Fast Fractal Image Compression Based on Domain-Range Pixel Value Difference

Venkata Rama Prasad Vaddella[1]

Ramesh Babu[2]

GJCST Classification
E.4. I.4.2. I.4.M

***Abstract*-Fractal image Compression is a lossy compression technique that has been developed in the early 1990s. It makes use of the local self similarity property existing in an image and finds a contractive mapping affine transformation (fractal transform) T, such that the fixed point of T is close to the given image in a suitable metric. It has generated much interest due to its promise of high compression ratios with good decompression quality. The other advantage is its multiresolution property, i.e. an image can be decoded at higher or lower resolutions than the original without much degradation in quality. However, the encoding time is computationally intensive. In this paper, a new method to reduce the encoding time based on computing the pixel value difference of domain and range blocks is presented. A comparison for best match is performed between the domain and range blocks only if the range block pixel value difference is less than the domain block pixel value difference. This reduces the number of comparisons, and thereby the encoding time considerably, while obtaining good fidelity and compression ratio for the decoded image. Experimental results on standard gray scale images (512x512, 8 bit) proved that the proposed method improved in performance when compared to conventional fractal encoding.**

Keywords-Fractal image compression, pixel value difference, adaptive scaling, classification.

I. INTRODUCTION

The basic scheme of fractal image compression is to partition a given image into non overlapping blocks of size rxr, called range blocks and form a domain pool containing all of possible overlapped blocks of size 2rx2r, called domain blocks associated with 8 isometries from reflections and rotations [19]. For each range block, it exhaustively searches, the domain pool, for a best-matched domain block with the minimum rms error after applying a contractive affine transform to the domain block. A fractal-compressed code for a range block consists of quantized contractivity coefficients in the affine transform, a luminance offset, the position of the best-matched domain block and its isometry. The decoding is to find the fixed point, the decoded image, by starting with any initial image. The procedure applies a compressed local affine transform on the domain block corresponding to the position of a range block until all of the decoded range blocks are obtained. The procedure is repeated iteratively until it converges. The problems that occur in fractal encoding are the computational demands and the existence of best range-domain matches [20]. The most attractive property is the resolution-independent decoding property. The image can be decoded at an enlarged size so that the compression ratio may increase exponentially [18]. However searching the domain pool is highly computationally intensive. For an nxn image, the number of range blocks are (nxn/rxr) and the number of domain blocks are (n-2r+1) x (n-2r+1). The computation of best match between a range block and a domain block is O (r2). If r is constant, the computation complexity of entire search is O (n4).

Yuval Fisher [18] proposed the quad tree-partitioning algorithm for fractal image compression. In this algorithm, the range blocks and domain blocks are classified in to 3 major classes based on the average of the pixels in four quadrants of the blocks. These are further divided in to 24 sub classes (! 4) based on the variance of the pixels in the four quadrants. Thus, the domains and ranges are classified in to a total of 72 classes. This algorithm is called classified search algorithm, as the domains and ranges belonging to the same class only are compared. But due to the large number of domains, the encoding time is very high. One of the simplest ways of decreasing coding time is to reduce the size of the domain pool. This is achieved by a spatial constraint on the domain pool for each range to which it is mapped [20]. Noting that a contractive mapping requires a domain with a higher variance than the range, domains with low variance may be excluded from the domain pool [5]. Alternatively, domain pools may be pruned in order to eliminate domains, which have similar invariant representations to other domains in the pool [15]. During the last decade several researchers have proposed methods to reduce the size of the domain pool based on various split decision functions [11]. The variance feature has been used [4,5,12] as a decision function by many researchers for domain pool reduction. Recently, the entropy function has also been reported as a split decision function [2] to reduce the domain pool. Tomas Zumbakis and Jonas Valantinas [23] have proposed an approach to improve the encoding times based on the classification of the range and domains based on their smoothness estimates in the frequency domain. Daniel Riccio and Michele Nappi [1] proposed a method for reduction of the encoding time by deferring the range and domain comparisons with respect to a preset block. In this paper, we present a new method for reducing the encoding times based on computing the pixel value

About[1]Professor of Information Technology, Sree Vidyanikethan Engineering College, A.Rangampet – 517 102, Near TIRUPATI, (A.P), INDIA(e-mail: vvramaprasad@rediffmail.com)
About[2] Professor of Computer Science and Engineering, Nagarjuna University, GUNTUR – 522 510, (A.P), INDIA

difference of the domain and range blocks. The comparison for a best match between a range and domain pair is then made only if the pixel value difference of the range block is less than the domain block pixel value difference. Quadtree partition algorithm is used [18] for partitioning the image. The domain and range classification is done based on the mean and variance.

