A Common Bitmap Block Truncation Coding for Color Images Based on Binary Ant Colony Optimization

Zhihong Li[1], Qiang Jin[1], Chin-Chen Chang[2,*], Li Liu[1] and Anhong Wang[1]
[1] Institute of Digital Multimedia and Communication,
Taiyuan University of Science and Technology, Taiyuan 030024, China
[e-mail: zy_lzh@sohu.com, 1021524125@qq.com, skies5315@sina.com, wah_ty@163.com]
[2,*] Department of Information Engineering and Computer Science,
Feng Chia University, Taichung 40724, Taiwan
[e-mail: alan3c@gmail.com]
[*] Corresponding Author: Chin-Chen Chang

Received November 19, 2015; revised March 13, 2016; accepted April 6, 2016; published May 31, 2016

Abstract

For the compression of color images, a common bitmap usually is generated to replace the three individual bitmaps that originate from block truncation coding (BTC) of the R, G and B channels. However, common bitmaps generated by some traditional schemes are not the best possible because they do not consider the minimized distortion of the entire color image. In this paper, we propose a near-optimized common bitmap scheme for BTC using Binary Ant Colony Optimization (BACO), producing a BACO-BTC scheme. First, the color image is compressed by the BTC algorithm to get three individual bitmaps, and three pairs of quantization values for the R, G, and B channels. Second, a near-optimized common bitmap is generated with minimized distortion of the entire color image based on the idea of BACO. Finally, the color image is reconstructed easily by the corresponding quantization values according to the common bitmap. The experimental results confirmed that reconstructed image of the proposed scheme has better visual quality and less computational complexity than the referenced schemes.

Keywords: Block truncation coding (BTC), binary ant colony optimization (BACO), common bitmap, image compression

This work has been supported in part by National Natural Science Foundation of China (No. 61272262), Program for New Century Excellent Talent in Universities (NCET-12-1037), International Cooperative Program of Shanxi Province (No. 2015081015), and Scientific and Technological project of Shanxi Province (2015031003-2).

http://dx.doi.org/10.3837/tiis.2016.05.020 **ISSN : 1976-7277**

1. Introduction

With the development of the Internet, digital images are transmitted extensively and used in many fields, such as multimedia systems, electronic publishing, and communications. However, digital images often contain a significant amount of data, which affects the storage and transmission of images. Thus, it is necessary to compress these images before they are stored and transmitted. The goal of compressing image is to remove redundancies in the image so it can be represented by a smaller number of bits with little, if any, degradation in the quality of the image.

Generally, image compression techniques are classified into lossless compression and lossy compression. Huffman coding, arithmetic coding [1-2], LZW coding and run length coding are lossless image compression methods, and they provide excellent visual quality of restored image, but their performance on compression ratio is low. However, lossy compression methods, such as wavelet transformation coding [3], vector quantization [4], fractal coding [5], sub-band coding [6], and block truncation coding [7] can maintain both good visual quality of the restored image and a high compression ratio.

As one of the lossy compression methods, block truncation coding (BTC) first was proposed by Delp and Mitchell [8]. Because of its advantages, including simplicity, fault tolerance, and high compression efficiency, BTC has been used extensively in the field of image compression and in many other fields, such as data hiding [9], image retrieval [10], and image sharing [11]. In this method, first, the image is divided into several non-overlapping blocks, and, second, the first two moment estimations are preserved for the purpose of selecting two levels for each block. The bit plane is generated by considering the mean value of the block as a threshold to divide the block into two levels. Third, the corresponding bit plane, the mean value of the block and the standard deviation of the block can be used to restore the image. Later, many schemes were proposed to improve the performance of this method mainly in two aspects, i.e., visual quality and bit rate. The absolute moment BTC [12] and adaptive BTC [13] are proposed to improve the visual quality of the restored image. Some other schemes, such as the prediction-based BTC method [14] and the pattern fitting method [15], were proposed to reduce the bit rate. However, these methods consider only two levels in the bit pattern. Three-level BTC algorithms [16-17] also have been proposed to improve the visual quality of the restored image. All of the proposed approaches originally were designed for gray images.

BTC method also can be used to compress color images because each channel in the color image corresponds to a grayscale image. It is well-known that color images are composed of three channels, i.e., R, G and B. Obviously, color images contain a lot of data, i.e., about triple the data in grayscale images. In order to remove redundant information efficiently and improve the compression ratio, it is important to be able to determine a suitable common bitmap to replace the three bitmaps generated from the three individual channels. Some approaches have been proposed to generate common bitmaps. In 1992, Wu and Coll [18] proposed a single-bitmap BTC method, which preserved the spatial details in the image's content and had low computational complexity. In this scheme, the weighted plane (W-plane) must be generated by properly weighting the corresponding R, G, and B channels. Then the W-plane is used to quantize the R, G, and B planes, thereby achieving better performance regarding the mean square error (MSE). Later, Yang et al. [19] proposed another single-bitmap method based on the moment-preserving principle and BTC. In this scheme, RGB images can be compressed directly without any transformation. The experimental results showed that this scheme achieved a higher compression ratio and better quality of the reconstructed image than previous schemes. In 2002, Chang and Wu [20] proposed the gradual search algorithm for one single-bitmap BTC (GSBTC), which generates an effective common bitmap that provides acceptable quality of the recovered image while maintaining low complexity computation. However, the common bitmap generated by the above methods are not the best possible, because they do not consider the minimized distortion of the entire color image. In 2008, Chang et al.'s scheme used a genetic algorithm (GA) to produce a near-optimal common bitmap based on the human visual system [21]. In order to further improve the visual quality of reconstructed image, cat swarm optimization (CSO) was adopted in Cui et al.'s scheme to generate the near-optimal common bitmap [22]. Although the GA and CSO had good global search ability, their computations were complicated and costly.

