


# Lossless data hiding for absolute moment block truncation coding using histogram modification

Cheonshik Kim<sup>1</sup>  · Dongkyoo Shin<sup>1</sup> · Lu Leng<sup>2,3</sup> · Ching-Nung Yang<sup>4</sup>

Received: 22 June 2016 / Accepted: 20 September 2016 / Published online: 19 October 2016  
© Springer-Verlag Berlin Heidelberg 2016

**Abstract** This paper presents a lossless data hiding method for an absolute moment block truncation coding (AMBTC) images, which is a compressed grayscale image. It is not easy to hide secret data in an AMBTC-compressed image because it is composed of bit planes. Thus, it is very sensitive to change some pixels. Nevertheless, to improve the hiding capacity, we present an efficient extension of the histogram modification technique by counting the coefficients of the bit planes in each  $4 \times 4$  block. In addition, our proposed scheme has low complexity and achieves a high embedding capacity with the good perceptual quality compared to the prior arts. In an experiment, we verified our proposed data hiding scheme through the comparison with previous methods.

**Keywords** Data hiding · AMBTC · Reversible · Histogram modification · Block Truncation Coding (BTC)

## 1 Introduction

Digital contents based on block truncation coding (BTC) [1, 2] need to be protected from illegal users that can redistribute the contents anywhere. Thus, watermarking [3, 5] and data hiding [4, 6–8] techniques are used to protect them. Furthermore, these techniques are also applied to private communications when transmitting various multimedia contents across the Internet. However, data hiding techniques cannot guarantee the safety of a message. Data hiding only conceals the existence of a message in an image, while cryptography [9] protects its content.

BTC is a lossy compression, i.e., it reduces the file size but loses some information originally present in the image. Since its introduction, BTC has been used in various applications such as the compression of graphics, video signals, and color images. However, BTC has two drawbacks. First, it produces a low-level image quality. Second, its compression rate is lower than that of JPEG [10] or JBIG [11].

A data hiding scheme based on the least significant bit (LSB) can be easily changed to embed a hidden secret message. However, hiding data in a BTC image is a more challenging task, since any pixel change can be easily detected by attackers, because it is composed of bit planes. To compensate for this defect, a diffusion algorithm was introduced for a BTC image [12]. In a diffused image, colors that are not available in the palette are approximated by the diffusion of colored pixels from within the existing palette. The human eye perceives the diffusion as a mixture of the colors within it.

---

✉ Cheonshik Kim  
mipsan@paran.com

✉ Dongkyoo Shin  
shindk@sejong.ac.kr

Lu Leng  
lenglu@126.com

Ching-Nung Yang  
cnyang@mail.ndhu.edu.tw

<sup>1</sup> Department of Computer Science and Engineering, Sejong University, Seoul, Republic of Korea

<sup>2</sup> School of Software, Nanchang Hangkong University, Nanchang 330063, People's Republic of China

<sup>3</sup> School of Electrical and Electronic Engineering, College of Engineering, Yonsei University, Seoul 120749, Republic of Korea

<sup>4</sup> Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien 97401, Taiwan

This paper proposes good solutions providing quality or coding gain improvements. The data hiding method for a BTC image uses the technique of flipping a bit plane in a block. Most schemes cannot recover the original image, because of the irreversible distortion introduced. This may not satisfy the requirements of some applications, where the content accuracy of the host medium must be guaranteed.

In the data hiding area, we are mainly concerned with how to hide more messages or more securely and reconstruct the original images. In 2004, Lin and Chang [13] proposed a data hiding scheme by performing LSB substitution operations on BTC high and low means and introducing the minimum distortion algorithm for BTC bit planes. In 2006, Chuang and Chang [14] embedded data in the BTC bit planes of smooth regions to obtain an improved embedded image quality. However, they did not reconstruct the original cover image. Reversible data hiding techniques are mainly classified into three main categories. The first is based on data compression [15–17], the second based on pixel value difference expansion [18–20], and the third based on histogram shifting [21–24]. As listed in Table 1, these types of techniques have advantages and disadvantages.

