HIGH TECHNOLOGY LETTERS | Vol. 25 No. 4 | Dec. 2019 | pp. 355-363

doi:10.3772/j. issn. 1006-6748. 2019. 04. 002

Single bitmap block truncation coding of color images using hill climbing algorithm①

Zhang Lige(张力戈)②, Qin Xiaolin, Li Qing, Peng Haoyue, Hou Yu
(Chengdu Institute of Computer Applications, Chinese Academy of Sciences, Chengdu 610041, P. R. China)
(School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100049, P. R. China)

Abstract

In order to generate an efficient common bitmap in single bitmap block truncation coding (SBBTC) of color images, an improved SBBTC scheme based on weighted plane (W-plane) method and hill climbing algorithm is proposed. Firstly, the incoming color image is partitioned into non-overlapping blocks and each block is encoded using the W-plane method to get an initial common bitmap and quantization values. Then, the hill climbing algorithm is applied to optimize an initial common bitmap and generate a near-optimized common bitmap. Finally, the quantization values are recalculated by the near-optimized common bitmap and the considered color image is reconstructed block by block through the common bitmap and the new quantization values. Since the processing of each image block in SBBTC is independent and identical, parallel computing is applied to reduce the time consumption of this scheme. The simulation results show that the proposed scheme has better visual quality and time consumption than those of the reference SBBTC schemes.

Key words: block truncation coding (BTC), common bitmap, parallel computing, hill climbing algorithm

0 Introduction

With the rapid development of the Internet, the number of digital images used by the public has risen sharply. As a result, a large amount of storage space is required for digital images and the transmission of these images through limited bandwidth channel is time consuming. Therefore, image compression techniques have become a kind of necessary ways to resolve these issues. Image compression refers to the process of representing an image in a compact way by reducing various redundancies in the image and it is generally classified into 2 categories: lossless compression and lossy compression[1]. Lossless compression techniques provide excellent visual quality of reconstructed images since there is no distortion introduced in their processes, but the compression ratio of them is low. So far, many algorithms used in lossless compression techniques have been proposed, such as run length encoding[2], Huffman coding[3], chain codes[4]. These techniques are generally used in some special fields such as medical images and military images. Lossy compression techniques provide high compression ratio and good visual quality of restored images. There are quite a few lossy compression techniques, such as vector quantization[5], fractal image compression[6], wavelet transform coding[7], block truncation coding[8]. These techniques are usually used to compress general-purpose digital images. Since most of the images used by the public are general-purpose digital images, the lossy compression has become one of the hot topics. As one of the lossy compression techniques, block truncation coding is famous for its simplicity which makes it appropriate to be applied in the compression of general-purpose digital images.

Block truncation coding (BTC) was proposed by Mitchel and Delp[8] for grayscale image compression. Due to the advantages such as low computational and fault tolerance, BTC has also been used in many fields, such as image retrieval[9], data hiding[10], image authentication[11]. In BTC, the grayscale image is divided into non-overlapping blocks and each block is encoded with a representative set of quantization values. Many schemes were proposed to improve the bit rate and visual quality of BTC. For instance, Mitchell

① Supported by the National Natural Science Foundation of China (No. 61402537), the Open Fund of Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis (No. HCIC201706) and the Sichuan Science and Technology Programme (No. 2018GZDZX0041).

② To whom correspondence should be addressed. E-mail: zhanglige15@mails.ucas.ac.cn

Received on Oct. 15, 2018

et al.[12] proposed the absolute moment block truncation coding (AMBTC), Hui[13] proposed the adaptive BTC, Haung et al.[14] proposed the hybrid BTC. These 3 typical schemes all have better visual quality than BTC dose. Later, BTC was extended to the compression of color images.

Color images are typically composed of 3 channels, i. e., R, G and B. Apparently, color images contain a lot of data which are triple the data in grayscale images. Conventionally, BTC is used to code 3 channels separately. However, the scheme has a low compression ratio. Therefore, it is important to utilize the correlation among 3 channels to reduce the bit rate. To achieve the goal, some schemes have been proposed. Wu and Coll[15] proposed the single bitmap BTC (SBBTC) scheme, which preserved the spatial details of the color image by using a common bitmap for 3 channels. In the scheme, the weighted plane (W-plane) method which has low complexity and good performance regarding the mean square error (MSE) was proposed. Kurita and Otsu[16] used an adaptive one-bit vector quantization algorithm to generate the common bitmap. Tai et al.[17] proposed a new SBBTC scheme. The scheme used the Hopfield neural network to generate a common bitmap. Later, Tai et al.[18] proposed another new scheme using the genetic algorithm and the absolute moment BTC (GA-AMBTC). The scheme had a better performance than the earlier scheme did. Chang and Wu[19] employed the gradual search algorithm with BTC to generate the common bitmap (GS-BTC). The scheme had low complexity and acceptable visual quality. Cui et al.[20] described a new scheme which incorporated the cat swarm optimization algorithm for generating the common bitmap (CSO-BTC). Recently, Li et al.[21] employed the binary ant colony optimization with BTC to generate the common bitmap (BACO-BTC). Both GA-AMBTC and CSO-BTC are combined with the intelligence optimization algorithms, and they can be used to find the global optimal common bitmap after multiple iterations of the intelligence optimization algorithms they employed. However, the time consumption for them to find the global optimal common bitmap is enormous. GSBTC and BACO-BTC have low computation complexity and can be employed to generate effective common bitmaps which have acceptable visual quality.