II. FRACTAL IMAGE COMPRESSION

Initially, the given image of size nxn is partitioned into overlapping domain blocks Di (of size 2rx2r), for each quadtree partition, where rxr is the size of the range blocks Ri. The domain step size used is δh=δv=4 in horizontal and vertical directions. The domains are classified based on the mean and variance of the pixels in the four quadrants of the block [18]. The domain pool D (search codebook) is constructed by placing the entire domain blocks Di, corresponding to same class in individual lists. The range-domain matching process consists of contracting each domain block to the size of the range block by averaging 2x2 pixels. During encoding, a potential range Ri, is also classified. The domain range matching process consists of searching the domain pool D for the Di and an affine transformation wi, which minimizes the rms distance between the range block Ri and the transformed domain block wi.Di, (i.e. wi .Di ≈ Ri). For a range block with n pixels, each with intensity ri and a decimated domain block with n pixels, each with intensity di, the objective is to minimize the quantity,

$$E\,(R_i, D_i) = \sum_{i=1}^{n}(s.d_i + o - r_i)^2 \qquad (1)$$

occurs when the partial derivatives with respect to s and o are zero. Solving the resulting equations will give the best coefficients s and o [5].

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

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

$$n\sum_{i=1}^{n} d_i^2 - \left(\sum_{i=1}^{n} d_i\right)^2 = 0,$$

then $s = 0$, and $o = \frac{1}{n}\sum_{i=1}^{n} r_i$

the rms error, erms $= \sqrt{E(R_i, D_i)}$ (3)

III. PROPOSED METHOD

An improved fractal image compression scheme based on the difference of pixels with maximum and minimum intensity values in the domain and range blocks is proposed. During the encoding phase, the range blocks with pixel value difference less than the pixel value difference of the domains are compared for further regression analysis (for a match). An adaptive parameter β is defined (range between 1.0 to 2.0) for scaling the pixel value difference of a domain block in different quad tree partitions. A significant reduction in encoding time is expected.

Pixel value difference

Consider a single pixel in a domain block Dj. The affine transformation in fractal encoding maps its pixel value pi to the range block Rj, using the equation,

pi (Rj) = s. pi(Dj) + o (4)

The contrast scaling parameter s must satisfy the condition 0<s<1. Let the maximum and minimum intensity level values of the pixels in a generic square block, B, are respectively, pmax (B) and pmin (B). The pixel value difference of the block B, is defined by the relation,

pdiff (B) = pmax (B) - pmin (B) (5)

Using the equations (4) and (5),

pdiff (R) = pmax (R)- pmin (R) (6)

= {s. pmax(D) + o}- {s. pmin (D)+ o}

= s.{ pmax (D) – pmin (D) }

= s. pdiff(D) (7)

Considering the contrast scaling requirement, 0<s<1, equation (7) can be written as,

pdiff (R) < pdiff (D) (8)

In the proposed implementation, for achieving better results, equation (8) is written as,

pdiff (R) < β. pdiff (D) (9)

Where, β, is an adaptive scaling parameter (varying between 1.0 and 2.0) for each quad tree partition. The condition given in equation (9) provides an effective decision rule to avoid an improper domain and range match. Only, domains satisfying the above condition will be compared for the regression analysis. Thus, many unqualified domains are avoided from comparison.

Adaptive scale parameter β for domain block pixel value difference

The parameter β is chosen adaptive for each quadtree depth, *i* to scale the pixel value difference of the domain blocks. For quadtree depth 0, (corresponding to min_part), β_0 is assigned a small initial value (in the present work, β_0=1.25). For other quadtree depths, the scale parameter is computed using the formula, $\beta_{depth}=1.25*\beta_{depth-1}$. This equation is fit, by conducting repeated experiments on images of different sizes and textures, testing for optimal value of encoding time, quality and compression ratio.

