Fast Fractal Image Encoder

Yung-Gi, Wu

Department of Computer Science & Information Engineering
Leader University, Tainan, Taiwan

Email: wyg@mail.leader.edu.tw

ABSTRACT

Although fractal image compression can achieve high compression ratio theoretically, it needs a lot of encoding time to encode an image so that it has not been widely applied as other coding schemes in the field of image compression. In this paper, an algorithm is devised to improve this drawback. The algorithm uses three major processes including classification, PDS (Partial distance search) and simplification of eight transformations to decrease the encoding time. Classification is the major factor to reduce the majority of encoding time. PDS decreases computation time of MSE (Mean Square Error). Simplification of eight transformations diminishes unnecessary computation. The experimental result shows that our proposed method makes the encoder much faster than the conventional fractal compression method and the quality is imperceptible to the conventional fractal-encoding algorithm. Compared to the published fast fractal encoding algorithms, the proposed method outperforms them. This paper contributes to the performance of source signal compression before the communication and raises the effectiveness of multimedia system.

Key words: fast fractal encoding; image compression; partial distance search; classification; simplification;

1. INTRODUCTION

A picture may be worth a thousand words, but it requires far more memory to store or bandwidth to transmit. With the successes of multimedia technology and the era of wideband network, peoples' desire to the high quality of multimedia still can not be satisfied. All the scholars in the universities and the engineers in the industry want to get the compromise between the limited network bandwidth and the unlimited human desire. Among all the digitized data that we people can touch every day, such as digital library, VCD, DVD, JPEG, etc, the kernel of the system or the standard that commercial products use is the compression technique. There are many coding schemes that have been developed such as DCT, VQ, Wavelets, BTC, Fractal, etc. In general, the criteria to evaluate the performance of a compression system include 1) compression ratio 2) reconstructed quality 3) processing time. Fractal compression can achieve the highest compression ratio among all the existing coding schemes theoretically; however, its encoding time is terrible intolerant to practical industrial applications. If this shortcoming of long encoding time can be improved, the application of fractal will be a practical consideration.

The cause of fractal image coding with high compression is that the minority of blocks through rotations represent the majority of blocks[1]. In a word, fractal encoding is based on Partitioned

Iterated Function System (PIFS). The detailed descriptions of PIFS can be found in [2-5]. There are some published papers concerning about the fast fractal encoding [6-12]. Some of them use variance for classification to improve the drawback of time consuming in encoding process. In the proposed method, we utilize classification, simplification of transformation and partial distance searching strategy to pruning those blocks whose characteristics do not match the range block to be processed and reduce unnecessary computation on best matching computation.

The remaining sections are organized as follows: Section 2 will depict the basic fractal image coding. Proposed method is given in Section 3 and results will be shown in Section 4. Conclusion of this paper is in Section 5.

2. BASIC FRACTAL IMAGE CODING

Let an original image be partitioned into non-overlapping regions called range blocks (*R*) and overlapping regions called domains blocks (*D*). The size of each domain block should be larger than that of the range block to satisfy the property of contraction [2]. Let *D'* denote the down sampled domain block of *D* and the *D'* size is equal to the size of *R*. The transformations are composed of a geometric transformation and a massic transformation. The geometric transformation consists of moving the domain block to the location of the range block and adjusting the size of domain block to match the size of range block. The massic transformation adjusts the intensity and orientation of the pixels in the domain block after it has been operated on by the geometric transformation. The geometric and massic transformation t_i can be depicted as follows:

$$t_i\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} A_i & B_i & 0 \\ M_i & N_i & 0 \\ 0 & 0 & s_i \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} E_i \\ F_i \\ o_i \end{bmatrix} \qquad (1)$$

where s_i controls the contrast and o_i controls the brightness. $z = f(x, y)$ is the gray level value at (x, y) and A_i, B_i, M_i, N_i can be used to denote the eight-symmetry such as **[1]**. Identity mapping [**2**] Rotation by 90 degrees **[3]** Rotation by 180 degrees **[4]** Rotation through -90 degrees **[5]** Reflection about mid-vertical axis **[6]** Reflection about mid-horizontal axis **[7]** Reflection about diagonal **[8]** Reflection about cross diagonal. E_i and F_i are used for position offset. The *i* in s_i and o_i denotes one of the above mentioned eight symmetries.