An improved scheme based on W-plane method is proposed. According to the characteristics of BTC, a hill climbing algorithm is employed to generate a near-optimal common bitmap and parallel computing is used to reduce the time consumption. The simulation results show that the proposed scheme has lower time consumption and better visual quality than the reference schemes.

The remainder of this paper is organized as follows. Some related work is presented in Section 1. Section 2 describes the proposed scheme in detail. Simulation results are summarized in Section 3, and the conclusions are given in Section 4.

1 Preliminaries

1.1 Absolute moment block truncation coding

AMBTC is computationally simpler than BTC while preserving the same compression rate as BTC. The basic steps of AMBTC are as follows.

Step 1 Partition a grayscale image into nonoverlapping $m \times n$ blocks of which mean intensity can be computed by

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

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

Step 2 Generate bitmap B and 2 quantization values x_H and x_L. Each pixel block is classified individually into 2 levels. For each pixel x_{ij} in one block, if $x_{ij} < \bar{x}$, it is assigned to the level of low and the value of which is set to 0 at position (i, j) in the bitmap; otherwise, the pixel is assigned to the level of high and the value of which is set to 1 at position (i, j) in the bitmap. The two quantization values can be computed by

$$x_H = \frac{1}{q}\sum_{x_{ij} \geqslant \bar{x}} x_{ij} \tag{2}$$

$$x_L = \frac{1}{m \times n - q}\sum_{x_{ij} < \bar{x}} x_{ij} \tag{3}$$

where q is the number of pixels with a value higher than $\bar{x}$. The bitmap is generated by

$$b_{ij} = \begin{cases} 1 & x_{ij} \geqslant \bar{x} \\ 0 & x_{ij} < \bar{x} \end{cases} \tag{4}$$

Step 3 Reconstruct each block of grayscale image M as follows:

$$m_{ij} = \begin{cases} x_H & b_{ij} = 1 \\ x_L & b_{ij} = 0 \end{cases} \tag{5}$$

1.2 W-plane method

Unlike traditional BTC of color images, SBBTC is used to compress the color image as one bitmap. W-plane method is one of the traditional SBBTC schemes.

In the W-plane method, the original color image is also partitioned into nonoverlapping $m \times n$ blocks. Let R_{ij}, G_{ij} and B_{ij} denote the pixel value of the R, G and B channels respectively in each block, $x_{ij} = (R_{ij},$

G_{ij}, B_{ij}) denotes the pixel vector. The weighted plane of each block is constructed by

$$w_{ij} = \frac{R_{ij} + G_{ij} + B_{ij}}{3} \tag{6}$$

and the mean intensity of each weighted plane can be computed by

$$\bar{w} = \frac{1}{m \times n} \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} \tag{7}$$

Then, the common bitmap of each block can be constructed according to w_{ij} and $\bar{w}$. Each bit of the common bitmap is determined by

$$b_{ij} = \begin{cases} 1 & w_{ij} \geqslant \bar{w} \\ 0 & w_{ij} < \bar{w} \end{cases} \tag{8}$$

According to the common bitmap, the quantization vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ can be computed:

$$\boldsymbol{x}_1 = \frac{1}{q} \sum_{b_{ij}=1} \boldsymbol{x}_{ij} \tag{9}$$

$$\boldsymbol{x}_2 = \frac{1}{m \times n - q} \sum_{b_{ij}=0} \boldsymbol{x}_{ij} \tag{10}$$

where q is the number of b_{ij} which equals 1.

Finally, each block of color image C can be reconstructed as follows:

$$\boldsymbol{c}_{ij} = \begin{cases} \boldsymbol{x}_1 & b_{ij} = 1 \\ \boldsymbol{x}_2 & b_{ij} = 0 \end{cases} \tag{11}$$

1.3 Hill climbing algorithm

Hill climbing algorithm attempts to compute a local optimal solution of the target function $f(x)$. The process of hill climbing algorithm can be summarized as follows:

Step 1 Select an initial point x_0 and record the optimal solution $x_{\text{best}} = x_0$. The set of the neighbourhood points for x_0 is denoted as $T = N(x_0)$.

