FINITE-STATE FRACTAL BLOCK CODING OF IMAGES

Hsuan T. Chang and Chung J. Kuo

Signal and Media (SAM) Laboratory
Department of Electrical Engineering
National Chung Cheng University
Chiayi, Taiwan, 62107 ROC

ABSTRACT

An image compression scheme based on the finite-state fractal block coding (FS-FBC) is proposed in this paper. Since the FS algorithm has been successfully employed by the vector quantization (VQ) techniques and there are many similarities between VQ and FBC scheme, we employ the FS algorithm on FBC technique to determine the fractal codes. The next-state functions for the domain pool size, contrast scaling factor, isometry, and mean are designed. In computer simulation, we first independently investigate the FS effect of each parameter and then combine all methods to test the overall effect. We obtain a reconstructed Lena image with 33.9 dB PSNR at 0.42 bit/pixel, while the result of original FBC scheme is 31.4 dB at 0.6 bit/pixel. The simulation results show that our FS methods can be successfully employed on the FBC technique to reduce the bit rate and the image quality is still preserved.

1. BACKGROUND

Image coding based on fractal theories and techniques are recently developed and received a great deal of attention [1]. Jacquin first proposed the *fractal block coding* (FBC) scheme to automatically convert an image into a partitioned iteration function system which is a set of contractive affine transformations (CATs) [2,3]. The small range block R and large domain block D used in CAT both are partitioned from an image. The two-level partitioned range blocks are of sizes 8×8 (parent) and 4×4 (child) and the corresponding domain-block sizes are 16×16 and 8×8, respectively. With pairing the range and transformed domain blocks, the coefficients (luminance shift $\triangle g$, contrast scaling factor α, isometry I_n, and domain block's position P_D) represent the best-match transformation which minimizes

$$\epsilon = \mathrm{MSE}(R, I_n\{\alpha[S \circ D] + \triangle g\}) \qquad (1)$$

(where $\mathrm{MSE}(\cdot,\cdot)$ represents the mean-squared-error measurement and $S\circ$ performs the spatial contraction) are recorded and thus called the *fractal code*. In decoding stage, we apply fractal codes to an arbitrary initial image. The reconstructed image is iteratively decoded using the CATs denoted by fractal codes and the iteration stops when the image nearly converges.

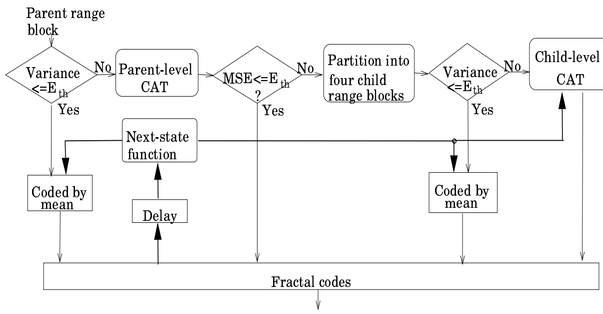
It is conceived that there are many similarities between vector quantization (VQ) and FBC scheme for digital image coding [2,3]. For example, both techniques partition an image into blockwise subimages and then encode the image using the image pieces, or vectors. Also the block searching, block matching, and MSE measurements are employed. VQ technique uses the code vectors in an off-line designed codebook to encode images by the corresponding indexes. On the other hand, FBC scheme uses the current image to generate the codebook (domain pool) and represents a compressed image by fractal codes which record CATs between domain and range blocks.

We first make an improvement based on FBC scheme via some modifications [7]. First of all, the domain pool is confined in the neighborhood of the range block and its size is fixed [4]. The luminance shift in fractal code is replaced by the range block's mean (i.e., average pixel value). Moreover, the range of the contrast scaling factor is adaptively adjusted and we use a full search to obtain the best one [5]. Finally, in decoding stage the iteration is fast and the blocking effects appearing at the mean-coded blocks are reduced by a post-processing technique.

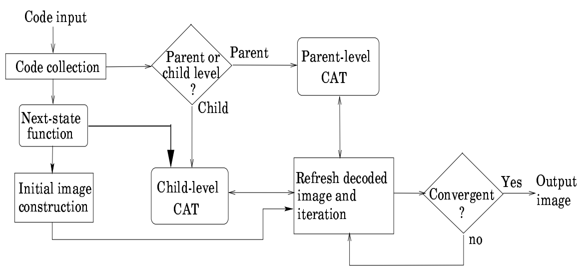
The finite-state (FS) algorithm has been successfully employed on VQ technique to improve the coding performance [6]. In FSVQ technique, the state codebook is determined by a specific next-state function which uses the information of past coded blocks. Based on the same idea, we employ the FS algorithm on determinating all parameters in fractal code.

