Quality Adapted Fractal Image Compression for Multimedia Communication

D. J. Duh
Dept. of Computer Science and Information Engineering,
Ching-Yun University, Taoyuan County 320, Taiwan
+886-3-250-3013 ~ 7701
duby@saturn.yzu.edu.tw

J. H. Jeng
Dept. of Information Engineering,
I-Shou University,
Kaohsiung County 840, Taiwan
+886-7-657-7711 ~ 6523
jjeng@isu.edu.tw

S. Y. Chen
Dept. of Computer Science and Engineering, Yuan-Ze University,
Taoyuan County 320, Taiwan
+886-3-463-8800 ~ 2357
cschen@saturn.yzu.edu.tw

ABSTRACT
In multimedia communication, the communication bandwidth is a key factor for real time data transfer. Image compression, such as fractal compression, is used for image communication. Encoding speed is a key factor rendering the fractal image compression unsuitable for real time multimedia communication applications. In this paper, we propose a mechanism for speed-quality control, which is capable of speeding the encoder by a specified factor with reasonable quality decay. This control strategy enables us to control the compression ratio and the reconstructed image quality in advance depending on the need of multimedia communication. The speedup method is accomplished through a classification scheme in the frequency domain using only 2 DCT coefficients, which produces very little overhead. Image qualities can be preserved as well since the scheme makes use of edge property of image blocks, in which the fractal similarity is embedded. Simulations using number of classes ranging from 1 to 30 are performed. When the number of classes is within 30, the speedup ratio can be approximately achieved with very little quality decay. As the number of classes is set to 30, the actual encoding speed can reach up to 29.1 times and there is only 0.57 dB quality decay.

Categories and Subject Descriptors
I.4.2 [IMAGE PROCESSING AND COMPUTER VISION]: Compression (Coding)-Approximate methods.

General Terms
Algorithms

Keywords: Fractal Image Compression, Speed-Quality Control, Classification, DCT, MSSIM.

1. INTRODUCTION
In multimedia communication, due to the huge amount of data and limited bandwidth, real-time image transfer is still an ongoing study. Image compression is needed for multimedia image communication. The idea of the fractal image compression (FIC) is based on the assumption that the image redundancies can be efficiently exploited by means of block self-affine transformations [1,2]. The fractal transform for image compression was introduced by M. F. Barnsley and S. Demko [3]. The first practical fractal image compression scheme, which utilized block-based transformations and an exhaustive search strategy, was introduced by A. E. Jacquin [4], E. W. Jacobs, Y. Fisher and R. D. Boss [5]. Their approach was an improved version of the system patented by Barnsley [6,7]. One of the major drawbacks of fractal coder is the computational complexity, i.e., the encoding speed. Commonly used methods to speed up the encoder are related to the partition or classification of image blocks [8-12] or limiting the search space [13,14]. Nappi et al. [11] concluded that there are two major categories for speedup techniques: classification based techniques and feature vector based techniques. In Y. Fisher’s classification method [9], a given image block was divided into four quadrants. For each quadrant, its average and variance were computed. According to certain combination of these values, 72 classes were constructed. This method reduced the searching space efficiently. However, it required large amount of computations and the arrangement of these 72 classes was complicated.

In Z. Wang, D. Zhang and Y. Yu [12], four types of range blocks were defined based on the edge of the decoded image. They used a hybrid type of coding mechanism to achieve higher compression ratios while maintaining reasonable image quality. However, they did not provide any guideline of threshold determination, the speedup ratio varied unstably from about 1.6 to 5. In C. K. Lee et al. [13], the search of domain blocks was limited according to the variance of the block. Although they provided a 1 to 10 times of speedup ratio, the ratio was image dependent. In C. M. Lai et al. [14], a single kick-out condition was proposed to speed up the encoder. They needed to combine the DCT inner product algorithm proposed in [15] to retain a speedup ratio of 4.

In this paper, we propose an edge-based classification scheme to speed up the fractal encoder with speed-quality control. A range of speedup ratios can be achieved with fair quality decay. The classifier utilizes only 2 DCT coefficients, the lowest horizontal and the vertical coefficients, to define an edge value. These two coefficients represent the strength of the horizontal and vertical energy variations in the block. With a proper criterion for threshold, image blocks are classified into classes of equal number of members according to their edge indices. In the encoding process, only the blocks of the same class are required to perform the similarity measurement. Since the searching space is reduced, the encoding speed can be improved. Also, since only 2 DCT coefficients are involved, a very small amount of overhead is imposed, in comparison with...
the complexity of the encoder. In the meanwhile, the structure of the classifier is simple in the sense of fractal coder because it merges the Dihedral transformations together. The thresholds for the classifier are adaptively determined so that the speedup ratio is proportional to the number of classes and independent to the given image. Although it is obvious that the quality decay of the retrieved images grows as the number of class increases, the quality can still be retained in a reasonable range. This is because the classifier is designed according to the edge property of image blocks. As a consequence, a mechanism of speed-quality control can be achieved.

