.3125
5	0.375
6	0.4375
7	0.500
8	0.5625
9	0.625
10	0.6875
11	0.750
12	0.8125
13	0.875
14	0.9375
15	1.0

Figure (3) the index and S_i possible values for 3 and 4 bit respectively

The quantized of offset is more simple, at the beginning the need of restriction of offset value to be positive since the gray level image is between 0 to 255 and assigned the number of bit of quantization between 1 bit to 8 bit (since the image value form0 to 255). The proposed (suggested) quantization method again uniform, depend on the quantize the offset value by dividing the offset value on the number of bits (i.e., the offset value we computed in the encoder after we specify the number of bits we need to quantize this value on it we only need to divide the offset value at he number of bits) and the de -quantize of offset value is multiply the quantized offset value by the number of bits determined earlier in the quantization, in other words,

OQuantized=Offset value /number of bits needed to quantize this value……..(9)

ODequantized= OQuantized * number of bits needed to quantize this value…..(10)

Table 2,and figure 4 illustrate the result of the new method of quantization and de-quantization applied on the fractal coefficients.

Table (2) The encoding results of the Quadtree-fractal partitioning method applied on the Lena image with different number of bits for scale and offset using the new (adaptive) method.

Maximum Block Size =4 *Minimum Block Size*=2 *Inclusion Factor*=0.5 *Ratio*=0.1 *Number Of Blocks*=7114			*Minimum Scale* =0 *Maximum Scale* =1 *Minimum Offset = 0* *Maximum Offset* =256 *Encode Time*=**44**		
Total Number of bits	**No. of Bit Scale**	**No. of Bit Offset**	**Comp. Ratio**	**PSNR**	**SNR**
4	2	2	3.50	32.6	25.4
8	3	5	2.94	32.6	25.6
8	4	4	2.94	33.0	25.8
9	5	4	2.83	33.2	26.2

Number bit Scale=2
Number bit Offset=2
CR=3.50
PSNR=32.6
SNR=25.4

Number bit Scale=3
Number bit Offset=5
CR=2.94
PSNR=32.6
SNR=25.6

Number bit Scale=4
Number bit Offset=4
CR=2.94
PSNR=33.0
SNR=25.8

Number bit Scale=5
Number bit Offset=4
CR=2.83
PSNR=33.2
SNR=26.2

Fig. (4) The Lena image (256× 256) with different Number of Bits for Scale and Offset based on new method.

It can easily be deduced that the proposed method gave better results that show when we use more bits we produce good quality image but in the proposed method we notice we require a less amount of bits since the concept we depend on it differ form the traditional method.

Discussion and Conclusion

In the paper, we developed a new method of quantization/de-quantization fractal parameters (scale, offset). We prove good results of the new method, study the essence of the new method and show the new method advantages over traditional methods. We propose new

strategy that improve the compression ratio and PSNR by exploit the difference of the original value of scale and the closest value depend on the number of bit used then we store the index value that be restricted to be integer and for offset value we use an equation to minimize (decrease) the value so we require a less a number of bits, therefore we gain by using minimum number of bits we have a high image quality and compression ratio.

This paper represent an attempt to replace (change) the steps of encoder and decoder system, since in the old system we managed (treat) a quantization as a separate step after encoder operation and de-quantization managed (treat) as a separate step before the decoding operation, this paper provide (introduce) new way (method) to treat these steps (quantization, de-quantization)concurrent(simultaneously) with encoder and decoder, figure 5a,b illustrate this concept