2. FS-FBC SCHEME

The system diagram of the proposed FS-FBC scheme is shown in Figure 1. For a parent range block, the coding procedures are based on our modified FBC scheme [7]. Once a parent range block is splitted into four child blocks with an MSE criterion, we then use the proposed FS algorithm to determine their fractal codes. Since neighboring child blocks have stronger correlations than that in parent blocks, we only employ the FS algorithm on determining fractal codes (except for mean, its FS algorithm is applied on both



(a)



(b)

Figure 1: The system diagram of the proposed FS-FBC scheme (a) Encoder and (b) Decoder. (Bold lines represent the FS paths.)

parent and child levels of child range blocks. The next-state function design for each parameter in fractal code is described as follows.

2.1 Domain Pool Design

For a child range block R_{c_i}, the domain block's position can be represented by the x- and y-direction displacements (m_{x_i}, m_{y_i}) from the bottom-left corner of the domain pool. Suppose that the domain block's number in the domain pool is $L_{x_i} \times L_{y_i}$, the next-state function η used to determine the domain block's number in the next domain pool can be expressed as

$$\left\{ \begin{array}{ll} L_{x_{i+1}} & = \eta(m_{x_i}, L_{x_i}, m_{x_j}, L_{x_j}) \\ L_{y_{i+1}} & = \eta(m_{y_i}, L_{y_i}, m_{y_j}, L_{y_j}) \end{array} \right. \quad (2)$$

where the index i and j represent the left and upper positions of the current range block $R_{c_{i+1}}$, respectively. We assign nine domain pool's sizes with domain blocks' number in it are 16×16, 16×8, 8×16, 16×4, 4×16, 8×8, 8×4, 4×8, and 4×4. The proposed next-state function $\eta(m_{x_i}, L_{x_i}, m_{x_j}, L_{x_j})$ acts as follows. First we calculate the summed absolute distance,

$$\left\{ \begin{array}{ll} d_{x_i} & = |m_{x_i} - L_{x_i}/2| + |m_{x_j} - L_{x_j}/2| \\ d_{y_i} & = |m_{y_i} - L_{y_i}/2| + |m_{y_j} - L_{y_j}/2| \end{array} \right. \quad (3)$$

Both the $L_{x_{i+1}}$ and $L_{y_{i+1}}$ sizes of the domain pool are then determined by the same next-state function.

{ *Next-state function for Domain Pool Design*}

if (row=0 or column=0) $L_{x_i} = L_{x_{i+1}} = 8$;
else
  if ($L_{x_i} = 16$) { if ($d_{x_i} < 7$) $L_{x_{i+1}} = 8$;
  else $L_{x_{i+1}} = 16$; }
  if ($L_{x_i} = 8$) { if ($d_{x_i} < 2$) $L_{x_{i+1}} = 4$;
  elseif ($d_{x_i} > 5$) $L_{x_{i+1}} = 16$;
  else $L_{x_{i+1}} = 8$; }
  if ($L_{x_i} = 4$) { if ($d_{x_i} > 1$) $L_{x_{i+1}} = 8$;
  else $L_{x_{i+1}} = 4$; }

The idea is originated from that if the domain blocks' positions of left and upper range blocks are far from the center of the domain pool, then the domain pool for the current range block should be extended. Otherwise, the domain pool should be contracted. The domain pool's size is not fixed now. It becomes adaptive with previous domain block's positions. If the neighboring blocks of the current child range block are not also child range blocks coded by fractal transform (that is, they could be parent range blocks coded by mean or fractal transform, or child range blocks coded by mean), we use following methods. For a parent range block coded by CAT, we directly apply the parameters $\{m_{x_i}, L_{x_i}, m_{x_j}, L_{x_j}\}$ in its fractal code. Otherwise, those parameters are preset to default values.

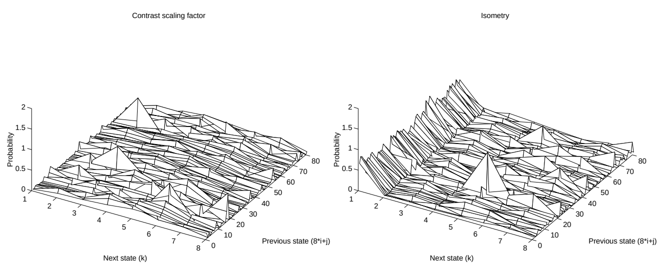
In computer simulation, the Lenna image is used to test the proposed scheme. We obtain a better peak-to-signal-ratio (PSNR) performance (34.06 dB) than that (33.96dB) of our modified FBC scheme that uses a fixed domain pool size (under the same bit rate 0.47 bit/pixel). It verifies that our design of the next-state function is more efficient than a fixed size for the domain pool. Also our previous work in [6] can be seen as a special case of the proposed FS-FBC scheme.

