ELSEVIER

Pattern Recognition Letters 19 (1998) 205–215

Pattern Recognition Letters

Color image compression using quantization, thresholding, and edge detection techniques all based on the moment-preserving principle [1]

Chen-Kuei Yang [a,2], Wen-Hsiang Tsai [b,*]

[a] *Department of Information Management, Ming Chuan University, 250, Sec. 5, Chung Shan N. Rd., Shih-Lin, Taipei 111, Taiwan, ROC*
[b] *Department of Computer and Information Science, National Chiao Tung University, HsinChu 300, Taiwan, ROC*

Received 14 May 1997; revised 27 October 1997

Abstract

A new approach to color image compression with high compression ratios and good quality of reconstructed images using quantization, thresholding, and edge detection all based on the moment-preserving principle is proposed. An input image with 24 bits per pixel is quantized into 8 bits per pixel using a new color quantization method based on the moment-preserving principle. The quantized image is then divided into $n \times n$ non-overlapping square blocks. Two representative colors for each block are computed by moment-preserving thresholding. A bit-map is then generated, consisting of 0s and 1s indicating whether the block pixels are assigned to the first color or the second according to the Euclidean distance measure. A moment-based edge detector is performed further on the bit-map of each non-uniform block. The two parameters l and θ of a line edge with the equation of $x \cos\theta + y \sin\theta = l$ are obtained. The image is finally coded with a codebook of a 256-color palette; a 1-bit indicator for each block which specifies whether the block is uniform or not; an 8-bit color index for a uniform block, or two 8-bit color indices, a 3-bit index for θ, and a 2-bit or 3-bit index for l for a non-uniform block. An average compression ratio of 22.49 or 33.32 can be obtained for 4×4 or 5×5 image blocks, respectively. Experimental results show the feasibility and efficiency of the proposed approach for color image compression.

Keywords: Color image compression; Color quantization; Bit-map; Moment-preserving principle

1. Introduction

The objective of an image compression technique is to remove as much redundant information as possible without destroying the image integrity. These two goals, however, are mutually conflicting. In a digital true-color image, each color component is quantized with 8 bits, and so a color is specified with 24 bits. As a result, there are 2^{24} possible colors for the image. However, the human vision system cannot differentiate so many colors. Furthermore, a color image usually contains a lot of data redundancy and requires a large amount of storage

* Corresponding author. E-mail: whtsai@cis.nctu.edu.tw.
[1] This work was supported partially by National Science Council, Republic of China under Grant NSC83-0408-E009-010.
[2] E-mail: ckyang@mcu.edu.tw.

0167-8655/98/$19.00
PII S0167-8655(97)00166-9

space. In order to lower the transmission and storage cost, image compression is desired, in which color quantization to reduce the number of possible colors is usually included. In this study, the moment-preserving principle is used in the image compression procedure to keep the quality of the resulting image good and acceptable for the human vision system.

The standard block truncation coding (BTC) algorithm is a simple block-based image compression algorithm first developed by Delp and Mitchell (1979) that preserves the block mean and the block standard deviation. The absolute moments BTC (Lema and Mitchell, 1984), upper and lower mean BTC (Udpikar and Raina, 1985), and adaptive BTC (Hui, 1990) were suggested to improve the quality of coded images. Some other modified or hybrid BTC algorithms have since been proposed to further reduce the bit rate either concerning the mean vector (a,b) (Delp and Mitchell, 1991), or the bit-map (Arce and Gallagher, 1983; Zeng et al., 1992; Udpikar and Raina, 1987), or both (Zeng and Neuvo, 1993; Wu and Coll, 1991; Efrati et al., 1991), or using the variable block size with hierarchical technique (Oshri et al., 1993; Roy and Nasrabadi, 1991; Wu and Coll, 1993). All of them use the moment-preserving principle first and then apply other techniques to truncate the image data. These algorithms were originally designed for gray-scale images. However, we can apply these methods separately on each color plane of color images, and then merge the resulting triple set mean vectors and bit-maps in some way. Therefore, reducing the three planes of bit-maps becomes very important in compressing color images. Wu and Coll (1992) used a single bit-map to quantize all the three color planes. This means that only one out of three bit-maps need be preserved. Kurita and Otsu (1993) used the mean vector and the covariance matrix of color vectors to compute the principal score for the pixels in an image block, and classified the pixels in the block into two classes. Two mean vectors and a bit-map are preserved.

