IFS Coding for Zero-Mean Image Blocks

Loay E. George
Department of Astronomy, College of Science, University of Baghdad. Baghdad-Iraq.
Received: 15/1/2005 Accepted: 31/10/2005

Abstract

In this research work a new fractal coding scheme based on IFS-transform for zero-mean range-domain blocks is investigated. Some improvements were performed on the IFS-matching stage, these improvements implies the use of moment indexing as a criteria to filter the domain blocks that suitable to match the range blocks, in addition to the use of stopping search condition based on the monitoring the minimum matching error, these additional coding steps will reduce the required long fractal coding time. The test results indicated that the proposed improvements had reduced the required coding time to more than 10 times without significant degradation in image quality (MSE or PSNR) level. Also the results indicated that the brightness preserved IFS-method gives compression results better than the traditional fractal coding method, and the offset coefficients values could be better encoded by using DPCM.

الخلاصة

في هذا البحث تم اقتراح هيكل جديد مستند على طريقة الترميز باستخدام الكسوريات وخصائص العزوم لترقيم المقاطع لمصفوفات المديات للصور. استخدمت طريقة العزوم لتحسين أداء مرحلة المطابقة، فخصائص العزوم قد تم استخدامها كمعيار لتصنيف مقاطع الصورة، وذلك من خلال حساب القيم المكممة. إن طريقة التصنيف المذكورة ستساعد على تحديد عدد المقاطع التي ستدخل في عملية المطابقة. إن استخدام شرط التوقف قد اسهم في زيادة كفاءة عملية الضغط بمقدار 15 مرة كما إن النتائج أثبتت ان الطريقة الجديدة اعطت نتائج جودة عيانية افضل بكثر من الطرق التقليدية.

Introduction

The development of a wide range of multimedia applications had led to increased research attention to data compression and particularly in image compression. Among the image compression techniques, the fractal image coding method based on the theory of iterated function systems (IFS) has captured increasing attention and interest. The application of fractal models to image compression has been prompted by Barnsley [1,2]. The first automated fractal coding algorithm based on Partitioned (local) Iterated Function System (PIFS) was developed by Jacquin [3,4] .

The basis of IFS image coding technique, known as the fractal inverse problem, is to find the IFS whose attractor close to a given image. Its basic premise is that images exhibit a type of redundancy called piecewise self-similarity [4]. In a piecewise self-similar image, a block of waveform data can be related to another one so that the two blocks resemble each other. Compression is achieved if one of them (a range block) is encoded by providing a reference to the other (a domain block). The encoding of a range consists of choosing the most similar domain and approximating the range by a linear combination of the domain and some predefined vectors. Since a range block and the corresponding domain block can be located anywhere in the image, piecewise self-similarity is different from local

redundancy exploited by more traditional image compression techniques.
The main difficulty with this method is it takes a long time to compress single image. Some methods to reduce the PIFS encoding time have been proposed. Some proposed methods involves combination of fractal coding either with cosine transform (DCT) [5] or Wavelet transform [6], other coding methods are based on utilization of pyramidal coding scheme [7,8], or using some classification criteria to classify range-domain blocks [9].

Image Fractal Coding

PIFS image encoder consists of a set of transforms on regions of the image. The set of regions (i.e., the domain blocks) from which the transform domains are chosen overlap, while the regions (i.e., the range blocks) forming the ranges of the transformation are tiled.
The set of transformations consist of a spatial contraction (e.g., averaging each 4 neighboring pixels) to construct a *kxk* blocks from a *2kx2k* blocks, followed by one of the 8 square symmetry operations (4 rotations and 4 reflections), followed by a contractive affine transformation on the grey scale values (for a block with pixel values.
For a range block with pixel values $(r_0, r_1, \ldots, r_{m-1})$, and the domain block $(d_0, d_1, \ldots, d_{m-1})$,
the contractive affine approximation is,

$$r_i' = s d_i + o \quad ,\ldots\ldots(1)$$

Where s (scale) and o (offset) are the affine transform coefficients, r_i' s are the approximate (constructed) range values. The scale (s) and offset (o) parameters are determined by applying the least sum (χ^2) of square errors between r' and r values [10]:

$$\chi^2 = \sum_{i=0}^{m-1} (r_i' - r_i)^2 \quad ,\ldots\ldots(2)$$

$$\frac{\partial \chi^2}{\partial s} = 0; \quad \frac{\partial \chi^2}{\partial o} = 0$$

$$s = \frac{m\sum_{i=0}^{m-1} d_i r_i - \sum_{i=0}^{m-1} d_i \sum_{i=0}^{m-1} r_i}{m\sum_{i=0}^{m-1} d_i^2 - \left(\sum_{i=0}^{m-1} d_i\right)^2} \quad ,\ldots\ldots(3)$$

$$o = \frac{\sum_{i=0}^{m-1} d_i^2 \sum_{i=0}^{m-1} r_i - \sum_{i=0}^{m-1} d_i \sum_{i=0}^{m-1} r_i d_i}{m\sum_{i=0}^{m-1} d_i^2 - \left(\sum_{i=0}^{m-1} d_i\right)^2} \quad ,\ldots\ldots(4)$$

In each range-domain matching instance before determining the value of χ^2, the scale *(s)* and offset *(o)* values should firstly imposed to the clipping conditions $(o_{min} \le o \le o_{max})$ and $(|s| \le s_{max})$, (o_{min}, o_{max}) are the lower and upper boundaries of the permissible values of offset, s_{max} is the maximum permissible scale value.

Secondly, they should quantized by using the following equations:

$$i_s = round\left(\frac{s}{s_{max}}(2^{a-1} - 2)\right) ,\ldots\ldots(5)$$

$$i_o = round\left(\frac{2^b - 1}{o_{max} - o_{min}}(o - o_{min})\right) ,\ldots\ldots(6)$$

$$s_q' = \frac{s_{max}}{2^{a-1} - 2} i_s \quad ,\ldots\ldots(7)$$

$$s_o' = \frac{o_{max} - o_{min}}{2^b - 1} i_o + o_{min} \quad ,\ldots\ldots(8)$$

Where, (i_s and i_o) are the quantization indices of scale and offset coefficients. (s_q and o_q) are the quantized values of scale and offset coefficients respectively.
The quantized values of scale and offset parameters should be used to construct the approximates r' and the sum of errors (χ^2)
To asses the involved computational complexity; consider an *nxn* image and *kxk* range blocks. The number of tiled range blocks is n^2/k^2, while the number of domain blocks is $(n-2k-1)^2$. The computation of best match between a range block and a domain block is $O(k^2)$. Considering k to be constant, the computational complexity of an exhaustive search is $O(n^4)$.
The most direct and easy way to reduce the search complexity is by monitoring the matching error; at any matching instance the IFS matching error is checked, if it is below a pre-defined permissible level ε (threshold) then the registered domain block is considered as the best matched block and, then, the search across the domain blocks is stopped.

IFS Coding for Zero-Mean Blocks

The traditional offset factor by using equation (3) has a dynamic range [-255,510], this may cause large errors in some image regions (or points) especially these belong to high contrast area. Also the analysis conducted in this research work indicated that the traditional offset factors require an additional bit (sign-bit), and the offset values of adjacent range blocks doesn't show a significant correlation similar to that registered between the average brightness values of the adjacent blocks. So, to handle this disadvantage a change in IFS scheme is followed, where the contractive affain transform is changed to become [10],

$$r_i' - \bar{r} = s(d_i - \bar{d}) \quad ,\ldots\ldots(9)$$

where,

$$\bar{r} = \frac{1}{m}\sum_{i=0}^{m-1} r_i \quad ,\ldots\ldots(10)$$

$$\bar{d} = \frac{1}{m}\sum_{i=0}^{m-1} d_i \quad ,\ldots\ldots(11)$$

To determine the scale (s) value, the method of least sum of square errors (depicted in equation 2) is applied to get,

$$s = \begin{cases} \dfrac{\frac{1}{m}\sum_{i=0}^{m-1} d_i r_i - Q}{\sigma_d^2} & \text{if } \sigma_d^2 > 0 \\ 0 & \text{if } \sigma_d^2 = 0 \end{cases} \quad ,\ldots\ldots(12)$$

$$\chi^2 = \sigma_r^2 + s\left[s\sigma_d^2 + 2Q - \frac{2}{m}\sum_{i=0}^{m-1} d_i r_i \right] \quad ,\ldots(13)$$