Based on the above analyses, in this paper, we propose a near-optimized common bitmap scheme based on Binary Ant Colony Optimization (BACO), which has low computation complexity, to produce a BACO-BTC scheme. First, we use the traditional BTC to compress each channel of a color image and produce three individual bitmaps and three pairs of quantization values. Then BACO is performed to determine a near-optimal common bitmap under the constraint of minimized distortion of the entire color image. The experimental results indicate that our method produces a recovered image with better visual quality and that the computational complexity is lower than that of other schemes.

The rest of this paper is organized as follows. Section 2 reviews some related work. In Section 3, our scheme, named BACO-BTC, is presented in detail. Some experimental results are discussed in Section 4, and our conclusions are presented in Section 5.

2. Related Work

BTC is the backbone of our method, and we used BACO to generate a near-optimal common bitmap. GSBTC is a similar method that can be treated as a reference. In this section, these important techniques are reviewed in details.

2.1 Block truncation coding

Block truncation coding (BTC) was proposed for grayscale-image compression. The basic steps of BTC can be summarized as follows.

Step 1. A grayscale image is divided into non-overlapping blocks with the size of m×n. Then the mean value u of each block is computed by using Eq. (1).

$$u = \frac{1}{m \times n} \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij}, \tag{1}$$

where x_{ij} is the pixel value located at the position (i, j) in each block.

Step 2. The pixels of the block are classified into two sets. If $x_{ij} \geq u$, we put the pixel into set H; otherwise, the pixel is placed into set L. We denoted the number of pixels in set H and set L as n_H and n_L, respectively. So the high value x_H and the low value x_L are defined as:

$$\begin{cases} x_H = \dfrac{1}{n_H} \sum x_{ij}, & if \ x_{ij} \geq u \\ x_L = \dfrac{1}{n_L} \sum x_{ij}, & otherwise \end{cases}. \tag{2}$$

Step 3. The bitmap $M = \{m_{ij} \mid m_{ij} \in \{0,1\}, 1 \leq i \leq m, 1 \leq j \leq n\}$ is created as follows:

$$m_{ij} = \begin{cases} 1, & if \ x_{ij} \geq u \\ 0, & otherwize \end{cases}. \tag{3}$$

Step 4. Finally, the bitmap M, the high value x_H and the low value x_L are combined together to reconstruct the original pixel values for each block. If the value of m_{ij} is 1, the value of the restored pixel at the position (i, j) would be x_H. Otherwise, the value of the restored pixel would be x_L.

Fig. 1 shows an example of BTC encoding and decoding. A 4×4 image block is shown in **Fig. 1(a)**. The mean value u of the image block is computed as 107 and the high value x_H and the low value x_L are computed as 182 and 62, respectively. The bitmap for this image block is generated, as shown in **Fig. 1(b)** and the reconstructed image block is shown in **Fig. 1(c)**.

123	95	220	30
76	252	89	44
110	53	51	153
82	78	22	234

(a)

→

1	0	1	0
0	1	0	0
1	0	0	1
0	0	0	1

(b)

→

182	62	182	62
62	182	62	62
182	62	62	182
62	62	62	182

(c)

Fig. 1. Example for BTC encoding and decoding, **(a)** an original image block, **(b)** bitmap, **(c)** the reconstructed image block

2.2 Gradual search algorithm for one single bitmap BTC

First, a color image is divided into non-overlapping blocks and each block is compressed by BTC, producing three bitmaps and three pairs of quantization values for the three channels, i.e., R, G and B. Then, the common bitmap is generated from these three bitmaps, as shown in **Figs. 2(a)**, **(b)**, and **(c)**. If the bits in the same positions of the three bitmaps are equal, we put them into the common bitmap at the corresponding positions and denote them as determined elements. The other bits are called non-determined elements, which are denoted as *, as shown in **Fig. 2(d)**. The initial common bitmap only contains the determined elements, which is shown in **Fig. 2(d)**. The non-determined elements are determined by the following steps [23].

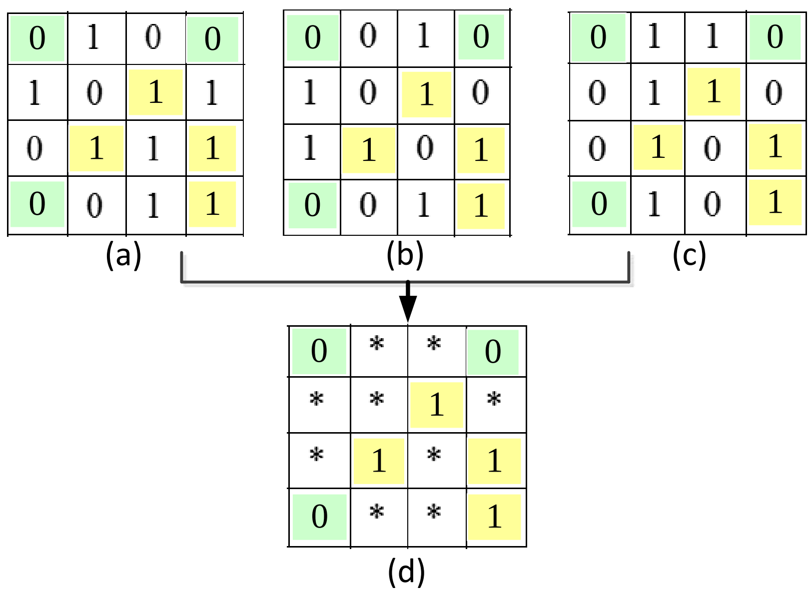


Fig. 2. Example of the initial common bitmap, **(a)** Bitmap of R channel, **(b)** Bitmap of G channel, **(c)** Bitmap of B channel, **(d)** The initial common bitmap

Step 1. Compute the initial mean square error (MSE) for the initial common bitmap as follows:

$$MSE = \sum_{c_{ij}=0} (x_{ij} - x_l)^2 + \sum_{c_{ij}=1} (x_{ij} - x_h)^2 , \qquad (4)$$

where

$$\begin{cases} x_l = \dfrac{1}{n_0} \sum_{c_{ij}=0} x_{ij} \\ x_h = \dfrac{1}{n_1} \sum_{c_{ij}=1} x_{ij} \end{cases} , \qquad (5)$$

and x_{ij} is the pixel value at position (i, j) of the block, c_{ij} is the element at the same position in the common bitmap, n_0 is the number of pixels in the case of c_{ij}=0, and n_1 is the number of pixels in the case of c_{ij}=1.