The reversible data hiding scheme for BTC images was proposed in 2008 by Hong et al. [23], where the secret data are embedded by toggling or preserving the BTC bit planes according to the secret bit and the relationship between the high mean and low mean. For each AMBTC-encoded block  $i$  with trio  $(a_i, b_i, B_i)$ , where the trio is composed of a high mean, low mean, and bit plane, respectively, if the embedded bit is “1,” then  $(a_i, b_i, B_i)$  is changed to  $(b_i, a_i, B'_i)$ , and otherwise, it remains the same. Although this scheme is reversible, when the high mean equals the low mean, there will be some problem in secret data extraction.

To overcome this problem [25], Chen et al. [26] recently proposed an improved reversible hiding scheme. The difference in the quantized values for each block is used to determine whether only “1” bit of the secret is hidden for the block or to toggle bits in the bitmap to hide more bits.

Although this scheme is reversible, when the high mean equals the low mean, it is impossible to hide a secret bit. Sun et al. [27] proposed a reversible data hiding scheme based on the joint neighbor coding technique. BTC-compressed data can be represented by a high mean table, low mean table, and bit plane sequence. This scheme is based on the relationships between the current value and the neighboring ones in mean tables. Thus, this scheme is to average 2 bits in each mean value. Lo et al. [28] proposed a reversible data hiding scheme for BTC, where they employed a histogram shifting technique to embed the secret data into the quantization levels of the compressed codes. There was no problem with recovering the original BTC image. However, the embedding capacity was not higher than the other previous schemes [25–27].

A histogram represents the statistical distribution of the pixels in an image. It is widely applied in many research fields, e.g., image enhancement, image indexing, and pattern recognition. In addition, it is possible to apply reversible data hiding. A histogram-based reversible data hiding method was introduced by Ni et al. in [21], where the message was embedded within the coefficients of the histogram. Embedding is done by shifting the peak and zero points of the histogram. In addition, the histogram shifting technique brings about overflow and underflow problems. Overflow is the condition where the gray value exceeds 255. Underflow is the condition where the gray value falls below 0. Embedding based on histogram modification has been presented in [21–23]. One of the major issues associated with all these techniques is that the peak and zero points need to be embedded along with the data during image embedding.

However, a histogram modification scheme cannot be directly applied to AMBTC [29] images. Therefore, we propose a simple and efficient reversible data hiding algorithm, which is an improved version of the histogram modification for AMBTC images.

The rest of this paper is organized as follows: In Sect. 2, we explain the related works. The proposed AMBTC histogram modification scheme is illustrated via examples and

**Table 1** Compare merits and demerits among three data hiding types

Type	Advantages	Disadvantages
Compression (DCT, DWT)	Strong from various attacks and conversions, e.g., cropping, resizing, and contrast	Low capacity and quality
Difference expand	It can also be applied to various formats (e.g., digital audio and video) Simple algorithm Easy to implement	Weakness for common signal processing operations (DE scheme based on LSB)  Low quality
Histogram shifting	High capacity and quality Computational complexity is low	Need location map to reconstruct an original image

algorithms in Sect. 3. In Sect. 4, the experimental results are given. Finally, some conclusions are made in Sect. 5.

## 2 Related works

### 2.1 Block truncation coding (BTC)

Delp and Mitchell [1] proposed BTC, which is a lossy compression technique for grayscale images. In their work, the image was divided into blocks of  $M \times N$  pixels, and each block was processed independently. They then calculated the mean value ( $\mu$ ) and standard deviation ( $\sigma$ ) of each block.

In addition, the first two sample moments were preserved in the compression. Each original block could be encoded with “0’s” and “1’s.” When a pixel value was smaller than the mean of the block, it was set to “0,” and otherwise, it was set to “1.” Block decompression required the  $\alpha$  and  $\beta$  values of each block of the image. Each block  $\mathbf{B}$  was composed of “0’s” and “1’s.” When BTC was decoded as it was uncompressed, each “0” bit of  $\mathbf{B}$  was set to  $\alpha$ , and each “1” bit of  $\mathbf{B}$  was set to  $\beta$ , where  $\alpha$  and  $\beta$  were computed according to Eqs. (1) and (2), respectively, and  $q$  and  $m$  represent the number of “1” and “0” bits in  $\mathbf{B}$ , respectively.

$$\alpha = \mu - \sigma \times \sqrt{\frac{q}{(m - q)}} \tag{1}$$

$$\beta = \mu + \sigma \times \sqrt{\frac{(m - q)}{q}} \tag{2}$$

BTC is a very simple algorithm. Hence, anybody can easily implement this technique. However, it does not provide good quality.