2. FULL SEARCH FRACTAL IMAGE COMPRESSION

The fundamental idea of fractal image compression is based on the Iteration Function System (IFS) with two governing theorems which are the Contractive Mapping Fixed-Point Theorem and the Collage Theorem [9].

Fractal image compression is an inverse problem, i.e., for the given set \( S \), find the IFS which has \( S \) as its attractor. It should be noted that, when \( S \) is a natural image, such IFS can hardly exist because most of the sub-images are not directly similar to the whole image. To solve this problem, the idea of local self-similarity is adopted to form the Partitioned Iterated Function System (PIFS) in which the contractive maps \( w_j \) is defined only on \( D_i \) where \( D_i \subset X \) for \( i = 1, \ldots, n \).

Let \( f \) be a gray level image of size \( N \times N \). Let the range pool \( R \) be the set of the \( (N/8)^2 \) non-overlapping blocks of size \( L \times L \) which is the size of encoding unit. Let \( \mathbf{E} \) be the domain pool makes up the set of \( 2^{31}(\mathbb{N} \cup \{0\}) \) for the given range block \( v \) in \( R \), one searches in the domain pool \( D \) to find the best match, i.e., the most similar domain block. The parameters describing this fractal affine transformation form the fractal compression codes of \( v \). At each search entry, the domain block is first down-sampled to \( 8 \times 8 \) and denoted as \( u \). The down-sampled block is transformed subject to the eight transformations in the Dihedral group on the pixel positions. If the origin of \( u \) is assumed to locate at the center of the block, the eight transformations of \( T_0 \), \( T_1 \), \( T_2 \) and \( T_3 \), respectively, performed by an additional flip along the main diagonal line.

In the idea of fractal coding, a contrast scaling and a brightness offset are allowed to operate on the transformed domain blocks. Thus the fractal affine transformation \( \Phi \) of \( u(x,y) \) in \( D \) can be expressed as:

\[
\Phi \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{2}
\]

where the 2x2 sub-matrix \( \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \) is one of the dihedral transformations in equation (1). In each search entry, eight separated computations of Mean Square Error (MSE) are required to find the index \( d \) such that:

\[
d = \arg \min \{ \text{MSE}((p_k, u_k) + q_k) : k = 0, \ldots, 7 \} \tag{3}
\]

where

\[
\text{MSE}(u,v) = \frac{1}{L^2} \sum_{i,j} (u(i,j) - v(i,j))^2
\]

Here \( p_k \) and \( q_k \) can be computed directly by calculus optimization method as:

\[
p_k = \frac{1}{L^2} \sum_{i,j} (u(i,j) - v(i,j))^2
\]

\[
q_k = \frac{1}{L^2} \sum_{i,j} v(i,j) - 1 \sum_{i,j} u_k(i,j)
\]

As \( u \) runs over all of the 58081 domain blocks in \( D \) to find the best match, the terms \( t_x \) and \( t_y \) in equation (2) can be obtained. Together with \( d \) and the specific \( p \) and \( q \) corresponding to this \( d \), the affine transformation in equation (2) is found for the given range block \( v \). In practice, \( t_x \), \( t_y \), \( p \), \( q \) and \( d \) can be encoded using 8, 8, 5 and 3 bits, respectively, which are regarded as the compression code of \( v \). Finally, as \( v \) runs over all of the 1024 range blocks in \( R \), the encoding process is completed. To decode the image, we first make up the 1024 affine transformations from the compression codes and choose an arbitrary image as the initial image. Then, we perform the 1024 affine transforms on the image to obtain a new image, and proceed recursively. According to Partitioned Iteration Function Theorem (PIFS), the sequence of images will converge. The stop criterion of the recursion is designed according to user’s application. The final image is the decoded image of fractal coding.

3. CLASSIFICATION SCHEME BASED ON EDGE PROPERTY

The proposed local search strategy is designed through a classification scheme based on the edge property of blocks. Such a strategy can speed up the encoder and preserve image quality. The classifier operates in the frequency domain and makes use of edge property incorporated with the Dihedral transformations.