Step 2. Pick an element from the non-determined elements as the determined element and compute two new MSEs by using Eq. (4) when this element is equal to 1 or 0.

According to the minimum new MSE, determine which value of the element should be selected. Then, put the corresponding value into the common bitmap as the determined element.

Step 3. Repeat Step 2 until all of the non-determined elements have been determined; at this moment, the common bitmap is generated.

2.3 Binary Ant Colony Optimization

Ant Colony Optimization (ACO), proposed by Dorigo and Gambardella [24-25], is a kind of simulated evolutionary algorithm based on the way ants find food. Following the idea of ACO, the BACO algorithm performs competitively in solving the discrete-domain problems because of its unique space structure of random binary search. The BACO's solution is represented by a binary bit string with each node selecting from two possible values, i.e., 0 or 1 [26]. As shown in **Fig. 3**, the ant colony optimization traversal of a binary random network [27] is designed as follows.

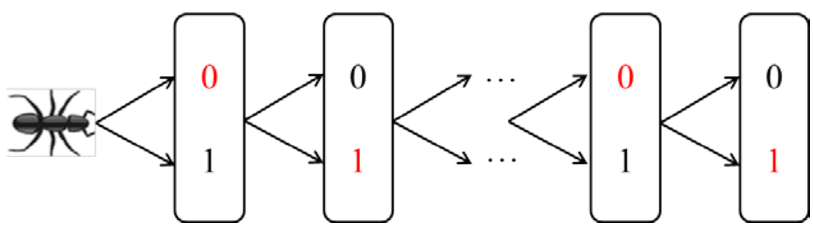


Fig. 3. Binary ant colony network

The basic steps of BACO are summarized as follows.

Step 1. Initialize a population of ants with the same pheromones. The ant selects 0 or 1 according to a probability related to its own pheromone:

$$p_{ij}^{k} = \frac{\tau_{ij}(t)}{\sum_{l\in\{0,1\}} \tau_{il}(t)}, i=1,2,...,n,\ j\in\{0,1\}, \tag{6}$$

Where p_{ij}^{k} is the probability of ant k choosing j=0 or j=1 at node i, and $\tau_{ij}(t)$ is the pheromone value of node i choosing j=0 or j=1.

Step 2. A solution in a binary bit stream is generated after each ant has searched all nodes. The pheromone $\tau_{ij}(t)$ is updated at the end of each solution by using Eq. (7).

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho\Delta\tau, i=1,2,...,n,\ j\in\{0,1\}, \rho\in[0,1], \tag{7}$$

where ρ is a parameter that describes the extent to which the information has evaporated. $\Delta\tau$ is the incremental pheromone.

Step 3. The circulation continues until the end condition is met. In this way, we can obtain an approximate optimum solution.

3. The Proposed Scheme

In this paper, we propose a novel image compression scheme for color images based on BACO. The main idea of our scheme is to generate a near-optimal common bitmap to replace the three individual bitmaps that are generated from different channels by BTC. **Fig. 4** shows the entire flowchart of our algorithm, and the details are described below.

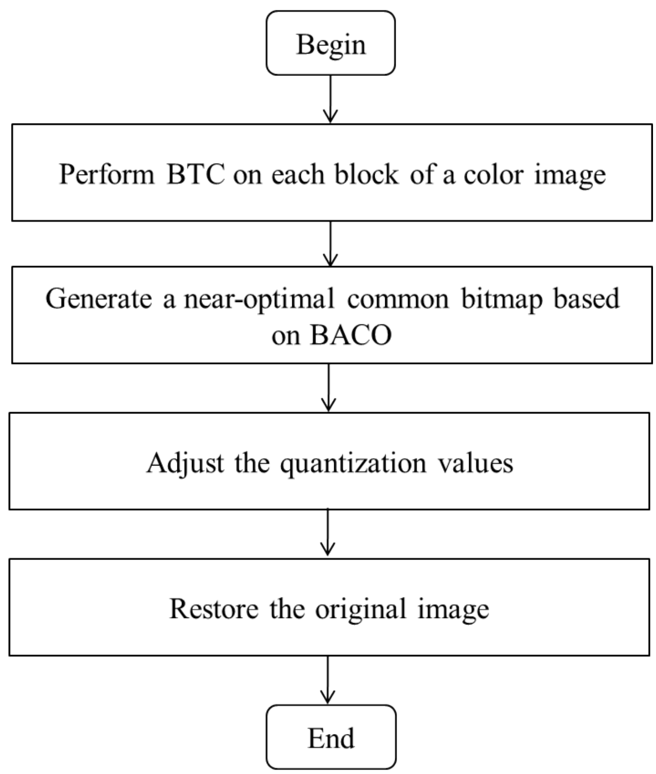


Fig. 4. Flowchart of our algorithm

3.1 Perform BTC on each block of a color image

First, a color image I is divided into a series of non-overlapping blocks with the size $m \times n$. Then every block can obtain three pairs of quantization values for the three channels, i.e., R, G and B, which are denoted as (x_{RH}, x_{RL}), (x_{GH}, x_{GL}), and (x_{BH}, x_{BL}) by using BTC, as is shown in Eq. (8).