In practice, we compare a range block and down sampled domain blocks using *MSE* metric as follows

$$MSE = \sum_{k=1}^{n\times n}\left(s_i \times a_k + o_i - b_k\right)^2 \qquad (2)$$

Where a_k represents the pixel value of the sampled domain blocks (*D'*) after eight transformations and b_k represents the pixel value of the range blocks and the block size for both *R* and *D'* is *n* by *n*. This *MSE* metric allows easy computation for optimal values of s_i and o_i in equation (1). This will give us contrast and brightness settings that make the affine transformed a_k values have the least squared distance from the b_k values. The minimum of *MSE* occurs when the partial derivatives with respect to s_i and o_i are zero, which occurs when

$$s_i = \frac{n\times n\left(\sum_{k=1}^{n\times n} a_k b_k\right) - \left(\sum_{k=1}^{n\times n} a_k\right)\left(\sum_{k=1}^{n\times n} b_k\right)}{n\times n\sum_{k=1}^{n\times n} a_k^2 - \left(\sum_{k=1}^{n\times n} a_k\right)^2} \qquad (3)$$

$$O_i = \frac{\sum_{k=1}^{n\times n} b_k - S_i \sum_{k=1}^{n\times n} a_k}{n\times n} \qquad (4)$$

There are many best matching criteria to choose. The *MSE* is usually used in fractal image coding and the minimal *MSE* always denotes better matching. We use equation (2) to find the optimal s_i and o_i and then quantize them for storage or transmission. In addition, the encoder must record the position of the best matched domain block (*D'*) and its transformation for each range block so as to reconstruct the decoded block on the decoder side. Suppose the data to be dealt with is 512 × 512 pixel image in which each pixel can be one of the 256 levels of gray(ranging from black to white). Let R_i be the 8x8 pixel non-overlapping range block (i=1,..,4096) and let D be the collection of all the 16×16 overlapped sub-squares of the image. The collection of D contains 497×497=247009 squares when we shift the position of D with one pixel at one step. For each R_i, search through all of collection of D_i to find the one which minimizes the *MSE* as equation (2); that is, find the part of the image that most looks like the image above R. There are 8 ways to map one square onto another, so that this means comparing 8×247009=1976072 squares with each of the 4096 range blocks. In addition, we must fulfill the down-sampling operation for each D_i to get the same size of R to carry out the later *MSE* computation. Choosing 1 from each 2×2 sub-square of D_i or averaging the 2×2 sub-square corresponding to each pixel of R can achieve the goal of down-sampling. It is obvious that the huge computation is needed from the above descriptions about the conventional fractal encoding. The time to search the best matched domain block for every range block is a time consuming job in practical application. Therefore, we develop a new encoding algorithm to reduce the time in this research. A lot of people have been making efforts in fractal improvement. Some investigate region-based image coding methods and some combine fractal with other algorithm such as wavelet in [6], genetic algorithms in [7], discrete cosine transform in [8] [9]. Saupe and Jacob employ a variance condition to decide whether or not to quadtree partition a block further [10]. In [11], C.K. Lee and W.K. Lee use the variance matching technique. The best matched domain block is searched within the searching window for in the neighborhood of the domain block with the closet variance to that of the range block. [12] uses a non-symmetric window to search the for the best matched domain block based on the local variance method. Almost all of them used the characteristic of image content to pruning the unnecessary computation to decrease time.

3. PROPOSED CODING SCHEME

In the proposed encoding algorithm, classification and transformation simplification are the two major contributions to decrease the encoding time. The ideas are intuitively due to the following facts. The first one is that it is unnecessary for a “complicated” range block to waste time to search the “pure” blocks in the domain pool. The second one is that the eight transformations can also be simplified so that the encoder does not have to calculate so many transformations to find its best matched domain block for each range block during the calculation of *MSE* metric. Detailed descriptions are given in the following sections.