Let \( u \) be a given image block of size \( L \times L \). The DCT of \( u \), denoted by \( U \), is computed from the formula:

\[
U(i,j) = \sum_{k=-L/2}^{L/2} \sum_{l=-L/2}^{L/2} u(k,l) \cos \left( \frac{\pi}{L} (k+i) \right) \cos \left( \frac{\pi}{L} (l+j) \right)
\]
\[
U(m,n) = \frac{2}{L} C_k \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} u(i,j) \cos \left( \frac{(2i+1)m\pi}{2L} \right) \cos \left( \frac{(2j+1)n\pi}{2L} \right)
\]

where \( m,n = 0,1,\ldots,L-1 \), and

\[
C_k = \begin{cases} 
1/\sqrt{2} & \text{if } k = 0 \\
1 & \text{else}
\end{cases}
\]

Taking \( L = 8 \) as an example, we have 64 DCT coefficients. Among the 64 DCT coefficients, lower coefficients possess most of the energy of the block. Higher coefficients contain only finer shape information. If the lower coefficients of two blocks are alike, the similarity measure between them won’t be large due to the energy concentration. On the other hand, if two blocks are similar except for some content details, the magnitudes of their lower DCT coefficients will be roughly the same. Among the lower coefficients, the quantity \(|V/u|\) which contains the main brightness information of the block can be integrated into \( q_k \) in equation (5) during the calculation of fractal code. Therefore, the lower coefficients excluding the DC term are sufficient for the similarity measure of two blocks. Let \( V = U(1,0) \) and \( H = U(0,1) \). For the case of \( L = 8 \), we have:

\[
V = \sqrt{2} \sum_{j=0}^{7} \sum_{i=0}^{7} u(i,j) \cos \theta_i
\]

\[
H = \sqrt{2} \sum_{j=0}^{7} \sum_{i=0}^{7} u(i,j) \cos \theta_j
\]

where \( \theta_i = (2i+1)\pi/16 \) and \( \theta_j = (2j+1)\pi/16 \) with \( i,j = 0,1,\ldots,7 \). It is clear that \( \cos \theta_i > 0 \) for \( i = 0,1,2,3 \) and \( \cos \theta_i < 0 \) for \( i = 4,5,6,7 \). Also, since \( \cos \theta_i \) is independent of \( j \), the quantity \( V \) reflects the energy variation between the left half and the right half of the block \( u \). Similarly, the quantity \( H \) measures the energy variation between the upper half and the lower half of \( u \).

The function \( \cos \theta_i \) is an odd function with respect to the middle vertical line of \( u(i,j) \). Thus the value of \( V \) is negated if \( u \) is transformed by \( T_1 \) given in equation (1). Since \( H \) is independent of \( \cos \theta_i \), it will remain unchanged. Similarly, if the horizontal flip \( T_2 \) is applied, \( V \) is unchanged and \( H \) is negated. The transformation \( T_3 \) flips \( u \) both vertically and horizontally, therefore both \( V \) and \( H \) are negated. The additional transformation \( T_4 \) for which \( T_{k+4} = T_k T_4 \), \( k = 0,1,2,3 \), flips a block about the main diagonal line, so it will exchange the indices \( i \) and \( j \) for \( u(i,j) \). According to equation (6), the quantities \( H \) and \( V \) are interchanged. The detailed changes of \( V \) and \( H \) corresponding to the eight Dihedral transformations are summarized in Table 1.

<table>
<thead>
<tr>
<th>( T_0 )</th>
<th>( T_1 )</th>
<th>( T_2 )</th>
<th>( T_3 )</th>
<th>( T_4 )</th>
<th>( T_5 )</th>
<th>( T_6 )</th>
<th>( T_7 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( V )</td>
<td>( -V )</td>
<td>( +V )</td>
<td>( +V )</td>
<td>( -V )</td>
<td>( +H )</td>
<td>( -H )</td>
<td>( +H )</td>
</tr>
<tr>
<td>( H )</td>
<td>( +H )</td>
<td>( -H )</td>
<td>( -H )</td>
<td>( +V )</td>
<td>( +V )</td>
<td>( +V )</td>
<td>( -V )</td>
</tr>
</tbody>
</table>

The design of the classifier is based on the relations between the magnitudes of \( H \) and \( V \) of given blocks, in which the eight Dihedral transformations are incorporated. First, we calculate \( H \) and \( V \) of a block. Then we perform the appropriate transformations according to Table 1 such that the new values of \( H \) and \( V \) of the transformed block are both positive and satisfy \( |V|/|H| \). That is, there are projected to the \( 45^\circ \) fan area and the \( H \)-axis in the first quadrant of \( H \)-\( V \) coordinate system. With this incorporation of the Dihedral transformations, we design the classifier according to the quantity \(|V|/|H|\) which will serve as an indication of edge property of an image block.