Step 2 Stop the algorithm and output x_{best} if $T = \varnothing$ ($\varnothing$ denotes the empty set) or other stopping criterions are reached. Otherwise, go to Step 3.

Step 3 Select one point x_n from T and calculate $f(x_n)$. Set $x_{\text{best}} = x_n$, $T = T - x_n$ and go to Step 2 if $f(x_n)$ is better than $f(x_{\text{best}})$. Otherwise, keep the value of x_{best}, set $T = T - x_n$ and go to Step 2.

2 The proposed scheme

An improved method based on W-plane method using parallel computing and hill climbing algorithm is proposed. The proposed scheme consists of 5 phases as Fig. 1 shows.

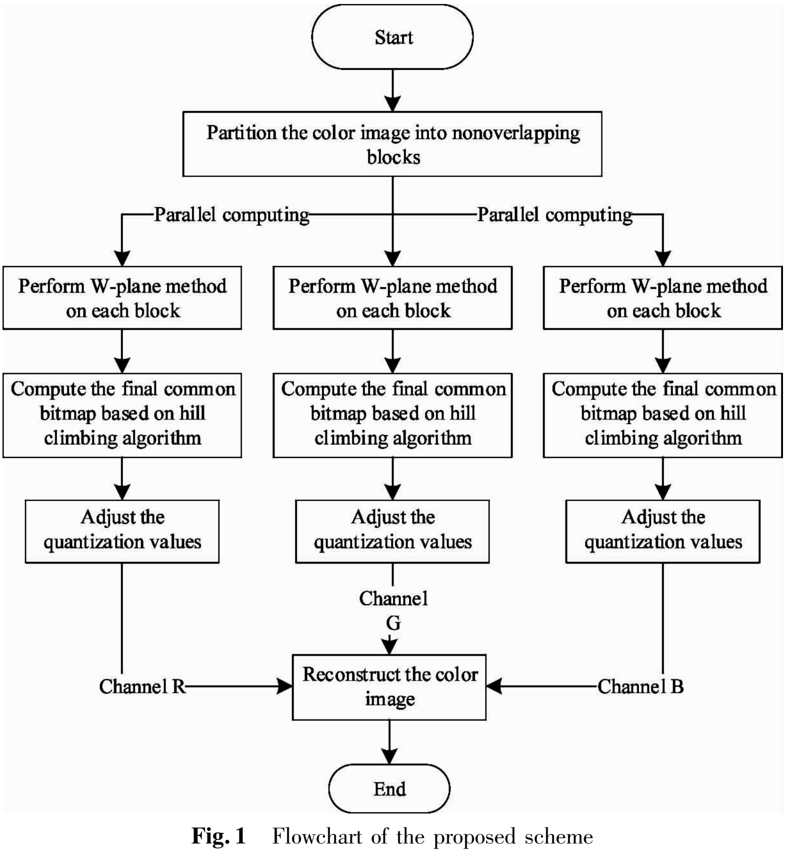


Fig. 1 Flowchart of the proposed scheme

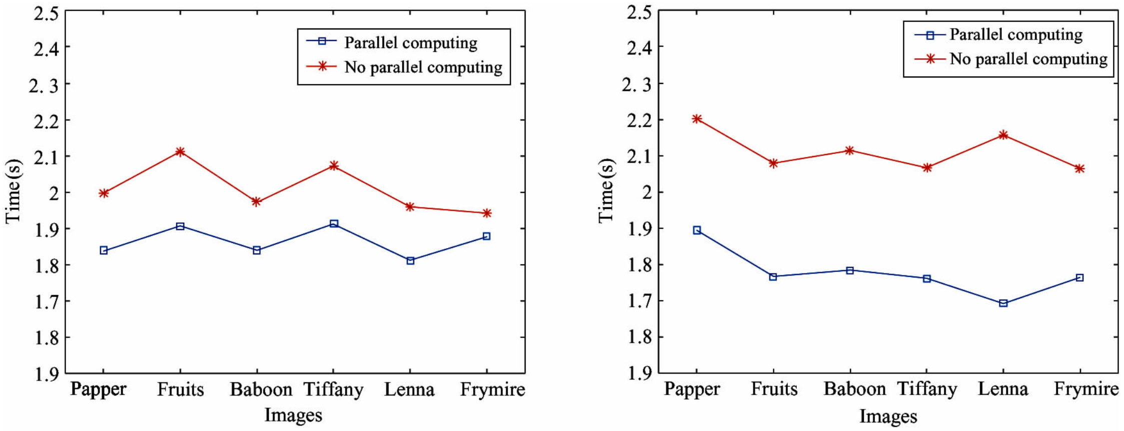
2.1 Partition the color image into nonoverlapping blocks

In the proposed scheme, the original color image should be divided into a series of nonoverlapping blocks at first and the number of the nonoverlapping blocks is recorded as B_{num}. Since the subsequent operations for each block are identical and there is no interaction between these operations, these blocks can be processed independently. Therefore, the proposed scheme uses parallel computing for the operations of these blocks to fully utilize the CPU resources and reduce the time consumption.