Where,

$$Q = \bar{d}\bar{r} \quad ,\ldots\ldots(14)$$

$$\sigma_d^2 = \frac{1}{m}\sum_{i=0}^{m-1} d_i^2 - \bar{d}^2 \quad ,\ldots\ldots(15)$$

$$\sigma_r^2 = \frac{1}{m}\sum_{i=0}^{m-1} r_i^2 \quad ,\ldots\ldots(16)$$

Block Indexing using Moments

For a 2-D discrete function f(x,y), the moment of order (p+q) about the center point (x_c,y_c) is defined as [11]:

$$M(p,q) = \sum_y \sum_x (x - x_c)^p (y - y_c)^q f(x,y) \quad ,\ldots\ldots(17)$$

Apply this definition to determine moments of the domain and range blocks to get,

$$\begin{aligned} M_d(1,0) &= \sum_{i=0}^{m-1} (x_i - k_c)(d_i - \bar{d}) \\ M_d(0,1) &= \sum_{i=0}^{m-1} (y_i - k_c)(d_i - \bar{d}) \\ M_r(1,0) &= \sum_{i=0}^{m-1} (x_i - k_c)(r_i - \bar{r}) \\ M_d(0,1) &= \sum_{i=0}^{m-1} (y_i - k_c)(r_i - \bar{r}) \end{aligned} \quad ,\ldots\ldots(18)$$

Where,

$$k_c = \frac{k-1}{2} \quad ,\ldots\ldots(19)$$

k is the block width (or height).

Now, let us consider the following Moments-Ratio factor (R):

$$R = \begin{cases} \dfrac{M(0,1)}{M(1,0)} & \text{if } |M(1,0)| \ge |M(0,1)| \\ \dfrac{M(1,0)}{M(0,1)} & \text{if } |M(0,1)| > |M(1,0)| \end{cases} \quad ,\ldots(20)$$

It is easily to prove that the magnitude of R factor is rotation and reflection invariant. Also combining equations (20), (18) and (9), we can easily prove that:

$$R_d = R_{r'} \quad ,\ldots\ldots(21)$$

This result implies that "*if the range and domain blocks satisfy the contractive affain transform* (equation 5)*, then their ratio factors (R_d and R_r) should have similar magnitudes. This doesn't means that any two blocks have similar R magnitudes are necessarily similar to each other*".

This fact is utilized to improve (speed up) the range-domain search task. Instead of compare all domain blocks with each affain transformed range block, we need only to test the domain blocks whose *R* magnitudes are similar to that of the tested range block. To implement this idea the following block indexing algorithm is conducted:

1. For each domain block:
   a. Determine its moment ratio R_d.

b. Determine the moment index value I_d using the following equation:

$$I_d = round(|R_d| \times N_m) \quad ,\ldots\ldots\ldots\ldots(22)$$

where, N_m is the maximum moment index value, taking into consideration that the magnitude of R_d doesn't exceed (1).

c. Store the position coordinates (x_d, y_d) of the domain block and its calculated moment index value (I_d) in a temporary array (L) of records.