### 2.2 Absolute Moment Block Truncation Coding (AMBTC)

AMBTC [29], which was proposed by Lema and Mitchell in 1984, is a version of the BTC, a lossy compression algorithm for grayscale or color images. The processing time for the BTC algorithm [1] has significant computational complexity, and therefore, it is not generally recommended for time-consuming applications. The grayscale image to be encoded is divided into non-overlapping blocks of size  $(4 \times 4)$  or  $(8 \times 8)$ , for example, and the average of the blocks is calculated as in Eq. (3):

$$\bar{x} = \frac{1}{W \times H} \sum_{i=1}^N x_i \tag{3}$$

where  $x_i$  denotes the  $i$ th pixel,  $N$  is the number of pixels in the block, and  $W$  and  $H$  are the width and height of the

block, respectively. A pixel of a grayscale image is composed of 8 bits, but that of a binary image is 1 bit. Therefore, we must convert the grayscale image pixels into binary values for AMBTC. Equation (4) shows how to construct a bit plane for AMBTC from the grayscale image. If  $x \geq \bar{x}$ , then “1” is assigned to  $b_i$ , and otherwise, “0” is assigned to  $b_i$ :

$$b_i = \begin{cases} 1, & \text{if } (x_i \geq \bar{x}), \\ 0, & \text{if } (x_i < \bar{x}), \end{cases} \tag{4}$$

The means  $\alpha$  for the higher range and  $\beta$  for the lower range are calculated using Eqs. (5) and (6), respectively:

$$\alpha = \left[ \frac{1}{t} \sum_{x_i \geq \bar{x}} x_i \right] \tag{5}$$

$$\beta = \left[ \frac{1}{(W \times H) - t} \sum_{x_i < \bar{x}} x_i \right] \tag{6}$$

where  $t$  is the number of pixels in a block with a gray level greater than  $\bar{x}$ . A binary block  $\mathbf{B}$  contains the bit planes that represent the pixels. The values of  $\alpha$  and  $\beta$  are used to decode the AMBTC-compressed image; every “1” in block  $\mathbf{B}$  is replaced by  $\alpha$  and every “0” is replaced by  $\beta$ :

$$g_i = \begin{cases} \alpha, & \text{if } (b_i = 1) \\ \beta, & \text{if } (b_i = 0) \end{cases} \tag{7}$$

where  $g_i$  is a grayscale pixel, and thus, the grayscale image is reconstructed. The compression rate for AMBTC is 2 bpp, because the total number of bits required for a block is 32 bits. BTC has a complex computation, while AMBTC has a simple computation and therefore requires less computation time than BTC.

Figure 1a shows the original image block of  $4 \times 4$  pixels. First, we compute the mean value of the block using Eq. (4). This gives a mean value of 161.25. Figure 1b shows the bitmap of an original grayscale image using Eq. (4). Then, the quantization levels for these two groups ( $\alpha$  and  $\beta$ ) are calculated using Eqs. (5) and (6), respectively. These two quantization levels are  $161 = \lfloor 160.5 \rfloor$  and 162, respectively. Each compressed image block forms a trio ( $\alpha, \beta, \text{bitmap}$ ) where each quantization level is stored in 8 bit. Finally, the compressed trio  $(161, 162, (1100110011001100)_2)$  is sent to the receiver.

An example of the image decoding procedure is described in the following. Suppose that the compressed trio  $(161, 162, (1100110011001100)_2)$  is received by the receiver. A pixel in the block is reconstructed by 161 if a corresponding bit valued “0” is found. Otherwise, it is reconstructed by 162. The reconstructed block of this example is shown in Fig. 1c.

According to the AMBTC algorithm, the bitmap image is shown in Fig. 2b, c is also reconstructed using Eqs. (5),

**Fig. 1** Example of AMBTC encoding and decoding: **a** original image block, **b** bitmap, and **c** reconstructed image block

162	162	161	160
162	162	161	160
162	162	161	160
162	162	161	160

(a)