IV. Proposed Algorithm

Step 1: Construct the domain pools $\mathbf{D}_{depth}$, corresponding to each quad tree partition level starting from minimum partitions to maximum partitions (depth=0 to max_part-min_part).
Step 2: Calculate the block pixel value difference using equation (5) of all the domain blocks in each pool $\mathbf{D}_{depth}$.
Step 3: Classify and sort the domains in each pool $\mathbf{D}_{depth}$ in ascending order of the pixel value difference, and place on a list structure.
Step 4: Search for a best match between a range and domain belonging to the same class.

write_header_info; (min_part, max_part, domain_step, hsize, vszie)
depth=0; e_c =rms_tol;
Function Quadtree(image, depth) {
best_rms=infinity;
β_0=initial value; β_{depth}=1.25* β_{depth} ;
While (depth<min_part) Quadtree (image, depth+1);
Set $R_1 = I^2$ and mark it uncovered.
While there are uncovered ranges R_i do {
//Select the domain pool list $\mathbf{D}_{depth}$
Corresponding to the current range block R_i.
For (j=1; j<num_domains; ++j) {
If ($R_{pdiff} < \beta * D_{pdiff}$) {
Compute s, o, sym_op;
Compute $E(R_i, D_i)$;
If $E(R_i,D_i) \leq$ best_rms {
best_rms= $E(R_i, D_i)$;
best_domain=(domain_x,domain_y)
}
}// End for num_domains
If (best_rms>e_c) and (depth<max_part)
Quadtree (image, depth+1);
Else
Write_transformations (best_domain, s, o, sym_op);
}// End while uncovered ranges
}// End function Quadtree()

V. Experimental Results

In this section, results of the experiments conducted on various images (512x512, 8 bit gray scale) are presented. The results are compared with Fisher's classified search method [18].
The following values are used for various parameters:
5 bits were used to quantize the scaling coefficient s, and 7 bits for the offset, o.
For all images, the maximum range size is 16x16 (minimum quadtree depth 5), and the minimum range size is 4x4 (maximum quadtree depth 7). Three levels of quad tree partition are used.
The domain pool is constructed with a domain skip distance, δ_h=4 and δ_v=4, i.e. the distance between adjacent domains is 4 pixels.

The rms error tolerance, e_c is given values of 1,4,8,10,15, and 20, leading to results ranging from low to high compression. PSNR is computed after post processing.

Encoding Parameters

The following values are assigned for other parameters (Common to all images).
Image size: 512x512 (8 bit gray scale)
Number of quad tree partitions = 3
Total Number of Domains:
Three different sizes of domains are computed, corresponding to the three quad tree partitions.
Size 32x32 = ((512-32)/4+1)*((512-32)/4+1)= 14,641
Size 16x16 = ((512-16)/4+1)*((512-16)/4+1)= 15,625
Size 8x8 = ((512-8)/4+1)*((512-8)/4+1) = 16,129
Total number of domains in all partitions = 46,395

In the proposed method, the adaptive parameter β (for scaling the domain block pixel value difference) is assigned an initial value, β_0=1.25, and $\beta_{depth} = 1.25* \beta_{depth-1}$.
The algorithm is implemented in C language, using VC++6.0 compiler. Execution is carried out on a Personal Computer with Intel Centrino Duo T2250 processor with clock frequency @1.73 GHz, with 1.0 GB of RAM.

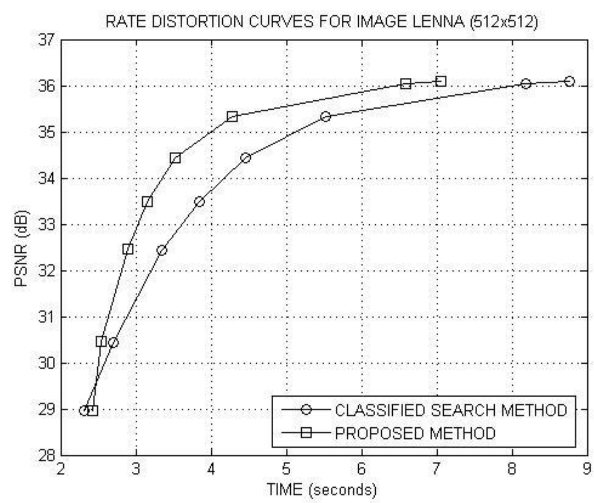

Fig.1. Encoding time vs. PSNR for Lenna Image

Table I

Results On Image Lena (512x512, 8 Bit)

Classified Search Method			Proposed Method		
CR	Time (sec)	PSNR (dB)	CR	Time (sec)	PSNR (dB)
4.36	8.76	36.09	4.36	7.06	36.09
4.85	8.18	36.04	4.85	6.59	36.04
8.67	5.53	35.34	8.66	4.28	35.34
11.96	4.46	34.43	11.96	3.53	34.43
15.46	3.84	33.49	15.45	3.15	33.49
19.13	3.35	32.44	19.08	2.89	32.46
29.23	2.70	30.43	29.13	2.54	30.46
41.91	2.31	28.95	41.71	2.42	28.96