$$
\begin{aligned}
x_{RH} &= \frac{1}{n_r} \sum\nolimits_{\mathrm{R}_{ij} \geq u_r} R_{ij}, \\
x_{RL} &= \frac{1}{m \times n - n_r} \sum\nolimits_{\mathrm{R}_{ij} < u_r} R_{ij}, \\
x_{GH} &= \frac{1}{n_g} \sum\nolimits_{\mathrm{G}_{ij} \geq u_g} G_{ij}, \\
x_{GL} &= \frac{1}{m \times n - n_g} \sum\nolimits_{\mathrm{G}_{ij} < u_g} G_{ij}, \\
x_{BH} &= \frac{1}{n_b} \sum\nolimits_{\mathrm{B}_{ij} \geq u_b} B_{ij}, \\
x_{BL} &= \frac{1}{m \times n - n_b} \sum\nolimits_{\mathrm{B}_{ij} < u_b} B_{ij},
\end{aligned} \tag{8}
$$

where u_r, u_g and u_b are the block's mean values of the R, G and B channels; R_{ij}, G_{ij} and B_{ij} are the original pixel values of these three channels, respectively; and n_r, n_g and n_b are the numbers of pixels that are not less than the block's mean value for each channel.

3.2 Generate a near-optimal common bitmap based on BACO

Our goal was to achieve a common bitmap for a color image with minimal distortion of the entire color image. BACO was used to achieve this goal. This process is detailed as follows.

Input: Original color image block BL, three pairs of quantization values, i.e., (x_{RH}, x_{RL}), (x_{GH}, x_{GL}), and (x_{BH}, x_{BL}).

Output: Common bitmap C^* for block BL.

Step 1. Generate a random binary matrix $C = \{C_{ij} \mid C_{ij} \in \{0,1\}, 1 \leq i \leq m, 1 \leq j \leq n\}$ as an initial common bitmap, and compute the initial *MSE* using Eq. (9).

$$
MSE = \sum_{C_{ij}=0} (x_{ij} - x_L)^2 + \sum_{C_{ij}=1} (x_{ij} - x_H)^2, \tag{9}
$$

where x_{ij}=(R_{ij}, G_{ij}, B_{ij}) are the original values from the three channels at the same position. x_H and x_L are the quantization values that correspond to x_{ij} from the three channels. For instance, if x_{ij}=R_{ij}, then x_H=x_{RH} and x_L=x_{RL}.

Step 2. Generate two initial pheromone matrices $\tau_{ij}(0)$ and $\tau_{ij}(1)$ with the same size of the block BL, and each component in these matrices is selected as $\tau_{ij}(k)$=0.5(k$\in$\{0,1\}) because each ant located in matrix C has the same probability to select path "0" or path "1" in the initial state.

Step 3. Select a value of matrix C and change it into the opposite value to generate a temporary matrix. In this matrix, only the selected value is different from the values in the matrix C. Compute the corresponding $newMSE_{ij}$ for this matrix by using Eq. (9).

Step 4. Compute the incremental pheromone $\Delta\tau_{ij}(k)$ according to Eq. (10),

$$\Delta\tau_{ij}(k)=\begin{cases}1/MSE & if\ C_{ij}=k,\\ 1/newMSE_{ij} & otherwise,\end{cases}\quad k\in\{0,1\}, i\in[1,m], j\in[1,n], \tag{10}$$

and update the initial pheromone matrices, i.e., $\tau_{ij}(0)$ and $\tau_{ij}(1)$, respectively, by using Eq. (11).

$$\tau_{ij}(k)=\tau_{ij}(k)+\Delta\tau_{ij}(k), k\in\{0,1\}, i\in[1,m], j\in[1,n], \tag{11}$$

where $\tau_{ij}(k)$ represents the pheromone at position (i, j) for selecting k, $k\in\{0,1\}$.

Step 5. Update the common bitmap according to the following rule.

$$C^{*}_{ij}=\begin{cases}1 & if\ \tau_{ij}(1)>\tau_{ij}(0)\\ 0 & otherwise\end{cases}\quad i\in[1,m], j\in[1,n]. \tag{12}$$

Step 6. Repeat Step 3 through Step 5 until all of the values in matrix C are dealt with. To this point, the near-optimal common bitmap C^* for image block BL is generated. Similarly, all of the image blocks are dealt with in the same way.

Considering that each position of the common bitmap has an independent effect on the distortion of a block, we assigned an ant for each position instead of assigning an ant to search all the positions in order to reduce the time required to determine the proper common bitmap. For a common bitmap, each position has only two choices, 0 or 1. Each ant is initialized to 0 or 1 randomly and an initial common bitmap is generated when all the ants has been initialized. Then select an ant each time and change its value to the opposite. The pheromone of this ant for choosing 0 or 1 is changed by computing new MSE for the corresponding block. By comparing the updated pheromone values of this ant, whether choosing 0 or 1 will be determined for this ant. When the pheromone of all the ants are updated, the near-optimal solution will be generated according to the updated pheromone matrices. Therefore, this process will not get into local optimum.

Fig. 5 shows an example of updating a bit at position (1, 1) of a common bitmap. Assume that MSE=10 and $newMSE_{11}$=5 for this block. **Fig. 5(a)** represents an initial common bitmap C. All of the initial pheromone values are 0.5 at each position. **Fig. 5(b)** is the pheromone matrix $\tau_{ij}(0)$ after updating the pheromone value $\tau_{11}(0)$, and **Fig. 5(c)** is the pheromone matrix $\tau_{ij}(1)$ after updating the pheromone value $\tau_{ij}(1)$. Because $\tau_{11}(1)>\tau_{11}(0)$, the corresponding value at position (1, 1) of the common bitmap is 1, as shown in **Fig. 5(d)**.