2. Sort the records of the array (L) in ascending order according to their moment index value.
3. Establish a set of pointers (P) refer to the start and end of each block of records hold same I_d value.
4. For each range block:

a. Determine its moment ratio (R_r), and the corresponding moment index value (I_r) using the following:

$$I_r = round(|R_r| \times N_m) \quad ,\ldots\ldots\ldots\ldots\ldots(23)$$

b. By the help of the pointers set (P) and the temporary list of records (L); match only the domain blocks whose I_d values equal to I_r. In each matching instance determine the s and χ^2 (equations 12 and 13) for all possible symmetry cases ($sym=0,1,\ldots,7$).

c. Compare the result (χ^2) of each matching instance with the minimum χ^2 registered during the previous matching instances. If χ^2 is smaller then put its value in minimum χ^2 register (beside to the associated values of s_q, sym, x_d, y_d).

d. In the case that the new registered minimum χ^2 is less than the permissible level of error between matched blocks then stop the search process, and output the set (s_q, Sym, x_d, y_d) as best encount-ered IFS match, and go to step (4f).

e. Otherwise, start test the domain blocks whose I_d values are $(I_r \mp 1)$ to get the best IFS match, if we haven't reach to an acceptable match instance try to match the domain blocks whose I_d values are $(I_r \mp 2)$, …. and so forth, until either the registered minimum error become less than (ε_{max}) or all the domain blocks are tested.

f. Output the set of IFS code $(\bar{r}_q, s_q, Sym, x_d, y_d)$ for the tested range block.

5. After the IFS coding of all range blocks, apply the DPCM coding method to encode the sequence of $\bar{r}$ quantized values.

Test Results

The proposed methods are tested on Lena image (256x256, 8 bits); the size of range blocks is set 4x4 pixels; the search step size is 1;, the number of bits allocated for the contrast scaling factor is 3 bits; and for the mean $(\bar{r})$ of range block the number of bits is taken 6 bits. So the compression ratio (without using DPCM) would be 4.57. All methods were programmed using visual basic 6.0 and implemented on a Dell Pc with Pentium III 996 MHz processor.

For full search encoding (including the 8 symmetric cases) the required time is 236.3second, *PSNR is* 32.6dB, and the obtained compression ratio is 5.09 (with the use of DPCM to encode the $\bar{r}$ values).

The test results listed in table (1) illustrate the effect of using only the error threshold (ε) as a stopping search condition on the encoding time (T_E) and *PSNR*. The results show that a large reduction in encoding time will occur (more than 3.5 times), without cause a significant degradation in image quality.

Table (1) Encoding results with different stopping threshold (ε)

ε	Time (s)	PSNR (dB)
1	235.6	32.60
2	207.5	32.58
3	129.1	32.41
4	88.0	32.05
5	67.8	31.59
6	53.6	31.19
7	44.3	30.73

Table (2) illustrates the effect of using domain blocks filtering based on moments indexing method. Different ranges of moments indices are tested and the results indicate that the proposed moments indexing method is suitable to reduce the computational complexity (more than 20 times).

Table (3) presents the encoding results obtained by using both the threshold of matching error and moments indexing criteria, it is obvious that the coding time could be reduced more than (50 times) without making PSNR less than 30.02. The listed results show

Table (2) Encoding results using moments indexing method.

N_m	$\varepsilon_{max} = 4$		$\varepsilon_{max} = 12$		$\varepsilon_{max} = 20$	
	Time	*PSNR*	*Time*	*PSNR*	*Time*	*PSNR*
20	170.6	32.55	79.4	32.30	64.2	32.15
50	148.5	32.51	45.2	32.12	28.1	31.73
100	141.1	32.48	33.7	31.91	15.8	31.41
150	138.6	32.46	29.8	31.77	11.6	31.16
200	137.3	32.45	27.9	31.69	9.6	31.01
250	136.6	32.43	26.8	31.61	8.4	30.86
300	136.2	32.42	26.0	31.55	7.6	30.77
350	139.9	32.41	25.5	31.51	7.0	30.70
400	135.6	32.41	25.1	31.45	6.6	30.57

Table (3) Encoding results using both error threshold and moments indexing.