3.1 Classification by Variance

In this paper, the block variance is the criterion to classify. The variance is usually used to classify the simplicity or complexity of block. The variance of block I is defined as

$$Var(I) = \frac{1}{n\times n}\sum_{i=1}^{n}\sum_{j=1}^{n}(x_{ij} - u(I))^2 \qquad (5)$$

where $n \times n$ is the size of the block and x_{ij} is the pixel value of the block I. *u(I)* is the mean value of the block *I*. Searching area is determined according to the variance difference between $Var(R)$ and $Var(D')$. Both R and D' are classified into a number of classes. For every *R*, it only searches the D's whose variances are close or adjacent to the class of R to reduce the searching time. Refer to Fig. 1, range block of No.2 (R_2) is an example. If the threshold is set to 20, R_2 will select *D's* that meet the criteria of $|Var(R_2)$ - $Var(D')| \leqq 20$. Therefore, *D*'s of No.2, No. 4 , No. 6, No.7 and No.9 will be selected for the next processing. The variance threshold has serious effect on the performance of proposed method. It is certain that the reduction of searched D's amount decreases the decoded quality as well. However, the searching time decreased dramatically while the degradation of quality is almost visual undistinguishable.

$|Var(R) - Var(D')| \leq 20$

No.	Var(R)	Mean(R)	Var(D')	Mean(D')
1	10	15	25	12
2	***50***	30	***40***	15
3	20	20	15	65
4	40	80	***45***	20
5	80	45	08	43
6	90	70	***70***	62
7	45	50	***30***	45
8			90	100
9			***60***	80

Fig. 1 Variance difference between $Var(R)$ and $Var(D')$

3.2 Simplification of Eight-Transformation

Conventional fractal encoding fulfills eight transformations for each D' to find the best matched one for every R. When the image size is 512×512, R is 8×8 and D is 16×16, eight transformations yield eight sets of (s_i, o_i). Conventional computed number to implement *MSE* is 247009×8 if we shift the domain block one pixel every time. This is the fatal factor making the long encoding time. Reducing the transformation number can decrease the total computation time effectively. We find some regulations can be used to decrease eight transformations by observing rotation of different block patterns. First, every D' is divided into four sub-blocks and then calculate the mean value of every sub-block. The mean value of each sub-block is used to generate the pattern of each D'. There are four sub-blocks within each D' so that there are total twenty-four possible patterns. Some of them do not need to make eight transformations after the following analysis.

P1	P2
P3	P4

Fig. 2 Four blocks

The mean values of the four sub-blocks are labeled as P1, P2, P3, and P4 as can be seen in Fig. 2. The simplified transformation algorithm can decrease eight transformations into four classes as following:

Class1: If P1=P2=P3=P4, it only makes one transformation. (Refer to Fig. 3)
Class2: If P1=P4 and P2=P3, it makes two transformations. (Refer to Fig. 4)
Class3:
(a) If P1=P2=P3 or P1=P2=P4 or P2=P3=P4 or P1=P3=P4, it makes four transformations. (Refer to Fig. 5)
(b) If P1=P2 and P3=P4, it makes four transformations. (Refer to Fig. 6)
(c) If P1=P3 and P2=P4, it makes four transformations. (Refer to Fig. 7)
(d) If P1=P4 and P2≠P3, it makes four transformations. (Refer to Fig. 8)
(e) If P2=P3 and P1≠P4, it makes four transformations. (Refer to Fig. 9)
Class4: If it doesn't belong to class1, class2, class3, it must make eight transformations.

Notice that "=" or "≠" is determined by the differences among P1 to P4 in the previous algorithm. If the difference between PX and PY is lower than a threshold, we regard PX=PY. Otherwise, PX≠PY.