0	1	0	1
1	0	0	1
0	1	1	0
0	1	1	0

(a)

0.6	0. 5	0. 5	0. 5
0. 5	0. 5	0. 5	0. 5
0. 5	0. 5	0. 5	0. 5
0. 5	0. 5	0. 5	0. 5

(b) k=0

0.7	0. 5	0. 5	0. 5
0. 5	0. 5	0. 5	0. 5
0. 5	0. 5	0. 5	0. 5
0. 5	0. 5	0. 5	0. 5

(c) k=1

1	1	0	1
1	0	0	1
0	1	1	0
0	1	1	0

(d)

Fig. 5. Example of updating common bitmap, **(a)** The initial matrix C, **(b)** The updated pheromone matrix with k=0, **(c)** The updated pheromone with k=1, **(d)** The updated common bitmap C^*

3.3 Adjust the quantization values

In order to obtain good visual quality, it is necessary to adjust the three pairs of quantization values according to the generated common bitmap C^*. Assume that the number of “1” in C^* is denoted as n_h. The three new pairs of quantization values are denoted as (x^*_{RH} , x^*_{RL}), (x^*_{GH} , x^*_{GL}), and (x^*_{BH} , x^*_{BL}) for the three R, G, and B channels, respectively, as shown in Eq. (13).

$$
\begin{aligned}
x^*_{RH} &= \frac{1}{n_h}\sum\nolimits_{C^*_{ij}=1} R_{ij}, \\
x^*_{RL} &= \frac{1}{m\times n - n_h}\sum\nolimits_{C^*_{ij}=0} R_{ij}, \\
x^*_{GH} &= \frac{1}{n_h}\sum\nolimits_{C^*_{ij}=1} G_{ij}, \\
x^*_{GL} &= \frac{1}{m\times n - n_h}\sum\nolimits_{C^*_{ij}=0} G_{ij}, \\
x^*_{BH} &= \frac{1}{n_h}\sum\nolimits_{C^*_{ij}=1} B_{ij}, \\
x^*_{BL} &= \frac{1}{m\times n - n_h}\sum\nolimits_{C^*_{ij}=0} B_{ij}.
\end{aligned} \tag{13}
$$

3.4 Restore the original color image

The restoration process is the inverse process of the compression process. The original color image can be restored based on the generated common bitmap and the three pairs of quantization values. **Fig. 6** shows an example of restoring an original image block. **Fig. 6(a)** is a generated common bitmap. **Fig. 6(b)** shows the restored image block of the R channel with x^*_{RH}=56 and x^*_{RL}=24. **Fig. 6(c)** shows the restored image block of the G channel with x^*_{GH}=195 and x^*_{GL}=127. And **Fig. 6(d)** shows the restored image block of the B channel with x^*_{BH} =136 and x^*_{BL} = 103.

0	1	1	0
1	0	1	0
0	1	0	1
0	0	1	1

(a)

24	56	56	24
56	24	56	24
24	56	24	56
24	24	56	56

(b)

127	195	195	127
195	127	195	127
127	195	127	195
127	127	195	195

(c)

103	136	136	103
136	103	136	103
103	136	103	136
103	103	136	136

(d)

Fig. 6. Example of restoring an image block, **(a)** the common bitmap, **(b)** The restored image block of R channel, **(c)** The restored image block of G channel, **(d)** The restored image block of B channel

4. Experiments and Discussions

In this section, some experimental results are presented to prove the feasibility of the proposed method. The simulation environment of our experiments was a PC with a 2.1 GHz CPU and 4 GB RAM. **Fig. 7** shows the six 512×512 color test images, i.e., “Airplane,” “Fruits,” “Scenes,” “Peppers,” “Barbara,” and “Boat”, that were used to test our scheme. Three schemes, GSBTC [20], GA-BTC [21] and CSO-BTC [22], were implemented by using MATLAB 2012b to compare their results with those of the proposed scheme.

The peak signal-to-noise ratio (*PSNR*) is used to measure the visual quality between the reconstructed color image and the original color image, and it was defined as follows:

$$PSNR = 10 \times \log_{10} \frac{255^2}{MSE}. \tag{14}$$

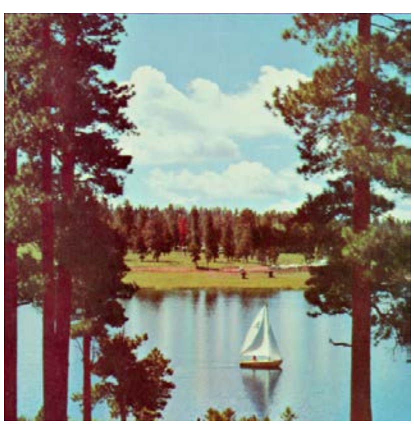
MSE for the color image is shown in Eq. (15).

$$MSE = \frac{1}{w \times h} \sum_{i=1}^{w} \sum_{j=1}^{h} \frac{(R_{ij} - R_{ij}^*)^2 + (G_{ij} - G_{ij}^*)^2 + (B_{ij} - B_{ij}^*)^2}{3}, \tag{15}$$

where w and h are the width and height of the original image, respectively; R_{ij}, G_{ij} and B_{ij} are the values of the three primary colors of the original image and R_{ij}^*, G_{ij}^* and B_{ij}^* are the values of the reconstructed image for the R, G and B channels, respectively.

Airplane

Fruits



Scenes

Peppers

Babara

Boat

Fig. 7. Six 512×512 color test images

Tables 1 and **2** contain the experimental results using different methods, with block sizes of 4×4 and 8×8, respectively. **Table 1** shows that average *PSNR* value of the proposed method for six images is 31.11 dB, which was higher than other schemes that were used for comparison. This illustrates that the proposed scheme obtained better quality recovered images. The data in **Table 2** also confirms this result. By comparing the data in **Tables 1** and **2**, it was apparent that the *PSNR* of the proposed scheme was lower when the block size is 8×8. This was due to the fact that larger block sizes worsen the scale of distortion of the recovered pixels. However, the proposed scheme still obtained higher *PSNR* values than the other schemes, proving that our algorithm provided a more approximate common bitmap.