Now we will partition the \( 45^\circ \) fan area into \( C \) sub-fan areas according to the argument above. Let \( E_u \) be the edge value defined as:

\[
E_u = \min \left[ \frac{|V|}{|H|}, \frac{|H|}{|V|} \right]
\]

Let \( \tau = \{\tau_0,\tau_1,\ldots,\tau_C\} \) be the set of \( C+1 \) predefined thresholds. The classification scheme will assign a block \( u \) to one of the \( C \) classes according to the magnitude \( E_u \) and the thresholds \( \tau \) by the criterion:

\[
c_i = j \quad \text{if} \quad \tau_j \leq E_u < \tau_{j+1}, \quad \text{for some} \quad j \in \{0,1,\ldots,C\}
\]

If \( E_u \) is small, the block \( u \) tends to have strong vertical or horizontal edges. On the other hand, if \( E_u \) is close to 1, \( u \) tends to have strong diagonal or sub-diagonal edges. The block diagram of the classifier is depicted in Fig. 1, and the steps are summarized as follows:

1. Compute \( V \) and \( H \) of \( u \) from equation (7) and equation (8).
2. Compute the edge value \( E_u \) from equation (9).
3. Search for \( \tau_j \) such that \( \tau_j \leq E_u < \tau_{j+1} \) and assign the index of the block as \( j \).

It is obvious that the classification scheme is simple and only few computations are required to calculate the lower DCT coefficients. The overhead of this classifier is very small compared to the fractal coding process. Moreover, since the classifier is designed according to the edge property, the image quality can be preserved.

4. SPEED-QUALITY CONTROL

The classification scheme based on edge property given in Section 3 is now applied to fractal image encoder to accomplish speed-quality control. Given a desired speedup ratio, the encoder will attain the ratio with reasonable quality decay. For a given ratio \( C \), we set the number of classes as \( C \) and calculates the thresholds \( \tau = \{\tau_0,\tau_1,\ldots,\tau_C\} \) from the domain pool \( D \) with \( \tau_0 = 0 \) and \( \tau_C = 1 \). The thresholds are adaptively determined so as to partition \( D \) into \( C \) sub-pools of nearly equal number of members, which are denoted as \( D_j \), \( 0 \leq j < C \).

Each domain block is assigned a class index respectively. To encode a range block, we first determine its class index according to the same thresholds \( \tau \) and find the best match from those domain blocks with the same class index. The block

![Figure 1. Block diagram of the classification scheme based on edge property.](image-url)
diagram of fractal image encoder with speed-quality control is given in Fig. 2, where the Threshold and Domain Sub-pool Determination block is the pre-processing stage.

The number of MSE computations for finding the best match, which is proportional to the speedup ratio, can be estimated. Let $M_D$ and $N_R$ denote the numbers of blocks in the domain pool $D$ and range pool $R$, respectively. Also, let $m_j$ and $n_j$ denote the numbers of blocks after classification in each domain sub-pool and range sub-pool, respectively. The total number of MSE computations for the encoder can be estimated by the following equation:

$$\sum_{j=0}^{C-1} m_j \times n_j = \sum_{j=0}^{C-1} \frac{1}{C} M_D \times n_j = \frac{1}{C} M_D \times \sum_{j=0}^{C-1} n_j = \frac{1}{C} M_D \times N_R$$

Since the encoding time essentially comes from the number of MSE computations and $M_D \times N_R$ is the total number of MSE computations for the full search method, we will achieve the speedup ratio of $C$ theoretically. For small $C$, since the overhead of the amount of MSE computations dominates and the domain pool is almost equally partitioned, the desired speedup ratio can be achieved. On the other hand, when $C$ is large, we might not be able to equally partition the domain pool. Therefore, along with the overhead of the classifier and the pre-processing unit, the speedup ratio can no longer be guaranteed. It will end up to a saturation value.