Partition the blocks evenly into several parts according to the number of the cores on the machine so that the parallel computing can be used. Assuming the machine has C_{num} cores, then the number of blocks allocated to each core can be calculated:

$$CB_{num} = \frac{B_{num}}{C_{num}} \tag{12}$$

To confirm the effectiveness of the parallel computing, the time consumption of the parallel computing is compared with that of no parallel computing for the proposed scheme. Fig. 2(a) shows the result of comparison with 4×4 block size and Fig. 2(b) shows the result of comparison with 8×8 block size. It is obvious that the proposed scheme using the parallel computing has less time consumption.



(a) with 4×4 block size　　(b) with 8×8 block size

Fig. 2 Comparison for the time consumption of the parallel computing with that of no parallel computing

2.2 Performing W-plane method on each block

For each block, W-plane method is used to generate the initial common bitmap $B_{initial}$. The quantization values x_{RH}, x_{RL}, x_{GH}, x_{GL}, x_{BH} and x_{BL} for the R, G and B channels are calculated according to $B_{initial}$ respectively, as shown in Eq. (13).

$$\begin{aligned} x_{RH} &= \frac{1}{q}\sum_{b_{ij}=1} R_{ij} \\ x_{RL} &= \frac{1}{m \times n - q}\sum_{b_{ij}=0} R_{ij} \\ x_{GH} &= \frac{1}{q}\sum_{b_{ij}=1} G_{ij} \\ x_{GL} &= \frac{1}{m \times n - q}\sum_{b_{ij}=0} G_{ij} \\ x_{BH} &= \frac{1}{q}\sum_{b_{ij}=1} B_{ij} \\ x_{BL} &= \frac{1}{m \times n - q}\sum_{b_{ij}=0} B_{ij} \end{aligned} \tag{13}$$

where R_{ij}, G_{ij} and B_{ij} are the pixel values at position (i, j) of the R, G and B channels; b_{ij} is the value at position (i, j) in the initial common bitmap; q refers to the number of b_{ij} that equals to 1; m and n stand for the size of each block.

2.3 Computing the final common bitmap based on hill climbing algorithm

The initial bitmap which is generated by W-plane method always results in poor visual quality. To overcome the problem, the proposed scheme uses hill climbing to generate the final bitmap.

Considering the enormous time consumption when optimizing the entire bitmap at once using intelligent optimization algorithms and the independent effect when changing different bit of bitmap independently, the proposed scheme optimizes each bit of the bitmap separately. The detailed steps are as follows.

Step 1 Create an empty matrix ***BF***. Calculate the initial MSE of M using the initial bitmap $\boldsymbol{B}_{initial}$ by Eq. (14).

$$MSE = \sum_{b_{ij}=0} (\boldsymbol{x}_{ij} - \boldsymbol{x}_L)^2 + \sum_{b_{ij}=0} (\boldsymbol{x}_{ij} - \boldsymbol{x}_H)^2 \quad (14)$$

where $x_{ij} = (R_{ij}, G_{ij}, B_{ij})$ stands for the pixel vector in the block, $\boldsymbol{x}_H = (x_{RH}, x_{GH}, x_{BH})$ and $\boldsymbol{x}_L = (x_{RL}, x_{GL}, x_{BL})$ are 2 vectors consisting of 6 quantization values.

Step 2 Select one element b_{mn} at position (m, n) in $\boldsymbol{B}_{initial}$ and initialize the optimal solution b_{best} and target function $f(b_{mn})$ with b_{mn} and M respectively.

Step 3 Find the neighbourhood points of b_{mn}. Since the range of each element in $\boldsymbol{B}_{initial}$ is $\{0, 1\}$, there is only one neighbourhood point of b_{mn}. Hence, neighbourhood point t of b_{mn} is calculated as follows:

$$t = 1 - b_{mn} \quad (15)$$

Step 4 Construct a temporary bitmap $\boldsymbol{BT}$ by Eq. (16) and calculate $f(t)$ by Eq. (17). Note that there is only one difference between $\boldsymbol{BT}$ and $\boldsymbol{B}_{initial}$, which is located at position (m, n).