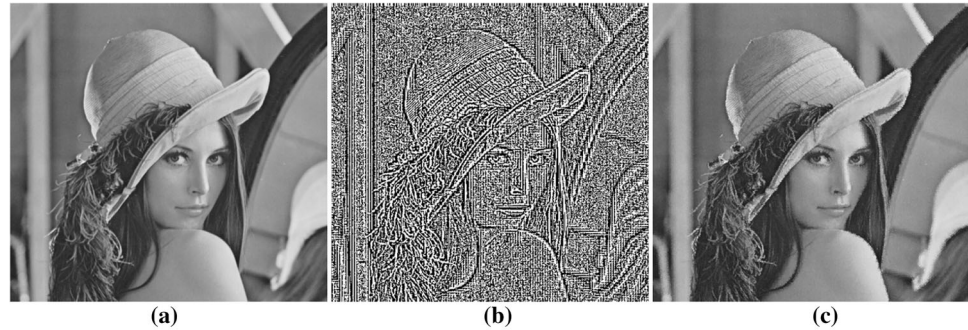
1	1	0	0
1	1	0	0
1	1	0	0
1	1	0	0

(b)

162	162	161	161
162	162	161	161
162	162	161	161
162	162	161	161

(c)

**Fig. 2** AMBTC example for “Lena” image. **a** Original Lena (512 × 512), **b** Bitmap, **c** AMBTC



(6), and (7). It becomes a grayscale image almost the same as the original “Lena.”

### 2.3 Histogram pairs method

Many researchers [15–23] have proposed reversible data hiding schemes that use histogram pairs of grayscale images. In the histogram, we first search for a *zero point* and then a *peak point*. A *zero point* corresponds to the grayscale value that is not possessed by any pixel in the given image. A *peak point* corresponds to a grayscale value equal to the largest number of pixels in the image. We find a *peak point* to raise the embedding capacity as much as possible [21]. In the following, we will refer to a data embedding scheme stage by stage. For an  $M \times N$  image, each pixel a grayscale value  $x \in [0, 255]$ :

- Step 1: Generate its histogram  $H(x)$ .
- Step 2: In the histogram  $H(x)$ , find the maximum point  $h(\alpha)$ ,  $\alpha \in [0, 255]$  and the minimum point  $h(\beta)$ ,  $\beta \in [0, 255]$ .
- Step 3: If the minimum point  $h(\beta) \geq 0$ , then record the coordinates  $(i, j)$  of those pixels and the pixel grayscale value  $\beta$  as overhead book-keeping information (referred to as *overhead information* for short), and set  $h(\beta) = 0$ .
- Step 4: Without loss of generality, assume that  $\alpha < \beta$ . Move the whole part of the histogram  $H(x)$  with  $x \in (\alpha, \beta)$  to the right by 1 unit. The result is that all of the pixel grayscale values in the interval  $(\alpha, \beta)$  are incremented by 1.

- Step 5: Scan the image until meeting the pixel whose grayscale value is  $\alpha$ ; check the to-be-embedded bit. If the to-be-embedded bit is 1, the pixel grayscale value is changed to  $\alpha + 1$ . If the bit is 0, the pixel value remains  $\alpha$ .

In this way, the actual data embedding capacity  $C$  is calculated as follows:

$$C = h(\alpha) - O \tag{8}$$

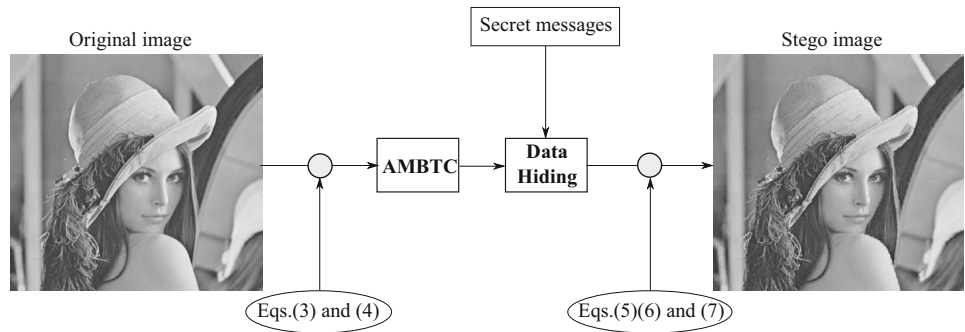
where  $O$  denotes the amount of data used to represent the overhead information. We refer to  $C$  as pure payload. Clearly, if the required payload is greater than the actual capacity, more pairs of maximum and minimum points need to be used.