Table 1. Comparison of PSNR (dB) among different methods with block size 4×4

Images	GSBTC [20]	GA-BTC [21]	CSO-BTC [22]	BACO-BTC
Airplane	32.34	32.32	32.45	32.47
Fruits	32.46	32.49	32.69	32.72
Scenes	28.45	28.47	28.50	28.52
Peppers	31.25	31.32	31.77	31.81
Barbara	29.66	29.64	29.68	29.71
Boat	31.35	31.38	31.44	31.46
Average	30.92	30.94	31.09	31.11

Table 2. Comparison of PSNR (dB) among different methods with block size 8×8

Images	GSBTC [20]	GA-BTC [21]	CSO-BTC [22]	BACO-BTC
Airplane	29.27	29.28	29.49	29.51
Fruits	29.80	29.83	29.97	30.00
Scenes	25.99	25.97	26.02	26.04
Peppers	28.54	28.58	28.64	28.65
Barbara	27.05	27.07	27.09	27.12
Boat	27.85	27.91	28.16	28.18
Average	28.08	28.11	28.23	28.25

The structural similarity (SSIM) index is used to measuring the similarity between two images. The SSIM index is a decimal ranging from -1 to 1. Only when two images are identical, a value of 1 is reachable. It was defined as follows:

$$SSIM(A,B)=\frac{(2\mu_A\mu_B+c_1)(2cov+c_2)}{(\mu_A^2+\mu_B^2+c_1)(\sigma_A^2+\sigma_B^2+c_2)} \tag{16}$$

where A is the original image; B is the reconstructed image; μ_A and μ_B are the average of the two images, respectively; cov is the covariance of B; σ_A and σ_B are the variance of A and B, respectively; c_1 and c_2 are two variables to stabilize the division with weak denominator and are set by default.

Tables 3 and 4 show the comparison of SSIM among different methods, with block sizes of 4×4 and 8×8, respectively. From Tables 3 and 4, it is obvious that the average SSIM values of our scheme are higher than other schemes, which confirms that our scheme is better.

Table 3. Comparison of SSIM among different methods with block size 4×4

Images	GSBTC [20]	GA-BTC [21]	CSO-BTC [22]	BACO-BTC
Airplane	0.981	0.975	0.983	0.983
Fruits	0.984	0.982	0.985	0.984
Scenes	0.970	0.973	0.972	0.975
Peppers	0.983	0.983	0.985	0.986
Barbara	0.972	0.974	0.967	0.973
Boat	0.981	0.980	0.983	0.982
Average	0.979	0.978	0.979	0.980

Table 4. Comparison of SSIM among different methods with block size 8×8

Images	GSBTC [20]	GA-BTC [21]	CSO-BTC [22]	BACO-BTC
Airplane	0.964	0.964	0.962	0.965
Fruits	0.958	0.957	0.958	0.961
Scenes	0.944	0.945	0.952	0.953
Peppers	0.967	0.964	0.962	0.967
Barbara	0.936	0.938	0.936	0.939
Boat	0.955	0.953	0.964	0.965
Average	0.954	0.954	0.956	0.958

Time consumption is another factor that can be used to evaluate the performance of a scheme. In GSBTC [20], the bits in a common bitmap are divided into two groups, i.e., determined elements and non-determined elements. When an element from the non-determined group is selected and moved into the determined group, the mean value changes. As a result, two new *MSE* values must be computed for each element in order to determine which value should be selected. However, in our algorithm, the predetermined quantization values can be used to compute *MSE* so that we just need compute a new *MSE* only once for each element, making our algorithm less complex than GSBTC. In GA-BTC [21] and CSO-BTC [22], the higher *PSNR* values of the recovered image are achieved than that in GSBTC, but the complexity of their algorithms are much higher. **Table 5** shows the time required for different schemes with a 4×4 block. These experimental results indicated that our scheme took the least amount of time while its *PSNR* values were the highest among the compared schemes. Therefore, the proposed scheme is more efficient than the other related schemes.

Table 5. Time consumption (second) of different schemes (block size 4×4)

Images	GSBTC [20]	GA-BTC [21]	CSO-BTC [22]	BACO-BTC
Airplane	14.60	334.56	172.43	12.09
Fruits	14.25	323.18	173.52	12.81
Scenes	15.93	342.23	175.92	13.37
Peppers	14.49	335.75	172.49	12.24
Barbara	12.85	322.87	172.86	11.99
Boat	14.10	332.19	174.10	12.09
Average	14.37	331.80	173.55	12.43

Combining PSNR and time consumption, Fig. 8 shows six double-column figures for different schemes with 4×4 blocks. The blue column represents PSNR values while the orange column represents the time consumption. From the below figures, it is obvious that our scheme achieves higher PSNR than other schemes. Meanwhile, the proposed scheme is efficient while keeping high PSNR values.