In this study, a color image compression approach yielding high compression ratios and good reconstructed image quality is proposed. The approach consists of three steps, namely, quantization, thresholding, and edge detection, which are all based on the moment-preserving principle. First, an input color image with 24 bits per pixel is quantized into 8 bits per pixel. The quantization process is based on the moment-preserving thresholding technique (Tsai, 1985). The color histogram is repeatedly sub-divided into smaller and smaller boxes, and two color values are computed automatically as two representative palette colors for every two separated boxes. A 3D lookup index table is constructed for use in pixel mapping. As a result, the computation in the quantization process is fast. The output is a limited-color image as is desired. Second, the quantized image is divided into $n \times n$ non-overlapping square blocks. Each block is requantized into two representative colors (R_1,G_1,B_1) and (R_2,G_2,B_2) by applying the moment-preserving thresholding technique to each color component. A bit-map is then generated, consisting of 0s and 1s indicating whether the block pixels are assigned to X_1 or X_2 where $X = R$, G or B. The criterion of pixel assignment is based on the Euclidean distance between the original color vector and the two representative color vectors. Third, a moment-based subpixel edge detection technique (Lyvers et al., 1989) is performed on the bit-map of each non-uniform block. The purpose is to divide the block into two regions separated by a line edge. This is reasonable, as demonstrated by our experimental results, because the block is small. The line edge is described by the equation $x \cos\theta + y \sin\theta = l$ which can be computed using analytic formulas, where l and θ are the intercept and the angle of the line. The values of l and θ are assumed to take certain specified values because the block size is small so that the number of possible edge orientations is limited. Thus, a total of no more than 6 bits of indices are found sufficient to code the two parameters l and θ. Furthermore, the values of l and θ need not be included in the transmission cost because these values are predefined and fixed if the block size is given. The image is finally coded with a codebook of 256 color palettes, one 1-bit indicator for each block which specifies whether the block is uniform or not, and one or two 8-bit indices of the color palette according to the block's uniformity. When the block is not uniform, additionally required are one 3-bit index for θ, and one 2-bit or 3-bit index for l for 4×4 or 5×5 window size, respectively. An average compression ratio of 22.49 or 33.32 can be obtained for a 4×4 or 5×5 image block, respec-

tively. Experimental results show the feasibility and efficiency of the proposed approach for color image compression.

The remainder of this paper is organized as follows. A brief review of moment-preserving thresholding and moment-base subpixel edge detection is first given in Section 2. In Section 3, the proposed color image quantization and compression methods are presented. Several experimental results showing the feasibility of the proposed approach are described in Section 4. Finally, some conclusions are given in Section 5.

2. Review of moment-preserving thresholding and subpixel edge detection

2.1. Moment-preserving thresholding

Given a gray-scale image f with T pixels whose gray value at pixel (x,y) is $f(x,y)$, the ith gray moment of f is

$$m_i = \frac{1}{T}\sum_x \sum_y f^i(x,y), \quad i = 0,1,2,\ldots. \tag{1}$$

Let O be an operator applied to the input image f, and g be the output image. If the ith moment of g is set equal to that of f, then O is said to preserve the ith moment of the input data in the output data. By definition, we see that if an operator preserves more moments, then more information of the input image will be retained in the output image.

Suppose that it is desired to threshold f into one with two levels h_1 and h_2 (with p_1 and p_2 being the fractions of pixels with h_1 and h_2, respectively). The algorithm by Tsai (1985) selects a threshold value in such a way that if all the pixels with below-threshold gray value in f are replaced by gray value h_1 and all the pixels with above-threshold gray value in f are replaced by gray value h_2, then the first three moments of image f are preserved in the resulting bi-level image g. The concept can be expressed as follows:

$$p_1h_1 + p_2h_2 = m_1, \quad p_1h_1^2 + p_2h_2^2 = m_2,$$
$$p_1h_1^3 + p_2h_2^3 = m_3, \quad p_1 + p_2 = 1, \tag{2}$$

where the left-hand terms in each equality of (2) is a moment of g and the corresponding right-hand term is a moment of f. To find the desired threshold value t, the equalities of (2) are first solved to obtain p_1, p_2, h_1 and h_2. Then, t is chosen as the gray value closest to the p_1-tile of the histogram of f. After t, h_1 and h_2 are computed, f can be bi-levelly thresholded into g as follows:

$$g(x,y) = \begin{cases} h_1 & \text{if } f(x,y) < t, \\ h_2 & \text{if } f(x,y) \geqslant t. \end{cases} \tag{3}$$

2.2. Subpixel edge detection

The subpixel edge detector proposed by Lyvers et al. (1989) is also based on the mass-moment preserving principle and is reviewed here. A continuous two-dimensional edge specified by h, k, l and θ is shown in Fig. 1. The edge is defined to lie within a unit circle. The mass moments of a two-dimensional function $f(x,y)$ are given by

$$M_{pq} = \int\int x^p y^q f(x,y)\,\mathrm{d}y\,\mathrm{d}x. \tag{4}$$

A rotation of the circular window by $-\theta$ will align all moments containing edge information along the x-axis. The parameters h, k and l can be solved along the x-axis independently of θ. A rotation of