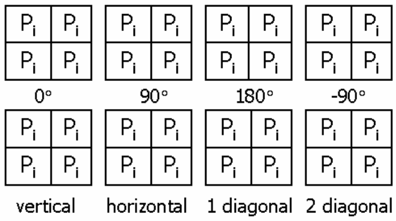

Fig. 3 Class 1

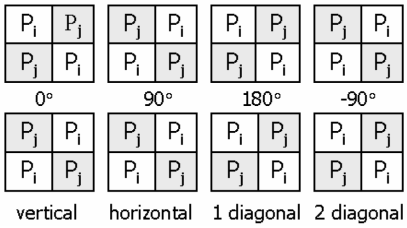

Fig. 4 Class 2

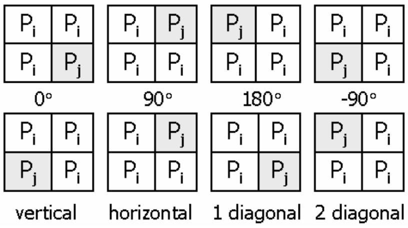

Fig. 5 Class 3 (a) P1=P2=P3

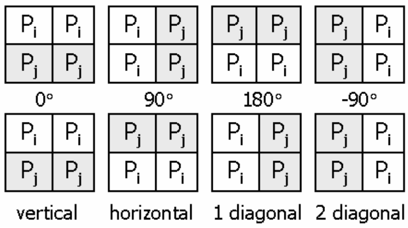

Fig. 6 Class 3 (b)

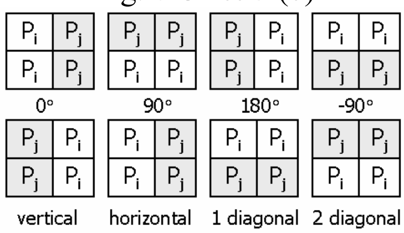

Fig. 7 Class 3 (c)

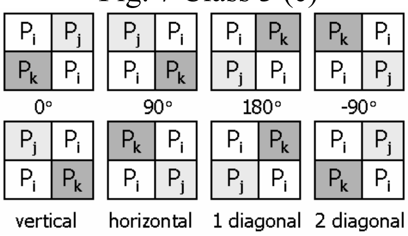

Fig. 8 Class 3 (d)

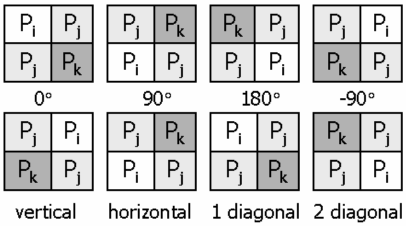

Fig. 9 Class 3 (e)

3.3 Computation Reduction of *MSE*

Previous section 3.1 depicts the algorithm to rule out the domain blocks whose variance and mean value do not close to the input range block to decrease the number of *MSE* computation by only calculating a portion of *D*'s instead of the whole block. Section 3.2 depicts the rule to decrease the number of eight symmetries. This section describes another methodology to decrease the *MSE* computation furthermore.

For every selected domain block, we must compute the error between the range block and the eight-transformed domain blocks to find the best matched one whose error is minimum. We use a method to reduce the computation of *MSE* metric. This algorithm is called as Partial Distance Search (PDS) which was devised in 1991 for the application of fast Vector Quantization (VQ) encoding [13]. The PDS algorithm discards all the domain blocks whose partial distortion relative to an input vector exceeds the current available minimum distortion during the calculation of *MSE*. The operation of the PDS algorithm is summarized as follows:

Input: Range block (R).
Selected sampled domain blocks D_i' ($i=1..z$). Notice that the number of total domain block is z .
Output: Best matched *D'* and its best isometric.
Step1: find s_i and o_i for first D'

i=0;

$$MSE = \sum_{k=1}^{n}\sum_{l=1}^{n}(D_1^{'}(k,l)\times s_i + o_i - R(k,l))^2 \qquad (6)$$

current_error =*MSE*;
best_D'=1;
best_isometric=0;

Step 2:
for(p=1;p<=z; p++)   // p is D' index
{
  Find s_i and o_i by equation (3)(4) for every D' ;
  if(p==1) a=1; else a=0;
      // the *MSE* of first D' and its first isometric has calculated already in Step 1;
  for(i=a; i<8;i++)
  {
    MSE=0;
    for(k=1; k<=n; k++)
    for(l=1; l<=n; l++)
    {

$$MSE += (D_i^{'}(k,l)\times s_i + o_i - R(k,l))^2 \qquad (7)$$

      if (*MSE*>=current_error) /* it exceeds the current minimum error */
        goto exit; /* pre-mature exit */
    }
    if(*MSE*<current_error)
    {
      current_error=*MSE*;

```
                best_D'=p;
                best_isometric=i;
            }
            exit: { };
        }
    }
    Return (best_D', best_isometric);
```