### 3 Proposed data hiding scheme

The histogram modification method is an excellent reversible data hiding technique for grayscale images. But this method cannot be directly applied to AMBTC images with existing previous schemes for grayscale images, because a BTC image is composed of bit planes. In order to apply that method to AMBTC images, the *run lengths* of the binary data are counted to generate and shift histogram bins. In this paper, we solve this problem and conceal a large quantity of information in an AMBTC image.

In this section, we shall present the proposed scheme, including embedding and extracting processes. Figure 3 shows the block diagram to explain data hiding procedure.

**Fig. 3** Block diagram for our proposed scheme



**Fig. 4** Scan order of  $4 \times 4$  image block

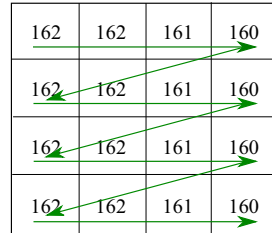
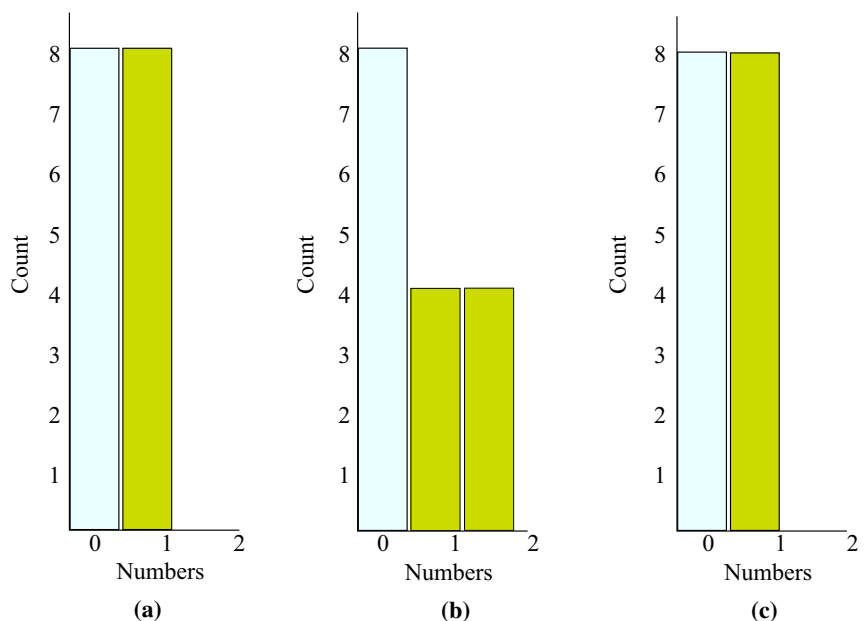


Figure 3 shows the procedure for our proposed scheme with AMBTC. That is, a bitmap of AMBTC is produced using Eqs. (3) and (4), and then, the histogram modification algorithm (including the embedding and extracting schemes) is applied to the bitmap image. Finally, we obtain the stego image embedded secret messages. To improve the hiding capacity, we present an efficient extension of the histogram modification technique by counting the coefficients of the bit planes in each  $4 \times 4$  block in an image while scanning from left to right and from top to bottom, as shown in Fig. 4.