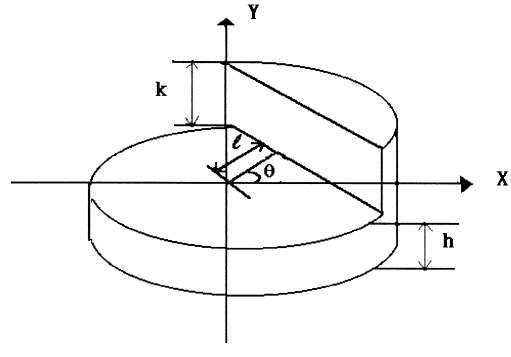

Fig. 1. Two-dimensional ideal edge model characterized by four parameters h, k, l and θ.

the window by an arbitrary angle ϕ transforms the original mass moments into

$$M'_{pq} = \sum_{r=0}^{p}\sum_{s=0}^{q}\binom{p}{r}\binom{q}{s}(-1)^{q-s} \times(\cos\phi)^{p-r+s}(\sin\phi)^{q+r-s}M_{p+q-r,r+s}. \quad (5)$$

The parameters h, k and l can be obtained as follows:

$$l = \frac{4M'_{20} - M'_{00}}{3M_{10}}, \quad (6)$$

$$k = \frac{3M_{10}}{2\sqrt{(1-l^2)^3}}, \quad (7)$$

$$h = \frac{1}{2\pi}\left[2M_{00} - k\left(\pi - \sin^{-1}l - 2l\sqrt{1-l}\right)\right], \quad (8)$$

where the values of the mass moments can be obtained from (4), the rotated mass moments also can be obtained from (5) and their relationship can be expressed in the following equations. The detail process can be seen in (Lyvers et al., 1989).

$$M'_{00} = M_{00}, \quad (9)$$

$$M'_{10} = \sqrt{M_{01}^2 + M_{10}^2}, \quad (10)$$

$$M'_{20} = \frac{M_{10}^2 M_{20} + 2M_{01}M_{10}M_{11} + M_{01}^2 M_{02}}{M_{01}^2 + M_{10}^2}, \quad (11)$$

$$M'_{02} = \frac{M_{10}^2 M_{02} - 2M_{01}M_{10}M_{11} + M_{01}^2 M_{20}}{M_{01}^2 + M_{10}^2}. \quad (12)$$

3. Proposed approach to color image compression

3.1. Proposed color quantization method

A color image can be defined as follows:

$$f: M \times M \to C \subseteq \Psi, \quad (13)$$

where $\Psi = \{(r,g,b) \mid 0 \leqslant r,g,b \leqslant 255\}$ is the RGB color space, $(x,y) \in M \times M$ are the spatial coordinates of a pixel, M is the integer set, and $C = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_N\}$ is a set of colors used in the image. There are $256 \times 256 \times 256$ combinations of red, green and blue components. It puts heavy burden to deal with so many colors! By color quantization, not only the complexity can be reduced but the compression rate is also increased. A quantized image may be regarded as a mapping defined by

$$q: M \times M \to R \subseteq \Psi, \quad (14)$$

where $R = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_k\}$ is a set of representative colors used in the quantized image.

According to the above definition, the process of color image quantization basically can be divided into two major steps. The first step, called color palette design, selects the best match set of colors for a specific image. The second step, called pixel mapping, associates each pixel of the image with a best match color in the color palette to yield an image with the highest quality. The optimization goal is to make the perceived difference between the original image and its quantized version as small as possible. It is very difficult to formulate a definitive solution to meet this goal in terms of perceived image quality. In fact, there is no good objective criterion available for measuring the perceived image quality. In this study, the mean absolute error is used to measure the difference between the original image f and its quantized reproduction q. Accordingly, the average quantization error is defined as follows:

$$d_M(f,q) = \frac{1}{T}\sum_{(x,y)\in M\times M} |f(x,y) - q(x,y)|, \quad (15)$$

where T is the total number of pixels of the image.

3.1.1. Subsampling for histogram creation

A histogram gives the relative frequency of occurrences of colors in an image. The original color image has 24 bits per pixel. There are thus 2^{24} possible colors. To save memory space and computation time, it is desirable to reduce the histogram resolution. Heckbert (1982) has suggested that the required resolution be at least 15 bits per pixel or 5 bits for each color component.

In our approach, to speed up histogram creation, not only the histogram resolution is reduced but also only the even-row-odd-column pixels are taken into

account for the histogram. Hence, only a quarter of the original image pixels are subsampled. The corresponding histogram was found still a good approximation of the color distribution of the original image.