TABLE II

Results On Image BABOON (512x512, 8 bit)

Classified Search Method			Proposed Method		
CR	Time (sec)	PSNR (dB)	CR	Time (sec)	PSNR (dB)
4.36	7.45	25.49	4.36	5.79	25.49
4.36	4.36	25.49	4.36	5.79	25.49
4.45	7.26	25.49	4.44	5.75	25.49
4.92	6.76	25.45	4.92	5.26	25.44
5.44	5.26	25.37	5.44	4.79	25.36
5.96	5.82	25.22	5.95	4.43	25.22
7.30	5.00	24.64	7.27	3.75	24.66
9.14	4.25	23.65	9.02	3.23	23.74
19.58	2.70	21.28	18.50	2.28	21.44

Table III

Results On Image Goldhill (512x512, 8 Bit)

Classified Search Method			Proposed Method		
CR	Time (sec)	PSNR (dB)	CR	Time (sec)	PSNR (dB)
4.37	9.01	33.87	4.37	7.56	33.85
4.61	8.72	33.86	4.61	7.29	33.84
5.11	8.09	33.72	5.11	6.68	33.71
6.76	6.75	32.90	6.76	5.62	32.89
9.15	5.62	31.75	9.14	4.57	31.75
12.50	4.52	30.64	12.43	3.85	30.66
25.63	3.01	28.31	25.44	2.81	28.33
43.55	2.29	26.87	42.52	2.39	26.92

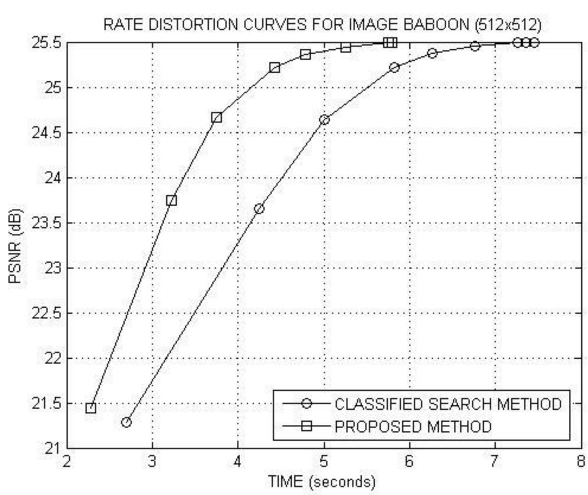

Fig.2. Encoding time vs. PSNR for Baboon Image

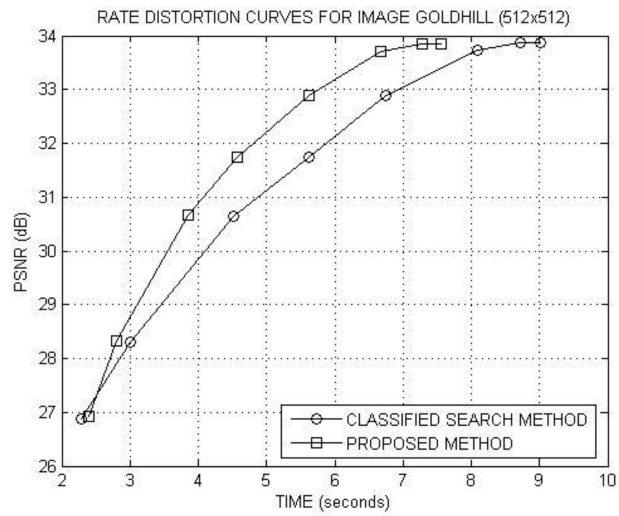

Fig.3. Encoding time vs. PSNR for Goldhill Image

Fig.4.Original Image of Lena (50%, 512x512, 8bit)

Classified Search Method			Proposed Method		
CR	Time (sec)	PSNR (dB)	CR	Time (sec)	PSNR (dB)
4.36	9.64	34.83	4.36	7.48	34.83
4.44	9.48	34.83	4.44	7.46	34.83
6.93	7.18	34.43	6.93	5.42	34.40
11.62	5.23	33.55	11.59	4.09	33.53
16.53	4.25	32.75	16.49	3.43	32.72
21.14	3.70	31.90	21.09	3.12	31.88
32.92	2.84	30.19	32.78	2.75	30.19
45.64	2.40	28.72	45.04	2.50	28.76