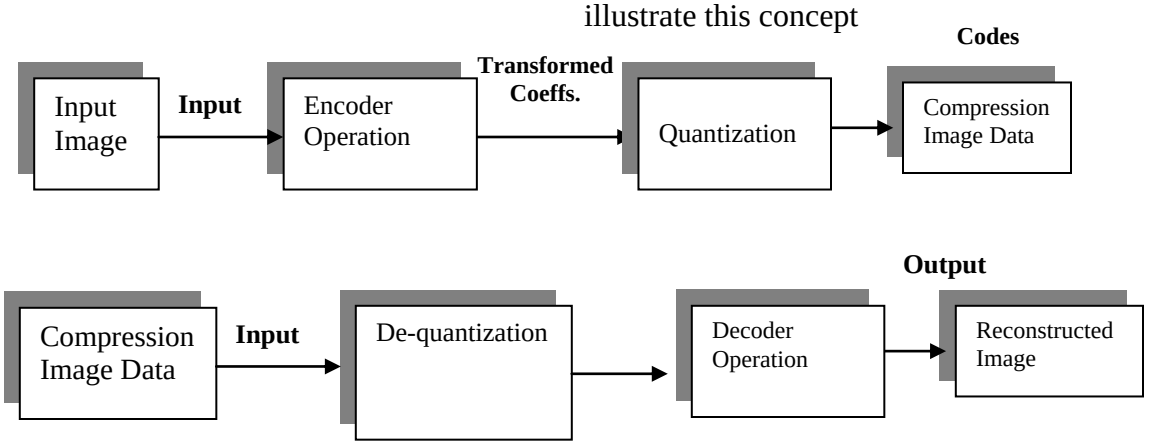

Fig (5.a) old Encoder/decoder system

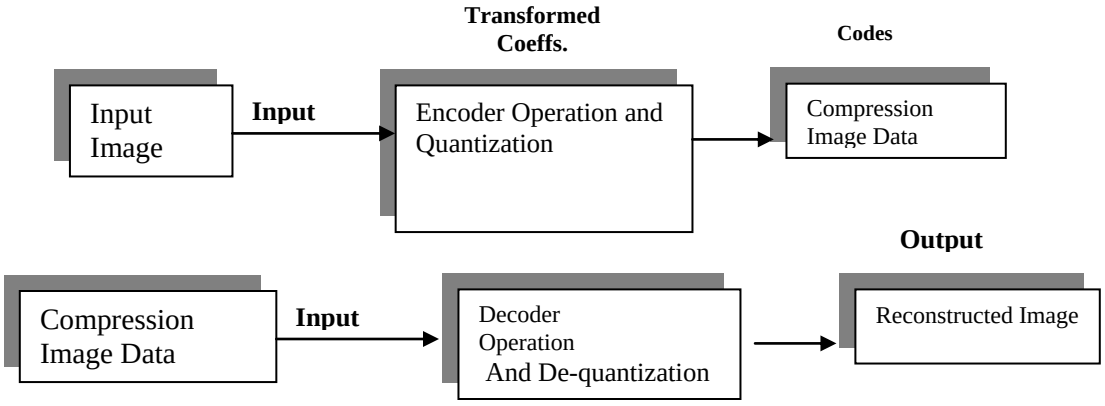

Fig (5.b) new Encoder/decoder system

References

1. Barnsley, M. F., and Hurd, L.P.,1993, Fractal Image Compression, Wellesley, MA: A.K. Peters.

2. Fisher, Y., 1995, Fractal Image Compression Theory and Application, New York: Springer Verlag.

3. Lu, N., 1997, Fractal Imaging Academic Press, New York.

4. Fisher, Y., 1998, Fractal Image Encoding and Analysis, Springer Verlag, Heidelberg.

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

6. Hamzaoui, R., Kopilovic, I., Saupe, D., Hartenstein, H., Nguyen, G., and Ruhi, M., 1996, Fractal Image Compression, Proc. ICIP-96 IEEE International Conference on Image Processing, Lausanne.

7. Jacquin, A., 1992, Image coding based on a fractal theory of iterated contractive image transformations, IEEE Trans. Image Proc., vol. 1, no. 1, pp. 18-30.

تحسين ضغط الصور باستخدام الكسوريات عن طريق عملية التكميم

غادة كاظم طعمة* **حيدر حميد رزاق ****

* مدرس مساعد /كلية العلوم /وحدة التحسس النائي .
** مدرس مساعد / كلية العلوم /قسم علوم الحاسبات /الجامعة المستنصرية.

الخلاصـــة:

قمنا بدراسة أولية (تمهيدية) لمعاملات التحويلات الخاصة بالكسوريات وقمنا استخدام (تطوير) طريقة جديدة لتحسين نسبة (قدرة) الضغط لهذة الطريقة . في اغلب أنظمة التشفير/ حل الشفرة القياسية عملية التكميم / حل (فتح) التكميم تعامل (تعالج) كخطوة منفصلة (مستقلة) هنا في هذة الطريقة الجديدة تمت معالجة عملية التكميم/ حل (فتح) التكميم في أثناء عملية التشفير / حل الشفرة في وقت واحد.كما ستلاحظ لاحقا أن نسبة ضغط إضافية ممكن إنجازها باستخدام هذة الطريقة مع المحافظة على نوعية الصورة الناتجة.