3.1.2. Design of color palette

Many algorithms have been proposed for the design of color palettes. The popularity algorithm (Heckbert, 1982) selects the N most frequently occurring colors in the image as the representative colors according to the image color distribution. In order to prevent the concentration of too many colors in one neighborhood, an improved popularity algorithm was proposed by Braudaway (1986). When a most frequently occurring color is selected, the frequency count of its neighboring colors is reduced. The median cut algorithm (Heckbert, 1982) uses a splitting technique to repeatedly sub-divide the color histogram into smaller and smaller boxes which contain approximately equal numbers of color occurrences, and pick the mean values of the boxes as the representative palette colors. The mean-split algorithm (Gentile et al., 1990) splits the box at the mean value instead of at the median, and recomputes the low mean and high mean for the two separated boxes as the representative palette colors. The variance-based algorithm (Wan et al., 1990) is the same as the median cut algorithm except that it splits the box whose color distribution has the largest values of variances, and the centroids of the sub-boxes are chosen as the representative palette colors. The maxmin algorithm (Houle and Dubois, 1986) tries to minimize the distances between the selected and the original image colors. The initial P palette colors are selected by the frequencies of occurrences which exceed a predetermined threshold. The choice of the K representative colors is recursively made where $K < P$. One choice for the initial color value could be the most frequently occurring color found in the image to be quantized. The new representative color value, selected from the unused colors, is the one whose minimum distance to the representative colors selected thus far is maximum. There are still a lot of other algorithms (Balasubramanian et al., 1994; Dixit, 1991; Orchard and Bouman, 1991; Goldberg, 1991; Wu and Witten, 1985; Wu, 1992) for the design of color palettes, but they are less relevant to this study and so are not described here.

Table 1
The comparative quantization errors in terms of MAE values of the proposed method, the median cut method, and the mean split method

	Proposed method	Median cut	Mean split
Lena	4.0595	4.4219	4.0472
Pepper	4.5827	5.7426	4.5178
Jet	4.0192	4.5178	4.2434
House	4.1007	4.7357	4.4269
Mandrill	6.5468	6.7324	6.5123
Candy	3.9946	4.9447	4.1294

In our approach, the entire color space is regarded initially as a single cube whose representative color is the mean vector of the R, G and B components. The histogram resolution is reduced to 5 bits per color component and the even rows and odd columns of the image pixels are subsampled. The variances of the color components are computed. The color plane with the largest variance is split and the color space becomes two separated boxes. Moment-preserving thresholding using Eqs. (2) and (3) is employed to compute two representative color values h_1 and h_2 for the two boxes and to choose as the cut point the value of the distribution closest to the p_1-tile of the corresponding histogram. Splitting of the color planes is repeated until the desired number of representative colors are achieved. $K-1$ times of splitting are required if K representative palette colors are needed. After the splitting process is completed, all the color coordinates in each box are regarded as identical and are represented by the representative color value computed by the moment-preserving thresholding method. Finally, all the representative colors are

Table 2
The comparative computation times (in seconds) of the methods mentioned in Table 1 with 4, 8, 16, 32, 64, 128 and 256 colors, respectively

Colors	Proposed method	Median cut	Mean split
4	4.81	5.38	5.45
8	4.93	5.49	5.51
16	5.22	5.70	5.79
32	5.78	6.13	6.67
64	6.52	6.92	7.74
128	7.98	8.48	9.19
256	10.19	11.07	11.93

collected to form a color palette and are stored in a one-dimensional array as a color map. The computation time of the proposed method is less than that of the maxmin, the mean-split, and the variance-based algorithms because some steps of the proposed method are performed using analytic formulas. The quantization error is less than that of the median-cut algorithm. These phenomena can be seen from Tables 1 and 2.

3.1.3. Pixel mapping

When a palette has been designed, the remaining step is to map the original color of each pixel in the input image to their best match in the color palette. Several techniques for this purpose have been introduced in the literature (Heckbert, 1982; Balasubramanian et al., 1994; Goldberg, 1991). The simplest way is to compute the distance between the original color vector and all of the representative color vectors, and then choose the one with the minimum distance. This exhaustive search method was proposed by Heckbert (1982). But the computation is slow. A locally sorted search method was also reported in (Heckbert, 1982). The computation time is reduced but it needs additional time to create a list of color vectors and sort the list by the distance key. The list is made by eliminating the representative color vectors whose distances to the centre of the box is above a threshold. Other faster methods include binary tree search (Orchard and Bouman, 1991) and $k-d$ tree search (Friedman et al., 1977). These algorithms are based on the arrangement of the K representative colors in a binary tree structure. By a proper arrangement of the tree, a search time of $O(\log K)$ can be achieved.