In general case, without the simplification of eight symmetries as given in the previous section, every D_i' has eight-symmetry forms and the number of D_i' (i=1..z) to be compared to every R is z; thus, the total computation of *MSE* is zx8. Step 1 calculates the *MSE* error between R and the first isometric of D_1'. Step 2 calculates all the other *MSE* including the other isometrics of D_1' and the isometrics of all the D_2' ~ D_z'; meanwhile, it compares the distortion between the current minimum error and increasing *MSE* in (7). Once the increasing *MSE* exceeds the current minimum error, it stops the *MSE* calculation and exits to the outer of the loop. Such a method can decrease the heavy computation of *MSE* effectively. We call this *MSE* as *PDS-MSE*.

3.4 Procedures of the Proposed Algorithm

The main purpose is to use the variance value to pruning those D's that are inconsistent to the R. Then, using the computation reduction of *MSE* and simplified transformation to find the best matched D'.

The steps of the proposed algorithm are given as follows.

Step 1: Partition the original image into non-overlapping range blocks (R) and overlapping domain blocks (D).

Step 2: Down-sample domain blocks ($D \rightarrow D'$) to the same size as R.

Step 3: Calculate variance of R and D'.

Step 4: For each R, select those D's that meet the criterion of $|Var(R) - Var(D')| \leq$ threshold.

Step 5: For all the selected D', using the simplified transformation analysis to decrease the 8 transformation.

Step 6: Fulfill *PDS-MSE* computation to find the best matched D' and its transformation. Then, store this position of searched D', mapped isometric, $\boldsymbol{s_i}$ and $\boldsymbol{o_i}$.

Step 7: Repeat Step 4-6 until all the Rs are processed.

4. EXPERIMENTAL RESULTS

The proposed fast fractal encoding is simulated using several *512×512* images with *256* gray levels. The conventional algorithm uses non-overlapping 8×8 range block and overlapping 16×16 domain block and full exhaustive search. For the purpose of performance comparison, the sizes for range and domain blocks are the same as the conventional scheme in the proposed experiment. The thresholds of variance difference for test images are given in Table 1. We got those thresholds from empirical experience and we find that setting the threshold to 40 can get the tradeoff between decoded quality and encoding time. The numbers of D's in each class through the simplified eight transformations algorithm are listed in Table 2. All of the above methods have been coded in C language and running on Pentium 4- 2.4 gigahertz CPU and main memory is 512MB. Windows XP is the operating system and the programming language is Visual C++ 6.0. PSNR is used to measure the quality of the decoding image.

Table 1 The thresholds for test images

Images	Lena	Zelda	Pepper
Threshold	40	36	48

Table 2 The number of D' after simplified eight transformations

512×512	Class 1	Class 2	Class 3	Class 4
Lena	322	495	64479	181713
Zelda	217	210	50922	195660
Pepper	369	423	62314	183903

Refer to Fig. 10 to Fig. 12, there are three different nature images including Lena, Zelda and Pepper used for visual evaluation. Conventional fractal encoding needs about 7 hours to encode them under the same hardware circumstance as we use; however, the proposed method only needs a little more than one minute. Note that the proposed method moves the domain block by shifting single-pixel manner in the simulations of Fig.10 to Fig. 12.