2.2 Contrast Scaling Factor and Isometry

Designing the next-state functions for contrast scaling factor and isometry are also only available for child range blocks here. For our modified FBC scheme, the statistics of the state transition probabilities of both parameters on the child-level are first obtained and shown in Figure 2. There are nine possible states that the left or upper range block could be. For range blocks coded by mean and CAT, we assign state '0' and states '1∼8' (represented by 3 bits in both contrast scaling factor and isometry), respectively. We thus have 9^2=81 possible combinational states $\{(S_i, S_j)|i$=0∼8, j=0∼8} for the two previous (left and upper) blocks. Two LUTs are then built by using their state transition probabilities $P(S_k|(S_i, S_j))$, where S_k denotes the next state.

Consider the building of the LUT for the contrast scaling factor, Figure 2(a) shows that only few states have a higher transition probability. For each previous state (S_i, S_j), if the sum of the state-transition probabilities of any four neighboring next states are greater than or equal to 0.8, then the possible contrast scaling factor for current range block is narrowed down to these four values and the representative bit number is reduced to two (original bit number is three). Otherwise, the range of the contrast scaling factor and the representative bit number are not changed.

Next, consider the LUT design for the isometry, Figure 2(b) shows that the identity '1' in all next states has



(a) (b)

Figure 2: Statistics of (a) contrast scaling factor, (b) isometry.

much higher probability than others. For each previous state (S_i, S_j), if the sum of the state transition probabilities of any four next states are greater than 0.8, then only these four states are tested in CAT and the representative bit number is reduced to two. Otherwise, eight isometries are tested and the representative bit number is still three. We thus use the LUTs to determine the contrast scaling factor and isometry for the current child range block in this adaptive FS-FBC scheme.

Our simulation results show that we obtain a 33.91 dB PSNR at 0.46 bit/pixel when the LUT for the contrast scaling factor is employed. On the other hand, we obtain a 33.86 dB PSNR at 0.46 bit/pixel when the LUT for isometry is employed.

2.3 Range block's mean

In our modified FBC scheme, the image is partitioned into two-level (parent and child) range blocks. Each range block (either parent and child) has its mean as a part of the fractal code. Usually the means of the neighboring range blocks are similar or change gradually. We can estimate a flexible range of the mean of the current range block by using the means of previous range blocks. The representative bit number for the range block's mean is no longer fixed, since the estimated range are adapted by the means of previous blocks.

Since the range block has two different sizes, we therefore use two different but similar next-state functions to estimate the range of current block's mean. Considering a parent range block, there are five child range blocks $R_{c_A} \sim R_{c_E}$ at its left, upper, and upper-left sides (see Figure 3(a)). These blocks' means $\mu_{c_A} \sim \mu_{c_E}$ are used for our next-function design. We first calculate the difference between the maximal and minimal values of these five blocks' means. This difference is first calculates by

$$\delta_p = \max(\mu_{R_{c_A}}, ..., \mu_{R_{c_E}}) - \min(\mu_{R_{c_A}}, ..., \mu_{R_{c_E}}) \qquad (4)$$

δ_p is used to estimate the range of the current range block's mean, $\mathcal{R}_p$, and the representative bit number of current range block's mean as follows. If the current range block is

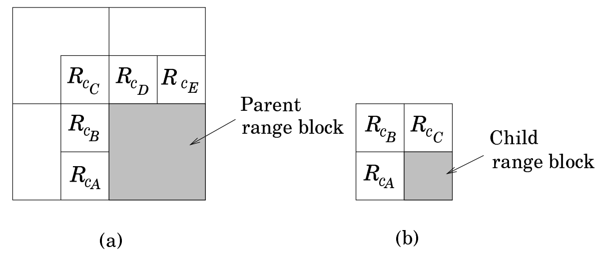


Figure 3: Blocks for estimating the range of the current range block's mean: (a) parent level, (b) child level.

coded by mean, then

$$\mathcal{R}_p = \begin{cases} \mu_{R_p} \text{ (6 bits)}, & \text{if } \delta_p \geq 64 \\ 32 \text{ (5 bits)}, & \text{if } 64 > \delta_p \geq 32 \\ 16 \text{ (4 bits)}, & \text{if } 32 > \delta_p \geq 16 \\ 8 \text{ (3 bits)}, & \text{elsewhere} \end{cases} \qquad (5)$$

Otherwise (i.e., the current range block is coded by CAT),

$$\mathcal{R}_p = \begin{cases} \mu_{R_p} \text{ (6 bits)}, & \text{if } \delta_p \geq 16 \\ 32 \text{ (5 bits)}, & \text{if } 16 > \delta_p \geq 8 \\ 16 \text{ (4 bits)}, & \text{if } 8 > \delta_p \geq 4 \\ 8 \text{ (3 bits)}, & \text{elsewhere} \end{cases} \qquad (6)$$