In our approach, after the color palette is designed, each box's region in the 3D histogram contains a set of color values which are represented by a single color. Each representative color is included in the color map mentioned previously. Regarding each color value in the box as a cell, we fill up all the cells in each box with identical pointers which all point to the corresponding representative color in the color map. After the cells in all boxes in the 3D histogram are filled with pointers, the histogram may be regarded as a lookup index table. The color value of each input pixel then can be mapped to the corresponding cell in the 3D lookup index table, and in turn the pointer in the lookup table can be retrieved and used to find the corresponding representative color in the color map. This is very convenient for pixel mapping because no search is needed. This direct pixel mapping method is shown in Fig. 2. Some experimental results and performance evaluations of the proposed method are given in Section 4.

3.2. Color image compression

After the input color image is quantized into 256 colors using the proposed quantization method, the image is divided into $n \times n$ non-overlapping blocks. Each block is requantized in this study into two colors by applying the moment-preserving thresholding to each color component again. Each image

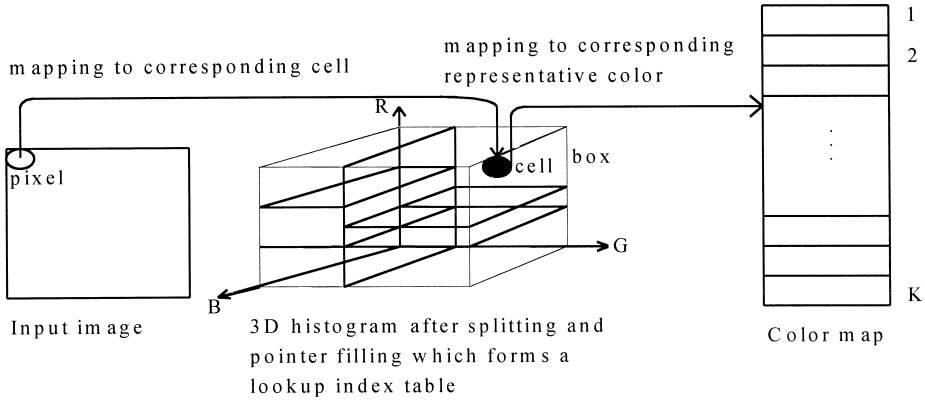

Fig. 2. The process of pixel mapping for reproducing a quantized image.

Table 3
The predefined values of l and θ for 4×4 and 5×5 window size

4×4				5×5				
l	-0.75	0.25	0.25	0.75	-0.8	-0.4	0	0.4	0.8
4×4 and 5×5									
θ	$-\pi/2$	$-\pi/3$	$-\pi/6$	0	$\pi/6$	$\pi/3$	$\pi/2$		

block can thus be coded with two indices of colors and one $n\times n$ bits bit-map. However, the two resulting color vectors may not be found in the 256-color palette. Thus, the minimum value of the Euclidean distance is used as a criterion to select the best match color in the 256-color palette. The resulting bit-maps still occupy more than 50% of the output codes. In order to increase the compression ratio, the moment-based edge detector (Lyvers et al., 1989) reviewed in Section 2.2 is performed on the bit-map. Hence, a non-uniform image block can be coded with two indices of colors and two edge parameters instead of the whole bit-map. And a uniform block can be coded with just a single index of color. Furthermore, since the block size is small, the range of the possible orientations θ of an edge in an image block is limited and so is the range of the intercept value l. The values of these two edge parameters can thus be fixed to certain specified values. Then, an edge block can be coded with two indices of colors and two indices of specified edge parameter values.

From our experimental experience, in general there are very few cases in which all the values of the bits are one or zero in the bit-map. However, the variation of the bit values in a 4×4 or 5×5 block will become noticeable when there exist four or more bits which form a cluster and show a shape of line and edge. According to this concept of visual discontinuity, a block may be assumed uniform when the value of the sum of all the bit values in the bit-map is greater than $n\times n-4$ or less than 4. Note that as the number of uniform blocks increases, the compression ratio is improved.

The difference between two edges with very close directions is not perceivable when the image resolution is high and the window size is small. Also, the number of the possible orientations of an edge in a small image block is limited. Therefore, the values of θ and l can be predefined and fixed to be some specified values. The values of l are predefined to be the distances from the center of the block to those pixels along the x-axis in the block, and the values of θ are selected to be those with the increments of $\pi/6$. These values of θ and l in a 4×4 and 5×5 circular window are shown in Table 3. Because these values are predefined and fixed, it is not necessary to count them in the coding cost.