$$bt_{ij} = \begin{cases} b_{ij} & (i, j) \neq (m, n) \\ t & (i, j) = (m, n) \end{cases} \quad (16)$$

$$f(t) = \sum_{bt_{ij}=1} (\boldsymbol{x}_{ij} - \boldsymbol{x}_H)^2 + \sum_{bt_{ij}=0} (\boldsymbol{x}_{ij} - \boldsymbol{x}_L)^2 \quad (17)$$

Step 5 Update b_{best} as follows:

$$b_{best} = \begin{cases} t & f(b_{mn}) > f(t) \\ b_{mn} & f(b_{mn}) \leqslant f(t) \end{cases} \quad (18)$$

Step 6 Update the final bitmap $\boldsymbol{BF}$ according to the position of b_{mn} as follows:

$$bf_{mn} = b_{best} \quad (19)$$

Step 7 Repeat Step 2 to Step 6 until all the elements in $\boldsymbol{B}_{initial}$ have been selected.

When all the elements in $\boldsymbol{B}_{initial}$ have been selected, all elements in BF are assigned. Up to this point, the final bitmap is obtained.

The main procedure of this part can be summarized as Algorithm 1.

Algorithm 1 Computation for the final common bitmap based on hill climbing algorithm

Input: Block C_{block} of the original color image, 6 quantization values x_{RH}, x_{RH}, x_{GH}, x_{GL}, x_{BH}, x_{BL} the initial common bitmap $B_{initial}$ of C_{block}.

Output: Final common bitmap $\boldsymbol{BF}$ of C_{block}.

Set M as the initial MSE;
Set $\boldsymbol{BF}$ as an empty matrix;
Set $\boldsymbol{BT}$ as a temporary bitmap;
for i: $=1$ to m do
  for j: $=1$ to n do
    $b_{best} \leftarrow b_{ij}$;
    $f(b_{ij}) \leftarrow M$;
    $t \leftarrow I - b_{ij}$;
    $bt_{ij} \leftarrow i$;
    $f(i) \leftarrow new$. MSE computed according to BT;
    if $f(b_{ij}) \leqslant f(i)$ then
      $bf_{ij} \leftarrow b_{best}$;
    else
      $b_{best} \leftarrow t$;
      $bf_{ij} \leftarrow b_{best}$;
    end if
  end for
end for

There is an example in Fig. 3. The selected element of initial common bitmap is b_{42} and the target function $f(b_{42})$ is 1.2. First, find neighbourhood point t of b_{42} and construct the temporary bitmap. Second, the result of target function $f(t)$ is 1.3 according to the temporary bitmap. Then, the optimal solution b_{best} is updated to b_{42} since $f(b_{42})$ is smaller than $f(t)$. In the end, the final common bitmap is updated by setting bf_{42} to b_{best}.

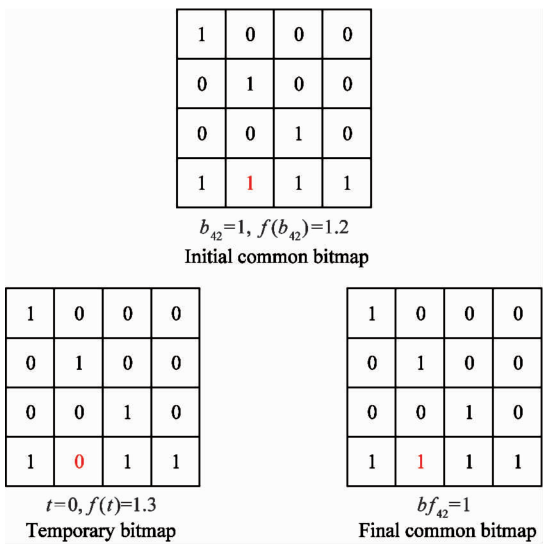


Fig. 3 An example for Algorithm 1

2.4 Recalculating the quantization values

The quantization values are one of the main components in BTC. To reconstruct the color image, the quantization values need to be recalculated. According to the final common bitmap $\boldsymbol{BF}$ which is generated in subsection 2.3, the quantization values can be recalculated by Eq. (20).

$$
\begin{aligned}
c_{RH} &= \frac{1}{q_c}\sum_{bf_{ij}=1} R_{ij} \\
c_{RL} &= \frac{1}{m \times n - q_c}\sum_{bf_{ij}=0} R_{ij} \\
c_{GH} &= \frac{1}{q_c}\sum_{bf_{ij}=1} G_{ij} \\
c_{GL} &= \frac{1}{m \times n - q_c}\sum_{bf_{ij}=0} G_{ij} \\
c_{BH} &= \frac{1}{q_c}\sum_{bf_{ij}=1} B_{ij} \\
c_{BL} &= \frac{1}{m \times n - q_c}\sum_{bf_{ij}=0} B_{ij}
\end{aligned} \tag{20}
$$

where c_{RH}, c_{RL}, c_{GH}, c_{GL}, c_{BH} and c_{BL} denote 6 quantization values; bf_{ij} stands for the value at position (i, j) in the final common bitmap $\boldsymbol{BF}$; and q_c is the number of bf_{ij} which equals to 1.