(a) Original image

(b) Traditional fractal decoding image
PSNR=30.128 dB. Time=21456 sec

(c) Proposed fractal decoding image
PSNR=29.27 dB. Time=42 sec

Fig. 10 Performance of Lena

(a) Original image

(b) Traditional fractal decoding image
PSNR=34.6999 dB. Time=21543 sec

(c) Proposed fractal decoding image
PSNR=33.45 dB. Time=41 sec

Fig. 11 Performance of Zelda

(a) Original image

(b) Traditional fractal decoding image
PSNR=30.9105 dB Time=21532 sec

(c) Proposed fractal decoding image
PSNR=29.88 dB. Time=43 sec

Fig. 12 Performance of Pepper

Because there are three major methods contribute to the performance, we also show the experimental results used for the performance evaluation of classification, simplification of eight transformation and PDS in Table 3. (A) denotes traditional fractal encoding method, (B) shows the encoding method with the variance classification, (B)+(C) shows the encoding method with the variance classification and the simplification of eight transformations but without *PDS-MSE*, (B)+(C)+(D) shows proposed complete encoding method including the variance classification, simplification of eight transformations and *PDS-MSE*. The decoded quality on the decoder side is expressed by PSNR in Table 3. The decoded quality by the proposed method is worse than conventional full search strategy. But, the visual degradation is not noticeable.

Table 3 Experimental results

Image	Method	PSNR (dB)	Encoding time (sec)	Numbers of MSE Computation
Lena	(A)	30.12	21456	1976072
	(B)	29.85	93	4856
	(B)+(C)	29.27	72	3634
	(B)+(C)+(D)	29.27	42	3634
Zelda	(A)	34.69	21543	1976072
	(B)	33.85	88	4034
	(B)+(C)	33.45	68	3221
	(B)+(C)+(D)	33.45	41	3221
Pepper	(A)	30.91	21532	1976072
	(B)	30.43	86	4356
	(B)+(C)	29.88	67	3356
	(B)+(C)+(D)	29.88	43	3356

Note that *PDS-MSE* does not increase any PSNR degradation while reducing encoding time. The data within the column of "number of MSE computation" in Table 3 denotes the numbers of fulfilling *MSE* for each range block. *PDS-MSE* does not decrease the number of *MSE* but it decreases the computation burden during the computation. If we move the domain blocks by shifting 4 pixels, the time for encoding Lena, Zelda and Pepper are 2.83 seconds, 2.67 seconds and 2.93 seconds, respectively. The PSNR values for the three images are 28.87 dB, 32.46dB and 29.05 dB.

In [11], it achieves speedup by using a symmetry window searching for the best matched domain blocks for each range block. The best matched domain block is searched within the searching window for in the neighborhood of the domain block with the closet variance to that of the range block. [12] a three-level quadtree partition scheme with range block of 4×4, 8×8 and 16×16 with different error thresholds for the quadtree splitting process and a domain grid of two are used. It uses a non-symmetric window to search the for the best matched domain block based on the local variance method. In the above two methods, the size of searching window influences the performance of encoding time and quality. The results of [12] are better than [11]. We have implemented the algorithm of [12] and running it on the same hardware and platform as we use. The window sizes being used in [12] are from 4.7% to 9.4%. But, it does not consider the simplification of transformation and the other factors that are contributed to the speedup. In the proposed method, the numbers of searched block are greatly decreased because of that we use variance to prune the number of domain blocks. And the simplification of eight transforms also reduces the numbers of MSE computation. The numbers of MSE computation by the proposed method are from about 0.15% to 0.18% compared to the conventional full search fractal encoding in out test images. We modify the parameter of classification threshold, shifting pixel of domain block, and the bit-allocation for the fractal codes in the proposed system to make the compression ratio close to [11] and [12]. The performance comparison among the three methods is listed in Table 4.

Table 4 Performance comparison

	Method	Compression ratio	PSNR (dB)	Encoding time (sec)
Lena	Proposed method	21.56	28.21	1.72
	[11]	20.32	32.12	101
	[12]	20.05	32.08	52
Pepper	Proposed method	24.32	29.01	2.01
	[11]	22.77	31.87	198
	[12]	23.13	31.84	48