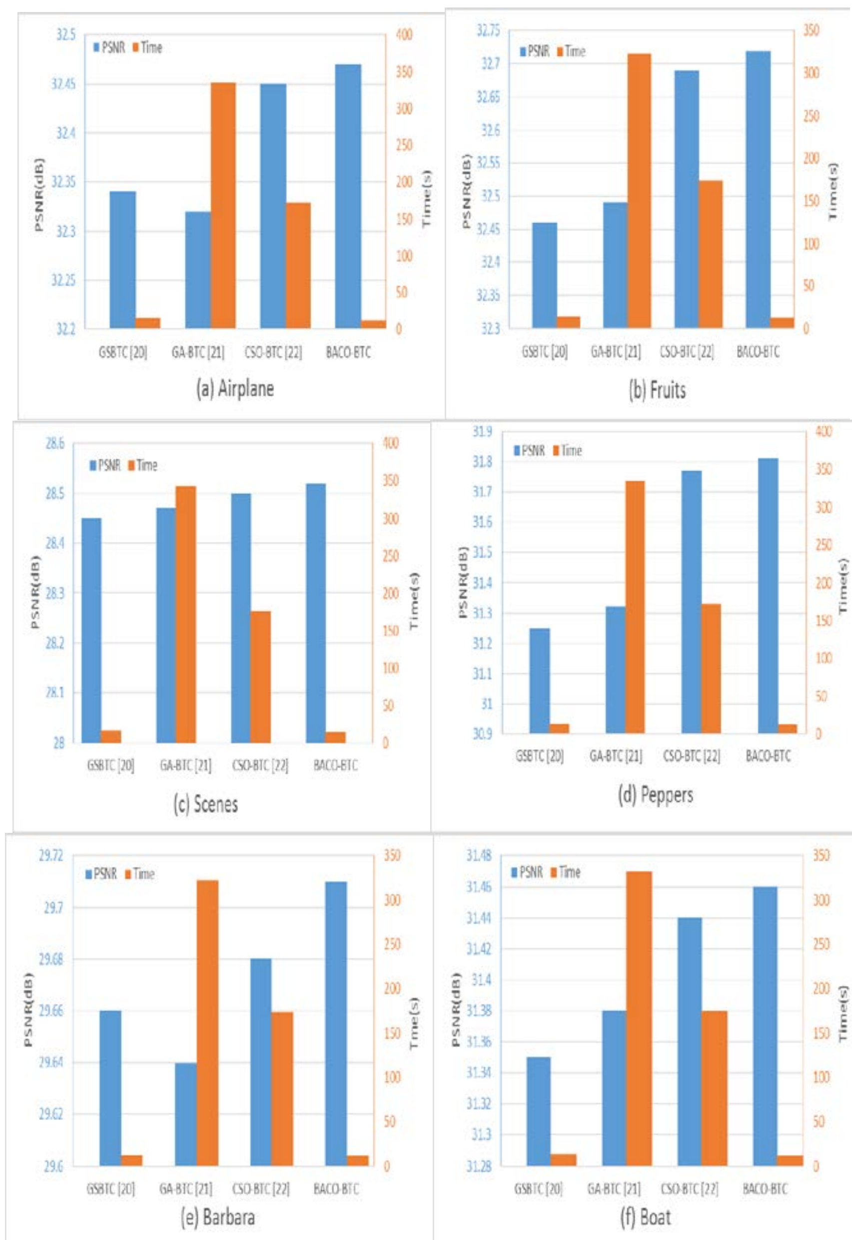


Fig. 8. Six double-column figures for different schemes with 4×4 block

5. Conclusion

A novel color image compression scheme based on BACO is proposed in this paper to compress a color image efficiently. First, the color image was compressed by traditional BTC to get three pairs of quantization values for each block. Then the common bitmap was generated to replace the original three bitmaps in order to improve the compression ratio. In our algorithm, BACO was used to generate a near- optimum common bitmap and the three pairs of quantization values were readjusted to reduce the distortion of the reconstructed image. The original image can be restored by using the common bitmap and the corresponding quantization values. The experimental results confirmed that the reconstructed images had better visual quality and consumed less time, which indicated that our scheme is efficient for obtaining an appropriate common bitmap and will be beneficial for the transmission and storage of color images. In the future, our method can be used in sharing color images and in other fields due to its efficiency and high visual quality.

References

[1] R.P. Jasmi, B. Perumal and M.P. Rajasekaran, "Comparison of image compression techniques using Huffman coding, DWT and fractal algorithm," in *Proc. of 2015 Int. Conf. on Computer Communication and Informatics*, pp. 8-10, 2015. Article (CrossRef Link).

[2] R. Bose and S. Pathak, "A novel compression and encryption scheme using variable model arithmetic coding and coupled chaotic system," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 53, no. 4, pp. 848-857, 2006. Article (CrossRef Link).

[3] B. Li and Q. Meng, "An improved SPIHT wavelet transform in the underwater acoustic image compression," in *Proc. of Int. Conf. on Measurement, Information and Control (ICMIC)*, pp. 1315-1318, 2013. Article (CrossRef Link).

[4] C. Qin, C.C. Chang, G. Horng, Y.H. Huang and Y.C. Chen, "Reversible data embedding for vector quantization compressed images using search-order coding and index parity matching," *Security and Communication Networks*, vol. 8, no. 6, pp. 899-906, 2015. Article (CrossRef Link).

[5] F. Daraee and S. Mozaffari, "Watermarking in binary document images using fractal codes," *Pattern Recognition Letters*, vol. 35, pp. 120-129, 2014. Article (CrossRef Link).

[6] P. Slusarczyk and R. Baran, "Piecewise-linear subband coding scheme for fast image decomposition," *Multimedia Tools and Applications*, pp. 1-18, 2014. Article (CrossRef Link).

[7] Y. Yang, Q. Chen and Y. Wan, "A fast near-optimum block truncation coding method using a truncated K-means algorithm and inter-block correlation," *International Journal of Electronics and Communications (AEU)*, vol. 65, no. 6, pp. 576-581, 2011. Article (CrossRef Link).

[8] E. J. Delp, and O. R. Mitchell, "Image compression using block truncation coding," *IEEE Transactions on Communications*, vol. 27, no. 9, pp. 1335-1342, 1979. Article (CrossRef Link).

[9] I.C. Chang, Y.C. Hu, W.L. Chen and C.C. Lo, "High capacity reversible data hiding scheme based on residual histogram shifting for block truncation coding," *Signal Processing*, vol. 108, pp. 376-388, 2015. Article (CrossRef Link).

[10] J.M. Guo, H. Prasetyo and J.H. Chen, "Content-based image retrieval using features extracted from halftoning-based block truncation coding," *IEEE Transactions on Image Processing*, vol. 24, no. 3, pp. 1010-1024, 2015. Article (CrossRef Link).