TABLE IV

RESULTS ON IMAGE PEPPERS (512X512, 8 BIT)

Fig.5. Decoded Lena Image by proposed method(CR=41.71, PSNR=28

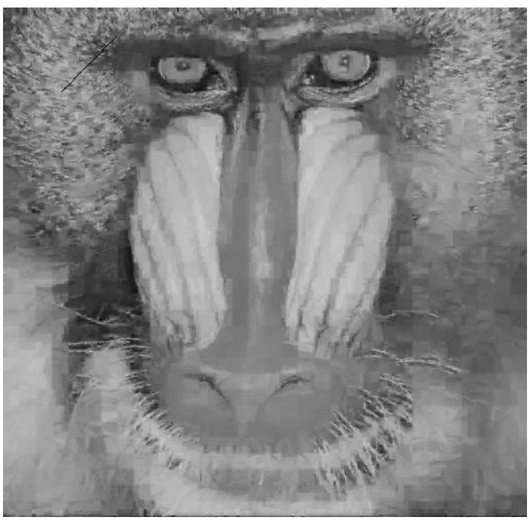

Fig.6. Decoded Baboon Image by proposed method (CR=18.50, PSNR=21.44)

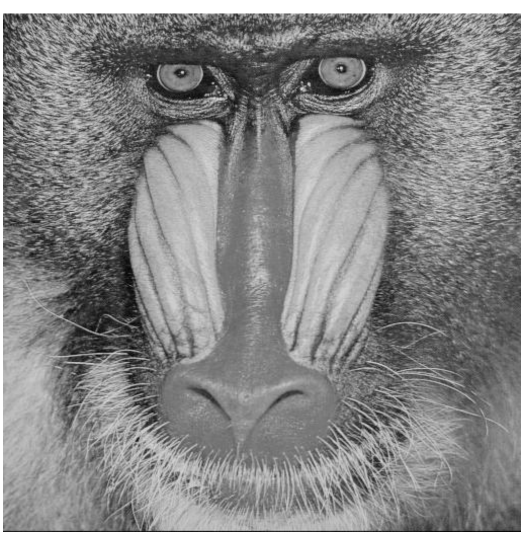

Fig.7.Original Image of Baboon (50%, 512x512, 8bit)

Fig.8.Original Image of Goldhill (50%, 512x512, 8bit)

Fig.9.Original Image of Peppers (50%, 512x512, 8bit)

Fig.10. Decoded Image Goldhill by proposed method (CR=42.52, PSNR=26.92)

Fig.11. Decoded Image Peppers by proposed method (CR=45.04, PSNR=28.76)

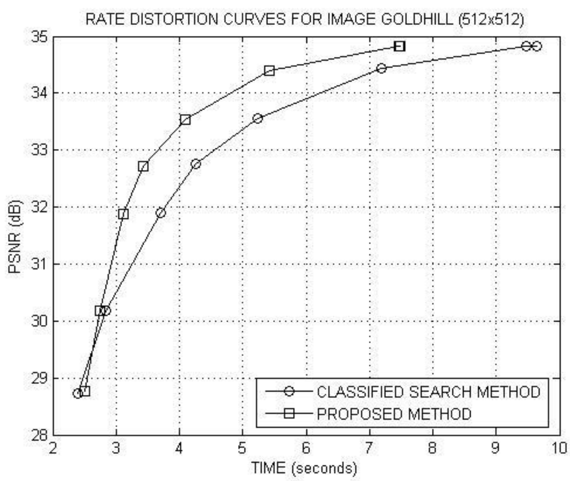

Fig. 12. Encoding time vs. PSNR for Image Peppers

VI. CONCLUSIONS

In this paper, an improved classified search algorithm for fractal compression algorithm based on adaptive pixel value difference technique is proposed. Experimental investigations revealed that the method reduces the encoding time significantly when compared to traditional classified search algorithm [18]. The reduction in PSNR is 0.05dB for peppers image. The reduction in compression ratio (CR) is by a factor of 1.73 for gold hill image.

VII. ACKNOWLEDGEMENT

The authors wish to acknowledge their gratitude to anonymous reviewers during the course of this work. They also express sincere thanks to Dr. V. Srinivasulu, Dr. N. B. Venkateswarlu, and the management of Sree Vidyanikethan for supporting this work.