2.5 Reconstructing the color image

After getting the final common bitmap and the quantization values, the color image can be reconstructed as follows:

$$
\begin{aligned}
R_{ij}^r &= \begin{cases} c_{RH} & bf_{ij} = 1 \\ c_{RL} & bf_{ij} = 0 \end{cases} \\
G_{ij}^r &= \begin{cases} c_{GH} & bf_{ij} = 1 \\ c_{GL} & bf_{ij} = 0 \end{cases} \\
B_{ij}^r &= \begin{cases} c_{BH} & bf_{ij} = 1 \\ c_{BL} & bf_{ij} = 0 \end{cases}
\end{aligned} \tag{21}
$$

An example for this section is shown in Fig. 4. Assume that the quantization values c_{RH}, c_{RL}, c_{GH}, c_{GL}, c_{BH} and c_{BL} are equal to 235, 226, 182, 156, 255 and 250. According to the final common bitmap in Fig. 4, the 3 channels can be reconstructed. Channel R is reconstructed by c_{RH}, c_{RL}. Channel G is reconstructed by c_{GH}, c_{GL}. Channel B is reconstructed by c_{BH}, c_{BL}.

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

Final common bitmap

235	226	226	226
226	235	226	226
226	226	235	226
235	235	235	235

Channel R

182	156	156	156
156	182	156	156
156	156	182	156
182	182	182	182

Channel B

255	250	250	250
250	255	250	250
250	250	255	250
255	255	255	255

Channel G

Fig. 4 An example for the reconstruction of the color image

3 Simulation results

The proposed scheme is simulated by Matlab 2016a and the simulation environment is a PC with 8 GB RAM and Inter Core i3 3.4 GHz CPU. The color images used to evaluate the performance of the proposed scheme are shown in Fig. 5.

Fig. 5 Color test images

In Table 1 and Table 2, the proposed scheme is compared with W-plane method. MSE is used to measure the performance of 2 schemes, and it is defined as follows:

$$
MSE = \frac{1}{m \times n}\sum_{i=1}^{m}\sum_{j=1}^{n} \frac{(R_{ij}^r - R_{ij})^2 + (G_{ij}^r - G_{ij})^2 + (B_{ij}^r - B_{ij})^2}{3} \tag{22}
$$

where R_{ij}, G_{ij} and B_{ij} stand for the pixel values of the 3 channels in the original color image at position (i, j) and R_{ij}^r, G_{ij}^r, B_{ij}^r refer to the pixel values of the 3 channels in the reconstructed color image at position (i, j), m and n are the size of the color image.

Table 1 shows the comparison of MSE between the W-plane method[15] and the proposed method. The block size of the comparison is 4×4. It is seen that the proposed scheme has better MSE performance than W-plane method on each color image. Table 2 shows the comparison of MSE, with the block size of 8×8. It is obvious that the MSE performance of the proposed scheme is also better than W-plane method on each color image. These results show that the improvement in the proposed scheme based on the W-plane method is efficient.

Table 1 MSE for images reconstructed by W-plane method and the proposed scheme with block size 4 ×4

Images	W-plane-BTC[15]	Proposed scheme
Pepper	11.1420	10.4732
Fruits	19.5508	16.9042
Baboon	95.9592	90.7869
Tiffany	19.2256	12.7839
Lenna	19.9605	18.7415
Frymire	254.1143	213.5660

Table 2 MSE for images reconstructed by W-plane method and the proposed scheme with block size 8 ×8

Images	W-plane-BTC[15]	Proposed scheme
Pepper	32.8668	29.8279
Fruits	35.9711	30.6846
Baboon	145.6842	135.5581
Tiffany	31.7653	22.9966
Lenna	38.6159	35.6669
Frymire	468.8183	394.8130

To further verify the validity of the proposed scheme, the proposed scheme is compared with other 3 related schemes, GA-AMBTC[18], GSBTC[19], BACO-BTC[21]. The parameters of GA-AMBTC are set according to Ref. [18].

Table 3 and Table 4 show the comparison of MSE among 4 different schemes. The block size of the color images used in Table 3 and Table 4 is 4 ×4 and 8 ×8, respectively. From 2 tables, it is apparent that the MSE of the proposed scheme is lower than that of the other 3 related schemes, which means that the proposed scheme is better.

Table 5 and Table 6 show the comparison of structural similarity (SSIM) index among 4 different schemes with block sizes of 4 ×4 and 8 ×8, respectively. SSIM is a method for measuring the similarity between 2 images. It can be calculated by Eq. (23).