5. EXPERIMENTAL RESULTS

The proposed speed-quality control mechanism for fractal coding is simulated and verified. The programs are implemented using Borland C++ Builder 5.0 running on Microsoft Windows 2000, Pentium III 1 GHz platform. In the simulation, 4 images, Lena, Pepper, F16 and Baboon, are tested and the statistics are analyzed. The speedup ratio is defined as the ratio of the encoding time of the full search method to that of the proposed method. The distortion or error between the original image $f(i,j)$ and the decoded image $\hat{f}(i,j)$ caused by lossy compression process is measured in Peak Signal to Noise Ratio (PSNR) defined by:

$$\text{PSNR} = 10 \log_{10} \left( \frac{255^2}{\text{MSE}} \right)$$

The quantity $C$, i.e., the number of classes, is the key term in the simulation. The full search method is equivalent to the proposed method with $C=1$. Fig. 3 shows the speedup ratio for $C$ ranging from 1 to 30. As illustrated, the speedup ratios nearly coincide with the numbers of classes for all of the four images. The quality of the decoded images versus the number of classes is shown in Fig. 4. It indicates that as $C$ increases, only small amount of quality decay is imposed. The curve of speed-quality control is depicted in Fig. 5. It reveals that high speedup ratios can be achieved by specifying the quantity $C$ with only little quality decay. The detailed simulation results of Lena are given in Table 2 with $C=1$ corresponding to the full search method. As shown in the table, when $C$ is chosen to be 30, the encoding speed is about 29 times faster with only 0.65 dB of quality decay in comparison with that of the full search method.
Table 2. Encode time, Speedup ratio and PSNR of Lena for small C.

<table>
<thead>
<tr>
<th>Image</th>
<th>Lena</th>
<th>Number of Classes</th>
<th>Encode Time (sec)</th>
<th>Speedup Ratio</th>
<th>PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1650.530</td>
<td>1.000</td>
<td>28.880</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>826.107</td>
<td>1.998</td>
<td>28.820</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>551.413</td>
<td>2.993</td>
<td>28.751</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>413.885</td>
<td>3.988</td>
<td>28.682</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>331.407</td>
<td>4.980</td>
<td>28.650</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>276.468</td>
<td>5.970</td>
<td>28.636</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>237.000</td>
<td>6.964</td>
<td>28.515</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>207.909</td>
<td>7.939</td>
<td>28.530</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>184.776</td>
<td>8.933</td>
<td>28.543</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>166.349</td>
<td>9.922</td>
<td>28.526</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>151.408</td>
<td>10.901</td>
<td>28.523</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>138.880</td>
<td>11.885</td>
<td>28.437</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>128.344</td>
<td>12.860</td>
<td>28.474</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>119.131</td>
<td>13.855</td>
<td>28.453</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>111.341</td>
<td>14.824</td>
<td>28.464</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>104.580</td>
<td>15.783</td>
<td>28.380</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>98.311</td>
<td>16.789</td>
<td>28.388</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>92.934</td>
<td>17.760</td>
<td>28.369</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>88.337</td>
<td>18.685</td>
<td>28.397</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>83.580</td>
<td>19.748</td>
<td>28.319</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>79.745</td>
<td>20.698</td>
<td>28.376</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>76.239</td>
<td>21.649</td>
<td>28.320</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>73.166</td>
<td>22.559</td>
<td>28.238</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>70.080</td>
<td>23.552</td>
<td>28.263</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>67.277</td>
<td>24.533</td>
<td>28.206</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>64.713</td>
<td>25.505</td>
<td>28.270</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>60.177</td>
<td>27.428</td>
<td>28.172</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>58.033</td>
<td>28.441</td>
<td>28.200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>56.722</td>
<td>29.099</td>
<td>28.233</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6. CONCLUSION

In this paper, the mechanism of speed-quality control of fractal image compression for multimedia communication is proposed. The control is designed through a classification scheme based on edge property of image blocks. The achievement of the control is owing to the following four main issues: 1) The classification scheme causes low overhead since only 2 DCT coefficients are considered. 2) The classification scheme is efficient since the fractal Dihedral transformations are incorporated. 3) Speedup ratios are guaranteed up to some extent because of adaptive threshold. 4) Image quality can be retained because edge property is adopted. Simulation results show that the speedup ratios can be attained with fair quality decays by specifying the corresponding number of classes less than 30. The controllability fades for higher speedup ratios and is dependent on the content of the image itself. This control strategy enables us to control the compression ratio and the reconstructed image quality in advance depending on the need of multimedia communication.

7. REFERENCES