VIII. REFERENCES

1) M. F. Barnsley, “Fractal Image Compression”, A. K. Peters, Ltd.Wellesly, MA,1993.

2) B. B. Eqbal, “Enhancing the Speed of Fractal Image Compression,” *Optical Engineering*, vol. 34, No.6, June 1995.

3) J. Cardinal, “ Fast Fractal compression of Gray scale Images,” *IEEE Transactions on Image Processing*, vol. 10, No.1, pp. 159-163, Jan. 2001.

4) R. Distasi, M. Nappi, and D. Riccio, “A Range/Domain Approximation Error-Based Approach for Fractal Image Compression,” *IEEE Transactions on Image Pocessing*, vol.15, No. 1, pp. 89-97, Jan. 2006.

5) R. Distasi, M. Polvere, and M.Nappi, “Split Decision functions in Fractal image Coding,” *Electronics Letters*, vol.34, No. 8, pp. 751-753, April 1998.

6) Y. Fisher, “Fractal Image Compression: Theory and applications,” Springer-Verlag, New York, 1994.

7) X. Gharavi and T.S. Huang, “Fractal Image Coding Using Rate-Distortion Optimized matching Pursuit,” *Proceedings of SPIE*, pp. 265-304, 1996

8) R. C. Gonzalez, R. E. Woods, and S. L. Eddins, “Digital image Processing,” 2 ed., Pearson Education, Inc.2004.

9) M. Hassaballah, M. M. Makky, Y. B. Mahdy, “A Fast Fractal Image Compression Method Based Entropy,” *Electronic Letters on Computer Vision and Image analysis*, 5(1), pp. 30-40, 2005.

10) R. Hamzaoui and D. Saupe, “Combining Fractal Image Compression and Vector Quantization,” *IEEE Transactions on Image Processing*, vol. 9, No. 2, pp. 197-208, 2000.

11) E.W. Jacobs, Y. Fisher, and R.D. Boss, “Image Compression: A study of the Iterated Transform Method,” *Signal Processing*, vol. 29, pp. 251-263, 1992.

12) A. E. Jacquin, “Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations,” *IEEE Transactions on Image Processing*, vol. 1, No. 1, pp 18-30, Jan. 1992.

13) C. M. Lai, K. M. Lam, and W. C. Siu, “Improved Searching scheme for fractal image coding,” *Electronics Letters*, vol. 38, No. 25, pp. 153-54, Dec. 2002.

C. K. Lee and W. K. Lee, “Fast Fractal Block coding Based on Local Variances,” *IEEE*

Transactions on Image Processing, vol. 7, No. 6, pp.888-891, June 1998.

14) M. Polvere and M. Nappi, " Speed-Up in Fractal Image Coding: Comparison of Methods," *IEEE Transactions on Image Processing*, vol. 9, No. 6, pp. 1002-1009, June 2000.
15) B. Ramanurthi and A. Gersho, "Classified Vector Quantization of Images," *IEEE Transactions on Communication*, COM-34, vol. 11, pp. 1105-1115, 1986
16) D. Saupe and S.Jacob, "Variance based quadtrees in fractal image compression," *Electronics letters*, vol. 33, No. 1, Jan. 1997
17) D. Saupe, "Lean Domain Pools for Fractal Image Compression," *Proceedings IS&T/SPIE Symposium on Electronic Imaging: Science & Technology, Still Image Compression II*, vol. 2669, June 1996.
18) D. Saupe,"Accelerating Fractal Image Compression by Multi-dimensional Nearest Neighbor Search," *Proceedings of Data compression Conference*, Mar. 1995.
19) C. S. Tong and W. Man, "Adaptive Approximation Nearest Neighbor Search for Fractal Image Compression," *IEEE Transactions on Image Processing,"* vol. 11, No. 6, pp. 605-615, 2002.
20) B. Wohlberg and Gerhard de Jager, "A review of the Fractal Image Compression Literature," *IEEE Transactions on Image Processing*, vol. 8, No. 12, pp. 1716-1729, Dec. 1999
21) C. G. ZHOU, K. MENG, and Z. QIU, "A fast Image Compression Algorithm based on Average Variance Function," *IEICE Transactions on INFORMATION & SYSTEMS*, vol. E89-D, No.3, pp.1303-1309, Mar. 2003.
22) T. Zumbakis, and J. Valantinas, "A New Approach to Improving Fractal Image Compression Times," *Proceedings of fourth Intlernational Symposium on Image and Signal Processing Analysis*, pp. 468-473, ISPA 2005