Table 3 MSE for images reconstructed by 4 schemes with block size 4 ×4

Image	GA-AMBTC[18]	GSBTC[19]	BACO-BTC[21]	Proposed scheme
Pepper	10.5770	13.2820	10.5253	10.4732
Fruits	18.1500	19.1409	16.9901	16.9042
Baboon	94.6031	98.5005	90.7963	90.7869
Tiffany	12.9141	24.1907	14.9055	12.7839
Lenna	19.0081	20.1871	18.7826	18.7415
Frymire	223.9071	336.0331	226.0877	213.5660

Table 4 MSE for images reconstructed by 4 schemes with block size 8 ×8

Image	GA-AMBTC[18]	GSBTC[19]	BACO-BTC[21]	Proposed scheme
Pepper	30.9609	35.4107	29.9243	29.8279
Fruits	33.3320	35.3892	31.2399	30.6846
Baboon	146.9367	146.2950	136.1672	135.5581
Tiffany	24.1108	35.0836	26.0522	22.9966
Lenna	37.3787	36.5837	35.4927	35.6669
Frymire	400.9378	551.3695	414.1567	394.8130

$$SSIM(X, Y) = \frac{2(\mu_X \mu_Y + C_1)(2\sigma_{XY} + C_2)}{(\mu_X^2 + \mu_Y^2 + C_1)(\sigma_X^2 + \sigma_Y^2 + C_2)} \tag{23}$$

where X is the original color image, Y is the reconstructed color image, μ_X and μ_Y refer to the average of the two images respectively, σ_{XY} is the covariance of the two images, σ_X and σ_Y stand for the variance of X and Y respectively, C_1 and C_2 are two default constants.

In Table 5 and Table 6, the proposed scheme has higher SSIM values than the other 3 schemes. The result shows that the proposed scheme has better visual quality.

Table 7 and Table 8 show the comparison of time consumption among 4 different schemes. To reflect the differences in time consumption among 4 methods more intuitively, the size of the test images is set to 1 024 × 1 024. The block size in Table 7 and Table 8 is 4 ×4 and 8 ×8 respectively. Consistent with the analysis in the introduction, time consumption of GA-AMBTC in 2 tables is significantly higher, and the time consumption of GSBTC and BACO-BTC is relatively low. It is obvious that the proposed scheme takes the least amount of time. Therefore, the proposed scheme is more efficient.

Table 5 SSIM for images reconstructed by 4 schemes with block size 4 × 4

Image	GA-AMBTC[18]	GSBTC[19]	BACO-BTC[21]	Proposed scheme
Pepper	0.9925	0.9911	0.9925	0.9926
Fruits	0.9798	0.9790	0.9804	0.9806
Baboon	0.9015	0.9029	0.9059	0.9060
Tiffany	0.9863	0.9758	0.9838	0.9865
Lenna	0.9867	0.9855	0.9867	0.9867
Frymire	0.9445	0.9218	0.9449	0.9471

Table 6 SSIM for images reconstructed by 4 schemes with block size 8 × 8

Image	GA-AMBTC[18]	GSBTC[19]	BACO-BTC[21]	Proposed scheme
Pepper	0.9875	0.9768	0.9790	0.9791
Fruits	0.9619	0.9606	0.9636	0.9641
Baboon	0.8523	0.8598	0.8640	0.8645
Tiffany	0.9762	0.9639	0.9740	0.9769
Lenna	0.9758	0.9760	0.9767	0.9767
Frymire	0.8946	0.8633	0.8927	0.8970

Table 7 Time consumption for images reconstructed by 4 schemes with block size 4 × 4

Image	GA-AMBTC[18]	GSBTC[19]	BACO-BTC[21]	Proposed scheme
Pepper	202.8485 s	3.7854 s	2.2714 s	1.8382 s
Fruits	201.0195 s	3.6766 s	2.1161 s	1.9070 s
Baboon	198.7589 s	3.7595 s	2.0913 s	1.8402 s
Tiffany	197.0844 s	4.0321 s	2.1336 s	1.9122 s
Lenna	235.8694 s	3.9194 s	2.0971 s	1.8118 s
Frymire	206.3119 s	3.3775 s	2.0989 s	1.8770 s

Table 8 Time consumption for images reconstructed by 4 schemes with block size 8 × 8

Image	GA-AMBTC[18]	GSBTC[19]	BACO-BTC[21]	Proposed scheme
Pepper	123.0612 s	3.3418 s	2.4626 s	1.8939 s
Fruits	125.5615 s	3.0716 s	2.6581 s	1.7678 s
Baboon	120.9918 s	3.3420 s	2.4037 s	1.7847 s
Tiffany	124.1227 s	3.6489 s	2.3483 s	1.7616 s
Lenna	112.3930 s	3.3900 s	2.4105 s	1.6931 s
Frymire	104.6438 s	2.8749 s	2.6115 s	1.7639 s

4 Conclusion

An efficient scheme for the compression of color images based on W-plane method is proposed. Since there is no interaction among the operations for each block in BTC, parallel computing is used to reduce time consumption. To obtain a near optimal common bitmap, hill climbing algorithm is employed in this scheme. The simulation results show that the proposed scheme has better visual quality and time consumption than those of the reference schemes. Therefore, the proposed scheme is efficient for color image compression and beneficial to storing and transmitting color images. In the future, this scheme can be further improved and used in other fields.