5.CONCLUSION

In this paper, the performance of the traditional image coding system in terms of speed is greatly improved, which can raise the performance of coding system. The experimental results show that our proposed method makes the encoder much faster than the conventional fractal compression method. Compared to other published methods [6-12], the proposed method gets better performance in terms of running time. We proposed a composite method of speeding up fractal image coding. There are major three ways to achieve the speed-up (1). Pruning the domain pools using a variance condition (variance of parent block should not be too different from the variance of the child block). (2). Pruning the isometrics being considered, based upon a classification of blocks based on local mean of block. (3). PDS-MSE to monitor the partial sums of the collage error as they are being computed, and exiting the loop as soon as the partial sum exceeds the minimum collage error encountered to date. Variance classification is the major contribution to reduce the majority of encoding time and control the quality of the decoded images. The classification by variance can be changed by adjusting the thresholds. Modifying the classification thresholds, different performance will be yielded to meet requirement. Simplification of eight transformations also diminishes the unnecessary number of transformation computation burden. PDS helps us to decrease *MSE* computation. Our proposed fractal encoding method improves the drawback of conventional fractal image coding in the cost of imperceptible quality. Notice that the proposed method moves the domain blocks by shifting single-pixel manner. If we move the domain block more than one pixel every time, the efficiency with respect to time will be raised. However; the decoded quality will be worse certainly.

ACKNOWLEGEMENT

This work was supported in part by the National Science Council of Taiwan under Contract No. NSC 95-2221-E- 426-004.

REFERENCE

[1] M.F. Barnsley. *Fractal Everywhere*. Academic Press, San Diego, 1988.
[2] Y. Fisher, *Fractal Image Compression. Theory and Application*, Springer-Verlag, New York 1995.
[3] A.E. Jacquin, “Fractal image coding: a review”, *Proc. of the IEEE*, vol. 81, NO.10, pp.1451-1456, 1993.
[4] A.E. Jacquin, “Image coding based on a fractal theory of iterated contractive image transformations,” *IEEE Trans. Image Process.* Vol.1, NO.1, pp.18-30, 1992.
[5] B. Wohlberg and G.D. Jager, “A review of the fractal image coding literature,” *IEEE Trans. Image Process.* Vol.8, NO.12, pp.1716-1729, 1999.
[6] G.M. Davis, “A wavelet-based analysis of fractal image compression,”*IEEE Trans. Image Process.* , vol. 7, NO.2, pp.141 -154, 1998.
[7] M. Takezawa, H. Honda, J. Miura, H. Haseyama and H. Kitajima,“A genetic-algorithm based quantization method for fractal image coding,” *in Proc. of IEEE International Conf. on Image Process*. Vol. 1, pp.458-461, Kobe, Japan, 1999.
[8] Y. Zhao and B. Yuan,“Image compression using fractals and discrete cosine transform,” *Electron. Lett*. vol.30, NO.6, pp.474–475, 1994.
[9] Y. Zhao and B. Yuan, “A hybrid image compression scheme combining block-based fractal coding and DCT,” *Signal Processing: Image Communication.* Vol.8, NO.2, pp.73-78, 1996.

[10] D. Saupe and S. Jacob, "Variance-based quad-trees in fractal image compression," *Electron. Lett.* Vol.33, NO.1, pp.46–48, 1997.
[11] C.K. Lee and W.K. Lee, "Fast fractal image block coding based on local variances," *IEEE Trans. Image Process*. Vol.7, NO.6, pp.888 –891, 1998.
[12] C.M. Lai, K. M. Lam and W. C. Siu, "Improved searching scheme for fractal image coding," Electronic Lett. Vol. 38, NO.25, pp.1653-1654, 2002.
[13] J. Ngwa-Ndifor and T. Ellis, "Predictive partial search algorithm for vector quantization," *Electron. Lett.* Vol.27, NO.19, pp.1722-1723, 1991.

Yung-Gi, Wu received his BS in information and computer engineering from Chung Yuan Christian University, Chung-Li, Taiwan, in 1992, and his MS and PhD in electrical engineering from the National Cheng Kung University, Tainan, Taiwan, in 1994 and 2000, respectively. From 1994 to1996, he was a Reserve Officers' Training Corps (ROTC) officer in the army. He was an assistant professor in the Department of Computer Science and Information Engineering and the Institute of Applied Information, Leader University, Tainan, Taiwan from 2000 to 2003. He is current an associate professor since December, 2003. His research interests include image database indexing, image processing, and nonlinear optimization.