A 256-color palette codebook needs $3\times5\times256$ bits. Each uniform block is coded with $1\times8+1=9$ bits and each edge block is coded with $2\times8+1+2+3=22$ bits or $2\times8+1+2\times3=23$ bits for 4×4 or 5×5 window size, respectively. the overall compression ratio so can be computed as follows:

$$r=\frac{M\times N\times3\times8}{256\times3\times5+b_0\times9+\left(\frac{M\times N}{n\times n}-b_0\right)\times b}, \tag{16}$$

where $M\times N$ is the image size, b_0 the number of uniform blocks, $n\times n$ the block size, and $b=5$ or 6

Table 4
The values of MAE and the compression ratios of the resulting images mentioned in Table 1 with 4×4 and 5×5 block sizes, respectively

Images	4×4 block size		5×5 block size	
	MAE values	Compression ratios	MAE values	Compression ratios
Lena	7.9226	22.06	8.4477	31.09
Pepper	8.9569	21.74	10.1498	30.30
Jet	8.1365	23.70	8.8268	32.79
House	5.2712	24.28	6.0135	37.69
Mandrill	16.4548	19.18	19.3235	28.07
Candy	4.9882	23.98	5.4691	39.98
Average		22.49		33.32

for the indices of the edge parameters with 4×4 or 5×5 window size, respectively.

4. Experimental results

The proposed algorithms have been tested on an IRIS Indigo workstation on several color images. Each color image has 24 bits/pixel and is 512×512 in size. We use the mean absolute error (MAE) as the criterion for the performance evaluation of image quantization and compression. Table 1 shows the comparative quantization errors in terms of the MAE values of the proposed method, the median cut method, and the mean split method, respectively. for each method, the images ''Lena'', ''pepper'', ''jet'', ''house'', ''mandrill'' and ''candy'' are tested and are quantized into 256 colors. Table 2 shows the comparative computation times of the methods mentioned in Table 1. From Tables 1 and 2, several facts can be observed. First, the quantization results of the proposed method are better than that of the median cut method. Second, the computation time of the proposed method is the least among the methods mentioned above. Table 4 shows the MAE values of the reconstructed images and the compression ratios of the images mentioned above with 4×4 and 5×5 window sizes, respectively. Fig. 3 shows the standard images of ''Lena'', ''pepper'' and ''mandrill'' of size 512×512 in top, middle and bottom, respectively. Fig. 4 shows the results of the proposed quantization method with 256 colors using the images shown in Fig. 3. Figs. 5 and 6 show the reconstructed results of the proposed compression approach using the images shown in Fig. 3 with 4×4 and 5×5 window sizes, respectively. From Fig. 4, we can find that the image quality is almost the same as the original. It is difficult to find differences between the original images and the reproduced results by the human vision system. However, little differences can be distinguished by the computed MAE values (see Table 1). It can be seen from Figs. 5 and 6 and Table 4 that the reconstructed image quality is still good and acceptable. Only some slightly zigzag edges and a litter color difference in regions with gradual color changes can be seen. This is due to the uses of BTC, fewer bits for

Fig. 3. The original images of ''Lena'', ''pepper'' and ''mandrill'' in top, middle and bottom, respectively.

Fig. 4. The quantization results with 256 colors of the images in Fig. 3.