[11] D. Ou, L. Ye and W. Sun, “User-friendly secret image sharing scheme with verification ability based on block truncation coding and error diffusion,” *Signal Processing*, vol. 29, pp. 46-60, 2015. Article (CrossRef Link).

[12] M.D. Lema and O.R. Mitchell, “Absolute moment block truncation coding and applications to color images,” *IEEE Transactions on Communications*, vol. 32, no. 10, pp. 1148-1157, 1984. Article (CrossRef Link).

[13] L. Hui, “An adaptive block truncation coding algorithm for image compression,” in *Proc. of IEEE Int. Acoustics, Speech, and Signal Processing*, pp. 2233-2236, 1990. Article (CrossRef Link).

[14] Y.C. Hu, “Predictive moment preserving block truncation coding for graylevel image compression,” *Journal of Electronic Imaging*, vol. 13, no. 4, pp. 871-877, 2004. Article (CrossRef Link).

[15] B.C. Dhara and B.Chanda, “Block truncation coding using pattern fitting,” *Pattern Recognition*, vol. 37, no. 11, pp. 2131-2139, 2004. Article (CrossRef Link).

[16] N. Efrati, H. Licztin and H.B. Mitchell, “Classified block truncation coding-vector quantization: an edge sensitive image compression algorithm,” *Signal Processing: Image Communication*, vol. 3, pp. 275-283, 1991. Article (CrossRef Link).

[17] I. Mor, Y. Swissa and H.B. Mitchell, “A fast nearly optimum equispaced 3-level block truncation coding,” *Signal Processing: Image Communication*, vol. 6, no. 5, pp. 397-404, 1994. Article (CrossRef Link).

[18] Y. Wu and D.C. Coll, “Single bit-map block truncation coding of color images,” *IEEE Journal on Selected Areas in Communications*, vol. 10, no. 5, pp. 952-959, 1992. Article (CrossRef Link).

[19] C.K. Yang, J.C. Lin and W.H. Tsai, “Color image compression by moment-preserving and block truncation coding techniques,” *IEEE Transactions on Communications*, vol. 45, pp. 1513-1516, 1997. Article (CrossRef Link).

[20] C.C. Chang and M.N. Wu, “An algorithm for color image compression based on common bit map block truncation coding,” in *Proc. of Joint Conf. Information Sciences*, pp. 964-967, 2002. Article (CrossRef Link).

[21] C.C. Chang, C.Y. Lin and Y. H. Fan, “Lossless data hiding for color images based on block truncation coding,” *Pattern Recognition*, vol. 41, no. 7, pp. 2347-2357, 2008. Article (CrossRef Link).

[22] S.Y. Cui, Z.H. Wang, P.W. Tsai, C.C. Chang and S. Yue, “Single bitmap block truncation coding of color images using cat swarm optimization,” *Recent advances in information hiding and applications, Springer Berlin Heidelberg*, pp. 119-138, 2013. Article (CrossRef Link).

[23] C.C. Chang, C.C. Lin, C.H. Lin and Y.H. Chen, “A novel secret image sharing scheme in color images using small shadow images,” *Information Sciences*, vol. 178, no. 11, pp. 2433-2447, 2008. Article (CrossRef Link).

[24] M. Dorigo and G.D. Caro, “Ant colony optimization: a new meta-heuristic,” in *Proc. of Congress on Evolutionary Computation, Washington, DC, USA: IEEE Press*, pp. 1470-1477, 1999. Article (CrossRef Link).

[25] M. Dorigo and L.M. Gambardella, “Ant colony system: a cooperative learning approach to the traveling salesman problem,” *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53-66, 1997. Article (CrossRef Link).

[26] M. Kong and P. Tian, “A binary ant colony optimization for the unconstrained function optimization problem,” in *Proc. of Int. Conf. on Computational Intelligence and Security, Part I, LNAI 3801, Springer*, pp. 682-687, 2005. Article (CrossRef Link).

[27] W. Xiong, L. Wang and C. Yan, "Binary ant colony evolutionary algorithm," in *Proc. of Int. Conf. on Intelligent Computing*, pp. 10-20, 2005. Article (CrossRef Link).

Zhihong Li is currently an associate professor in Taiyuan University of Science and Technology, China. He received the B.Eng. in electronic information engineering from Taiyuan University of Science and Technology in 1994. His research interests include compressed sensing, and secret image sharing. He was participated in the projects on distributed video coding and now is leading one research project on image secret from Shanxi Natural Science Foundation.

Qiang Jin was born in Shanxi Province, China, in 1989. He is currently pursuing the M.E. degree in Taiyuan University of Science and Technology. His current research interests include data hiding, and image compression.

Chin-Chen Chang received the B.S. degree in applied mathematics and the M.S. degree in computer and decision sciences from National Tsing Hua University, Hsinchu, Taiwan, R.O.C., in 1977 and 1979, respectively. He received his Ph. D in computer engineering in 1982 from the National Chiao Tung University, Hsinchu, Taiwan. Since February 2005, he has been a Chair Professor of Feng Chia University. In addition, he has served as a consultant to several research institutes and government departments. His current research interests include database design, computer cryptography, image compression and data structures.

Li Liu received her B.E. degree in communication engineering in 2002, from Lanzhou Railway University and M.E. degree in communication and information system in 2006, from Lanzhou Jiaotong University. Her current research interests include image compression and secret sharing.

Anhong Wang received B.E and M.E. degrees from Taiyuan University of Science and Technology (TYUST), and Ph. D degree in Institute of Information Science, Beijing Jiaotong University in 2009. She is now the director of Institute of Digital Media and Communication, Taiyuan University of Science and Technology. Her research interest includes image/video coding, compressed sensing, and secret image sharing. Now she is leading two national research projects from National Science Foundation of China.