ε_{max}	ε	N_m=300		N_m=400		N_m=500	
		Time	PSNR	Time	PSNR	Time	PSNR
8	1	55.13	31.99	54.32	31.92	53.84	31.89
	2	55.01	31.98	54.21	31.92	53.76	31.89
	3	53.97	31.94	53.49	31.88	53.19	31.85
	4	52.94	31.79	52.70	31.75	52.58	31.72
	5	52.51	31.56	52.37	31.52	52.30	31.51
	6	52.20	31.33	52.14	31.29	52.12	31.29
12	1	26.04	31.55	25.15	31.45	24.57	31.38
	2	25.88	31.55	25.01	31.45	24.52	31.38
	3	24.88	31.51	24.28	31.40	23.93	31.35
	4	23.92	31.38	23.50	31.29	23.31	31.23
	5	23.40	31.17	23.15	31.08	23.05	31.04
	6	23.11	30.96	22.94	30.87	22.87	31.84
16	1	11.95	31.09	10.99	30.92	10.42	30.87
	2	11.78	31.08	10.88	30.92	10.33	30.86
	3	10.78	31.05	10.41	30.89	9.77	30.84
	4	9.74	30.93	9.37	30.79	9.15	30.73
	5	9.30	30.74	9.01	30.60	8.86	30.56
	6	9.02	30.53	8.80	30.40	8.69	30.38
20	1	7.56	30.77	6.57	30.57	6.00	30.48
	2	7.40	30.77	6.46	30.57	5.91	30.47
	3	6.39	30.74	5.74	30.54	5.35	30.45
	4	5.37	30.62	4.97	30.44	4.72	30.35
	5	4.92	30.45	4.61	30.27	4.43	30.20
	6	4.62	30.26	4.39	30.09	4.28	30.02

that the effect of adding error threshold criteria becomes significant only when the moments stopping criteria (i.e., $\varepsilon_{max} \geq 16$) is high.

Conclusions

The performances of the suggested moments indexing method is promising and greatly reduces the image fractal encoding time without making a significant degradation in image quality. Further work is needed to enhance the moments filter method either by adding other type of filtering criteria or by using combined scheme of moment criteria. Also, the proposed algorithm should be modified to imply quadtree partitioning scheme to increase the compression ratio.

References

1. M. F. Barnsley and L. Hurd, "*Fractal Image Compression*". A. K. Peters, Wellesley, MA, **1993**.
2. M. F. Barnsley, "*Fractals Everywhere*". Academic Press, **1993**.
3. A. E. Jacquin, "*Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformation*". IEEE Transaction on Image Processing, **1992**, vol. 1, no. 1, pp. 18-30, **1992**.
4. Y. Fisher, *Fractal Image Compression*, SIGGRAPH Course Notes, **1992**.
5. N.T. Thao, K. Asai, and Vetterli, "*Set Theoretic Compression with an Application to Image Coding*". Proceedings IEEE International Conference Image Processing, vol. II, pp. 336-340, **1994**.
6. D. J. Hebert and E. Soundararajan, "*Fast Fractal Image Compression with Triangulation Wavelets*". Proceedings SPIE Conference on Wavelet Applications in Signal and Image Processing, pp. 67-74, **1998**.
7. H. Lin, and A.N. Venetsanopoulos. "*A Pyramid Algorithm for Fast Fractal Image Compression*", Proceedings of **1995** IEEE International Conference on Image Processing, vol. 3, pp596-599, Washington, USA, **1995**.
8. D, Saupe, R. Hamzaoui, "*Complexity Reduction Methods for Fractal Image Compression*", IMA Conference Image Processing: Mathematical Methods and applications, J.M. Blackledge (Ed.), Oxford University Press, Oxford, **1995**.
9. R. Distasi, M. Nappi and S. Vitulano. "*Speeding Up Fractal Encoding of Images Using a Block Indexing Technique*", Proceedings of ICIAP97 9th International Conference on Image Analysis and Proceeding, A. Del Bimbo (Ed.), Lecture Notes in Computer Science, vol. 1311, pp.101-107, Springer-Verlag, **1997**.
10. Chong Sze Tong, Minghong Pi, "*Fast Fractal Image Encoding Based on Adaptive Search*". IEEE Transactions on Image Processing, vol. 10, no. 9, pp. 1269-1277, **2001**.
11. R.F. Gonzalez, R. E. Woods, "*Digital Image Processing*". Pearson Education Internation-al, Prentice Hall, Inc., 2nd Edition, **2002**.