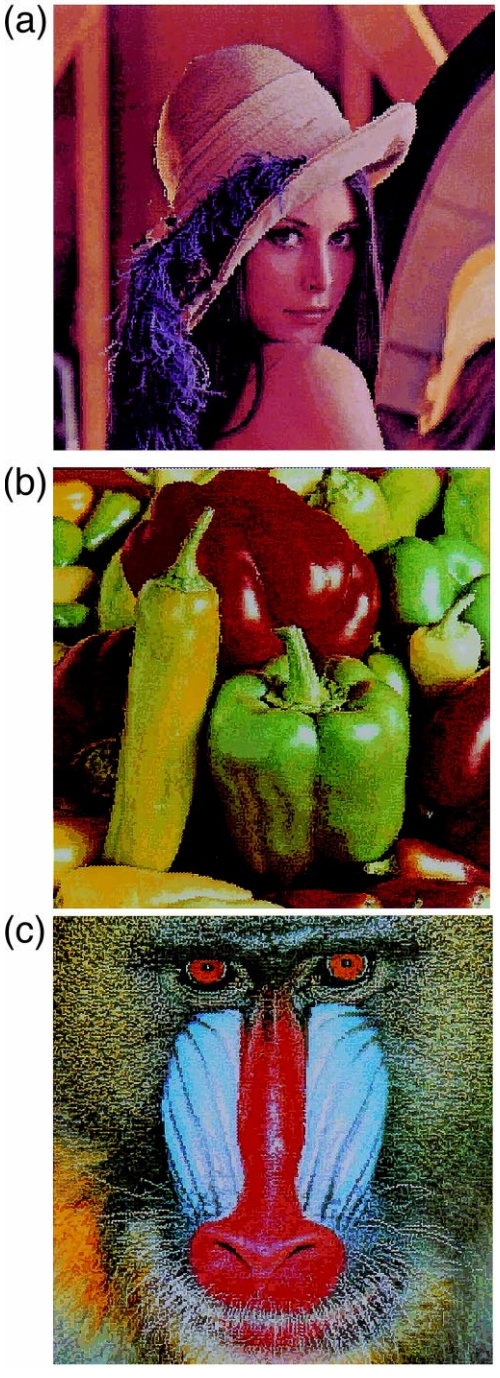

Fig. 5. The reconstructed results of the images in Fig. 3 with 4×4 windows.

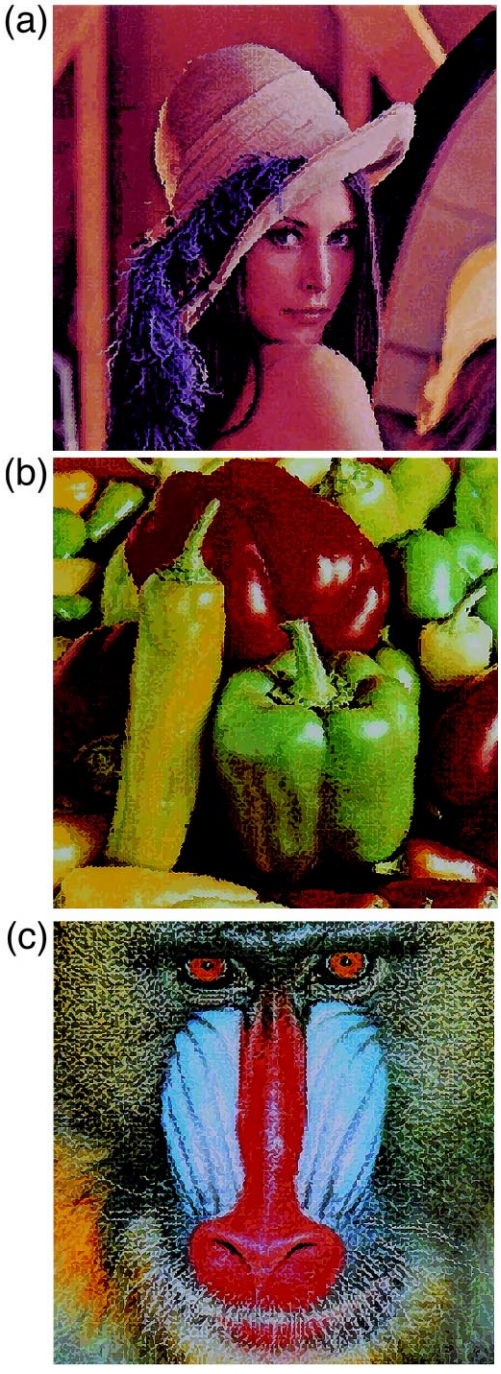

Fig. 6. The reconstructed results of the images in Fig. 3 with 5×5 windows.

the indices of l and θ, as well as the detected line edges only in the blocks.

5. Conclusions

A new color image compression algorithm using quantization, thresholding, and edge detection all based on the moment-preserving principle has been proposed. The major contributions of this study include the following. (1) A new approach to the design of color palettes is proposed. (2) A fast pixel mapping method for color quantization and reproduction is presented. (3) An edge operator is performed on the bit-map to avoid using the whole bit-map. (4) An efficient coding of the edge detection results has been proposed. (5) Reconstructed images with good quality and reasonable compression ratios have been obtained. (6) The computation is fast and thus suitable for real-time applications due to the use of analytic formulas for some of the quantization and compression steps. Some possible improvements and further research topics are as follows.

The human visual system is more sensitive to details in the luminance component than to details in the chrominance component. Hence, the RGB components can be first linearly transformed into the YIQ components, and the luminance and the chrominance components can be treated differently at the time of resampling the histogram with resolution reduction to save the memory space of storing the histogram. But this requires additional time for color model transformation. The size of an image block also need not be fixed. The use of variable block sizes in the compression scheme will yield generally higher compression ratios. The zigzag effect and image distortion in the compressed image can be improved by increasing the number of allowed specified orientations of the detected edges, or by detecting more complicated features in the bit-map such as curves and lines.

References

Arce, G.R., Gallagher, N.C., 1983. BTC image coding using median filter roots. IEEE Trans. Commun. 31 (6), 784–793.