References

[1] Rabbani M, Jones P W. Digital Image Compression Techniques[M]. Bellingham, Washington: SPIE, 1991: 6-7
[2] Robinson A H, Cherry C. Results of a prototype television bandwidth compression scheme[J]. *Proceedings of the IEEE*, 1967, 55(3): 356-364
[3] Huffman D A. A method for the construction of minimum-redundancy codes[J]. *Proceedings of the IRE*, 1952, 40(9): 1098-1101
[4] Sanchez-Cruz H, Rodriguez-Dagnino R M. Compressing bilevel images by means of a three-bit chain code[J]. *Optical Engineering*, 2005, 44(9): 097004
[5] Goldberg M, Boucher P, Shlien S. Image compressionus-

ing adaptive vector quantization[J]. *IEEE Transactions on Communications*, 1986, 34(2): 180-187
[6] Fisher Y. Fractal image compression[J]. *Fractals*, 1994, 2(3): 347-361
[7] Antonini M, Barlaud M, Mathieu P, et al. Image coding using wavelet transform[J]. *IEEE Transactions on Image Processing*, 1992, 1(2): 205-220
[8] Delp E, Mitchell O. Image compression using block truncation coding[J]. *IEEE transactions on Communications*, 1979, 27(9): 1335-1342
[9] Guo J M, Prasetyo H. Content-based image retrieval using features extracted from halftoning-based block truncation coding[J]. *IEEE Transactions on Image Processing*, 2014, 24(3): 1010-1024
[10] Sun W, Lu Z M, Wen Y C, et al. High performance reversible data hiding for block truncation coding compressed images[J]. *Signal, Image and Video Processing*, 2013, 7(2): 297-306
[11] Lin C C, Huang Y, Tai W L. A novel hybrid image authentication scheme based on absolute moment block truncation coding[J]. *Multimedia Tools and Applications*, 2017, 76(1): 463-488
[12] Lema M, Mitchell O. Absolute moment block truncation coding and its application to color images[J]. *IEEE Transactions on Communications*, 1984, 32(10): 1148-1157
[13] Hui L. An adaptive block truncation coding algorithm for image compression[C]//International Conference on Acoustics, Speech, and Signal Processing, Albuquerque, USA, 1990: 2233-2236
[14] Haung C S, Lin Y. Hybrid block truncation coding[J]. *IEEE Signal Processing Letters*, 1997, 4(12): 328-330
[15] Wu Y, Coll D C. Single bit-map block truncation coding of color images[J]. *IEEE Journal on Selected Areas in Communications*, 1992, 10(5): 952-959
[16] Kurita T, Otsu N. A method of block truncation coding for color image compression[J]. *IEEE Transactions on Communications*, 1993, 41(9): 1270-1274
[17] Tai S C, Lin Y C, Lin J F. Single bit-map block truncation coding of color images using a Hopfield neural network[J]. *Information Sciences*, 1997, 103(1-4): 211-228
[18] Tai S C, Chen W J, Cheng P J. Genetic algorithm for single bit map absolute moment block truncation coding of color images[J]. *Optical Engineering*, 1998, 37(9): 2483-2490
[19] Chang C C, Wu M N. An algorithm for color image compression base on common bit map block truncation coding[C]//Proceedings of the 6th Joint Conference on Information Science, North Carolina, USA, 2002: 964-967
[20] Cui S Y, Wang Z H, Tsai P W, et al. Single Bitmap Block Truncation Coding of Color Images Using Cat Swarm Optimization[M]. Berlin: Springer, 2013. 119-138
[21] Li Z, Jin Q, Chang C C, et al. A common bitmap block truncation coding for color images based on binary ant colony optimization[J]. *KSII Transactions on Internet and Information Systems (TIIS)*, 2016, 10(5): 2326-2345

Zhang Lige, born in 1995. He is currently working toward the Ph. D degree in computer science at University of Chinese Academy of Sciences. He received the B. S. degree in software engineering from Taiyuan University of Technology in 2015. His research interests include image compression, machine learning and intelligent algorithm.