There are four ranges in the next-function design and the center value of the range is the average value of these five means. Basically, a larger difference provides a larger range (more representative bits) of mean and vice versa. For a child range block, there are only three child range blocks in its left, upper, and upper-left sides (see Figure 6(b)). Therefore,

$$\delta_c = \max(\mu_{R_{c_A}}, \mu_{R_{c_B}}, \mu_{R_{c_C}}) - \min(\mu_{R_{c_A}}, \mu_{R_{c_B}}, \mu_{R_{c_C}}) \qquad (7)$$

Its next-state function is similar to the case above. The mapping from δ_c to range $\mathcal{R}_c$ for the current range block is the same as Equation (6) except that δ_p and μ_{R_p} are replaced by δ_c and μ_{R_c}, respectively. Finally, the current range block's mean is determined and restricted in the range estimated by the next-state function.

The representative bits for the range block's mean is changing according to the previous range blocks' means now. It is more efficient since we explore the spatial continuity of the range block's mean. Therefore, we expect more saving on bit rate in this FS design of mean. We obtain a 33.83 dB PSNR with 0.43 bit/pixel. Here the bit rate is significantly reduced comparing with two FS designs for the isometry and contrast scaling factor.

2.4 Combined Result

In previous subsections, we independently discuss the next-state function design of each parameter in fractal code. Now we combine all the next-state functions above to obtain the total effect. We finally obtain a decoded Lenna image with 33.86 dB PSNR at 0.42 bit/pixel (shown in Figure 4(a)). With our post-processing technique, the blocking effects are reduced and the PSNR is further improved to 33.90 dB.

Table 1 summarizes the performances of our modified FBC scheme and the proposed adaptive FS-FBC scheme for Lena image.

.

Scheme	[2]	[5]	Modified FBC	Domain pool	Contrast scaling	isometry	Mean	Combined
Bit rate (bpp)	0.6	0.48	0.47	0.47	0.46	0.46	0.43	0.42
PSNR (dB)	31.4	33.8	33.96	34.06	33.91	33.86	33.83	33.86

Table 1. Coding performances of Lenna image based on different schemes.

We observe that our FS design for domain pool gets a better PSNR than that of a domain pool with a fixed size. Therefore, our adaptive FS design of the domain-pool size based on the next-state function is more efficient.

The combined result achieves a decoded Lena image with 33.86 dB at 0.42 bpp. We greatly reduce the bit rate at a negligible degradation of image quality. Comparing with the result of out modified FBS scheme (shown in Figure 4(b)), the image quality is almost the same. The PSNR result is only 0.1 dB less than our original design and the bit rate is reduced 0.05 bit/pixel (i.e., 10% decreased).

(a)

(b)

Figure 4: Decoded Lena image based on (a) FS-FBC scheme: 33.9 dB at 0.42 bpp (b) modified FBC scheme: 34.0 dB at 0.47 bpp.

3. CONCLUSION

In this paper, based on the similar idea of designing the state codebook in FSVQ, we propose the FS algorithm for FBC scheme to determine the fractal code. With the computer simulation, the proposed adaptive FS-FBC scheme provides a decoded Lena image with 33.9 dB at 0.42 bit/pixel. The simulation results show that we obtain a superior performance among other fractal image coding schemes. Since we don't reduce the bit rate in the case of the domain pool design and only reduce little bit rate in the cases of the contrast scaling factor and isometry, it is possible to reduce more bit rate if we can design a better next-state function for the determination of the domain pool, contrast scaling factor, and isometry. Our future work will focus on designing better next-state functions for FS-FBC scheme to obtain a lower bit rate.

4. ACKNOWLEDGEMENT

This research was supported by the National Science Council, Taiwan under the contract NSC-85-2213-E-194-030.

References

[1] Y. Fisher, editor, *Fractal Image Compression: Theory and Application*, Springer-Verlag, New York, 1995.

[2] A.E. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations," *IEEE Trans. on Image Processing*, vol. 1, no. 1, pp. 18–30, January 1992.

[3] A.E. Jacquin, "Fractal image coding: a review," *Proceedings of the IEEE*, vol. 81, pp. 1451–1465, October 1993.

[4] H.T. Chang and C.J. Kuo, "Fractal block coding using simplified finite-state algorithm," *SPIE's Symposium on Visual Communication and Image Processing'95*, vol. 2501, no.3, pp. 536–544, Taiwan, May 1995

[5] H.T. Chang and C.J. Kuo, "An improved scheme for fractal image coding," *IEEE 1995 International Symposium on Circuits and Systems*, vol. 3, pp. 1624–1627, Seattle, May 1995

[6] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Chapter 14, Kluwer Academic Publishers, Taiwan, 1992.

[7] H.T. Chang and C.J. Kuo, "Advances in fractal block coding scheme for digital images," Submitted to *Signal Processing: Image Communications*, 1996