Balasubramanian, R., Bouman, C.A., Allebach, J.P., 1994. Se-

quential scalar quantization of color images. J. Electronic Imaging 3 (1), 45–59.

Braudaway, G., 1986. A procedure for optimum choice of a small number of colors from a large color palette for color imaging. In: Proc. Electronic Imaging '86, San Francisco, CA, pp. 75–79.

Delp, E.J., Mitchell, O.R., 1979. Image compression using block truncation coding. IEEE Trans. Commun. 27, 1142–1335.

Delp, E.J., Mitchell, O.R., 1991. The use of block truncation coding in DPCM image coding. IEEE Trans. Signal Process. 39 (4), 967–971.

Dixit, S.S., 1991. Quantization of color images for display/printing on limited color output device. Comput. Graphics 4 (1), 561–567.

Efrati, N., Liciztin, H., Mitchell, H.B., 1991. Classified block truncation coding–vector quantization: An edge sensitive image compression algorithm. Signal Processing: Image Communication 3, 275–283.

Friedman, J.J., Bentley, J.L., Finkel, R.A., 1977. An algorithm for finding best matches in logarithmic expected time. ACM Trans. Math. Software 3, 209–226.

Gentile, R.S., Allebach, J.P., Walowit, E., 1990. Quantization of color images based on uniform color spaces. J. Imaging Technol. 16 (1), 12–21.

Goldberg, N., 1991. Color image quantization for high resolution graphics display. Image and Vision Computing 9 (5), 303–312.

Heckbert, P., 1982. Color image quantization for frame buffer display. Comput. Graphics 16 (3), 297–307.

Houle, H., Dubois, E., 1986. Quantization of color images for display on graphics terminals. In: Proc. IEEE Global Telecommun. Conf., pp. 1138–1142.

Hui, L., 1990. An adaptive block truncation coding algorithm for image compression. In: Proc. ICASSP 90, Vol. 4, pp. 2233–2236

Kurita, T., Otsu, N., 1993. A method of block truncation coding for color image compression. IEEE Trans. Commun. 41 (9), 1270–1274.

Lema, M.D., Mitchell, O.R., 1984. Absolute moment block truncation coding and its application to color images. IEEE Trans. Commun. 32, 1148–1157.

Lyvers, E.P., Mitchell, O.R., Akey, M.L., Reeves, A.P., 1989. Subpixel measurement using a moment-based edge operator. IEEE Trans. Pattern Anal. Machine Intell. 11 (12), 1293–1309.

Orchard, M.T., Bouman, C.A., 1991. Color quantization of images. IEEE Trans. Signal Process. 39 (12), 2677–2690.

Oshri, E., Shelly, N., Mitchell, H.B., 1993. Interpolative three-level block truncation coding algorithm. Electron. Lett. 29 (14), 1267–1268.

Roy, J.U., Nasrabadi, N.H., 1991. Hierarchical block truncation coding. Opt. Engrg. 30 (5), 551–556.

Tsai, W.H., 1985. Moment-preserving thresholding: a new approach. CVGIP 29, 377–393.

Udpikar, V.R., Raina, J.P., 1985. A modified algorithm for block truncation coding of monochrome images. Electron. Lett. 21 (20), 900–902.

Udpikar, V.R., Raina, J.P., 1987. BTC image coding using vector quantization. IEEE Trans. Commun. 35 (3), 352–356.

Wan, S.J., Prusinkiewicz, P., Wong, S.K.M., 1990. Variance-based color image quantization for frame buffer display. Color Res. Appl. 15 (1), 52–58.

Wu, X., 1992. Color quantization by dynamic programming and principal analysis. ACM Trans. Graphics 11 (4), 348–372.

Wu, X.L., Witten, I.H., 1985. A fast k-means type clustering algorithm. Technical Report 85/197/10, Dept. of CS, U. of Calgary, Calgary, Canada.

Wu, Y., Coll, D.C., 1991. BTC-VQ-DCT hybrid coding of digital images. IEEEE Trans. Commun. 39 (9), 1283–1287.

Wu, Y., Coll, D.C., 1992. Single bit-map Block Truncation Coding of color images. IEEE J. Selected Areas Commun. 10 (5), 952–959.

Wu, Y., Coll, D.C., 1993. Multilevel block truncation coding using a minimax error criterion for high fidelity compression of digital images. IEEE Trans. Commun. 41 (8), 1179–1191.

Zeng, B., Neuvo, Y., 1993. Interpolative BTC image coding with vector quantization. IEEE Trans. Commun. 41 (10), 1436–1437.

Zeng, B., Neuvo, Y., Venetsanopoulos, A.N., 1992. Interpolative BTC image coding. In: Proc. IEEE ICASSP 92, Vol 3, pp. 493–496.