ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 6, No. 4, 2011, *pp*. 287-294

Speeding Up Fractal Image Compression Using DCT Descriptors

Dr. Loay E. George[1] and Nevart A. Minas[2]

[1]University of Baghdad/ College of Science

[2]Technical Institute of Kirkuk/ Computer Systems Department

(*Received January 25, 2011, accepted May 22, 2011*)

Abstract. The main problem in the Fractal image compression (FIC) method is the long encoding time. In order to speed up the encoding stage, an improved PIFS scheme is introduced in this paper; In the introduced PIFS scheme the input color image is, converted from RGB color space to YUV color space, and then the chromatic bands are down sampled by 2. The zero-mean range and domain blocks are classified using a block descriptor which is determined using a pair of low frequency DCT coefficients due to the energy packing properties of DCT. The DCT descriptors are also used to address the isometric state of each range and domain blocks. The coefficients of the optimal affine approximation are quantized, and then coded using differential pulse code modulation (DPCM) and shift coding. The conducted tests results on Lena's standard image indicated that the encoding time is reduced to (1 sec) while keeping the image quality above the acceptable level and no significant reduction in the compression ratio was occurred.

1. Introduction

The basic idea of fractal image compression technique was introduced by Barnsley [1] in 1980 according to the contractive mapping fixed-point theorem. In his proposed iterated function system (IFS), the contacted transform consisting of a sequence of affine transformations is applied to the entire image. Later Jacquin [2] proposed a partitioned IFS (PIFS) associated with a block-based automatic encoding algorithm where those affine transformations are applied to partitioned blocks. This scheme search process is extremely time-consuming. The main reason is that the similarity matching computation between range and domain blocks is complex. In response to this problem, an effective method is that the image blocks are classified before matching, so as to speed up the encoding process.

The major methods are used the grey statistical classification characteristics, such as variance or mean of image blocks in Jacquin[2] and Fisher[3]. Some methods used the image blocks' low frequency DCT coefficients [4]. Other methods classified the image blocks from different aspects, such as with inner product[5], with moment features[6].

This paper is organized as follows. In Section 2 the basic fractal image encoding method is described. Section 3 introduces, the proposed classification algorithm, in Section 4, the tests of the proposed method are presented and compared to that of other fast classification schemes. In the last section conclusions of the comparison are drawn.

2. Traditional PIFS Encoder

The basis steps of the encoding procedure could be summarized as: "the image, to be compressed, is partitioned into blocks; each block can be approximated by other image block after some scaling operations. The result block approximation is represented by sets of transform coefficients.

The traditional PIFS encoder loads the 24-bit bitmap color image, and decomposes the image data into three 2D-arrays (i.e., Red, Green and Blue). The color bands are converted from RGB color space to YUV space. The two chromatic bands (U and V) are down sampled by 2 to increase the compression gain. These two down sampled components with the Y component are passed, as three separate gray images, to PIFS encoder.

The implemented PIFS-mapping stage implies the following processes:

(a) Range Pool Generation: Each input color band (*WxH*) is partitioned into a non-overlapping blocks of size *m*, using fixed size blocks scheme with block size (*m=LxL*). The number of range blocks N_r is determined as follows:

$$N_r = [W/L][H/L] \tag{1}$$

(b) Domain Pool Generation: The domain array is generated by down sampling the input color-band (by 2) using averaging method. The domain array dimensions are calculated as follows:

$$H_h = H \backslash 2\ ,\quad W_h = W \backslash 2 \tag{2}$$

The domain array is partitioned into overlapping fixed size blocks; using certain jump step size (Δ) for horizontal and vertical shifts. The number of domain blocks in each row and column of the domain pool (i.e., N_{xd} and N_{yd}) and total number of domain blocks (N_d) are determined using the following equations:

$$N_{xd} = \left\lfloor \frac{W_h - L}{\Delta} \right\rfloor + 1,\quad N_{yd} = \left\lfloor \frac{H_h - L}{\Delta} \right\rfloor + 1 \tag{3}$$

$$N_d = N_{xd} N_{yd} \tag{4}$$

(c) PIFS mapping: In order to avoid the computational redundancy, some of involved coding parameters have been predetermined before entering the nested loops, among these parameters are:

(1) The quantization steps of both offset and scale parameters using the following equations:

$$Q_r = \frac{255}{2^{b_r} - 1}\ ,\quad Q_s = \frac{s_{\max}}{2^{b_s - 1} - 1} \tag{5}$$

Where,

b_r is number of bits used to represent offset coefficient

b_s is number of bits used to represent sacle coefficient

S_{max} is the highest permissible value of scale coefficients(255).

(2) The average ($\overline{d}$) and variance ($m\sigma^2_d$) of each domain block (d_i):

$$\overline{d} = \frac{1}{m}\sum_{i=0}^{m-1} d_i \tag{6}$$

$$m\sigma_d^2 = \sum_{i=0}^{m-1} d_i^2 - m\overline{d}^2 \tag{7}$$

Where, *m* is the block size

(3) The reference coordinates of each domain block (x_d and y_d).

In the PIFS encoder, the search process implies that all the domain blocks (d_i) listed in the domain pool ($d_i,\ldots,d_n$) should be matched with the considered range block (r_i), to list out the optimal affine approximation (r_i'). The PIFS mappings are done between the range and domain blocks which have zero means. This kind of mapping is described by in the following equation:

$$r_i' = s(d_i - \overline{d}) + \overline{r_i} \tag{8}$$

where,

d_i is the corresponding pixel value in the domain block.

s is the scaling coefficient.