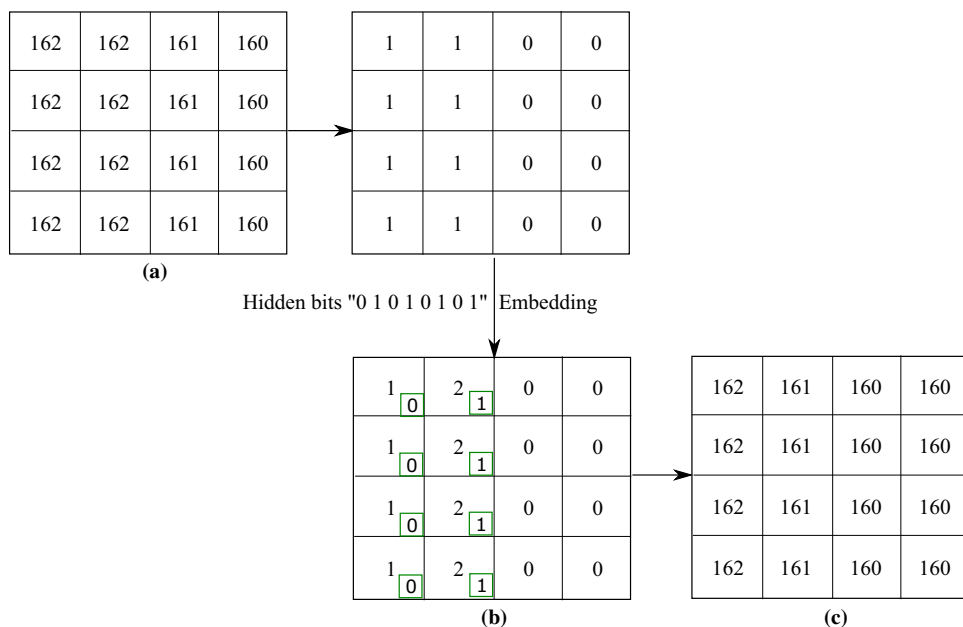
Our proposed scheme offers a high embedding capacity while maintaining low distortion. Figure 5a shows a histogram of Fig. 1b, where the number of “0” pixels is 8 and “1” pixels is 8. Exploiting the bit planes in a block, we decide to add bin 2 as shown in Fig. 5a. Thus, bin 2 would be empty. After the bins are ready, it begins data embedding by dealing out bin 1, according to the message. By doing this, we exploit “1” bits as shown in Fig. 5b. That is, if the secret bit is “0,” keep “1” (bin 1) values without moving to 2’s (bin 2). Otherwise, the “1’s” (bin 1) move to 2’s (bin 2). Figure 5b shows the result of the embedding procedure.

Figure 6 shows an example of hiding secret bits in a  $4 \times 4$ . The method used to hide secret bits in a block is explained in Fig. 5. Figure 6b shows that the hidden bits (e.g., “0 1 0 1 0 1 0 1”) are concealed in a block with histogram modification as in Fig. 5b. Figure 6c shows a reconstructed grayscale image based on AMBTC, i.e., it is possible to obtain the block using Eqs. (4), (5), (6), (9), and (10).

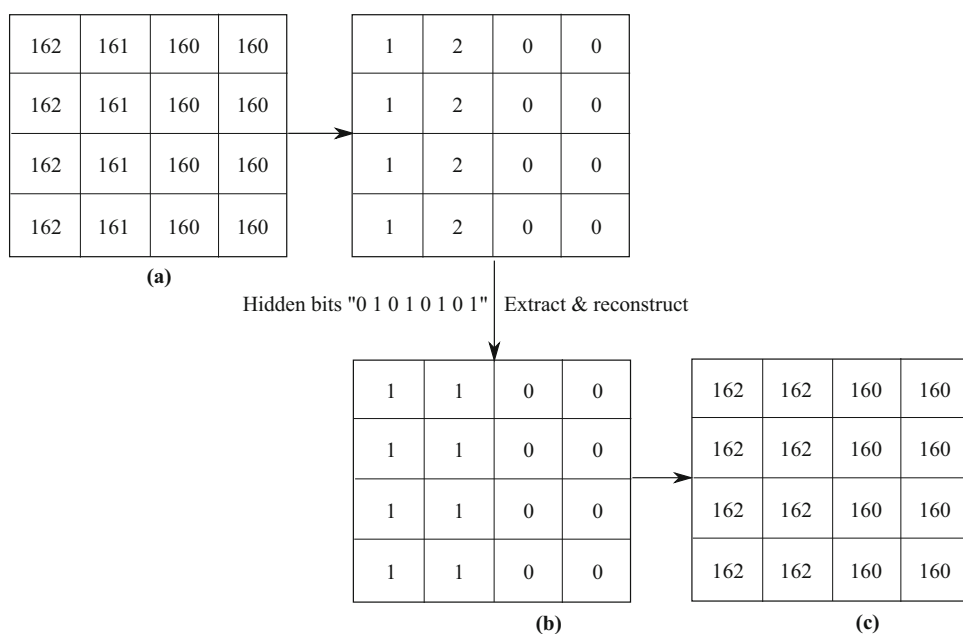
**Fig. 5** Concept of data embedding using histogram modification for AMBTC image: **a** original histogram, **b** bitmap planes, **c** stego BTC block



**Fig. 6** Example of data hiding with AMBTC: **a** original image block, **b** bitmap (including hidden bits), **c** stego image block



**Fig. 7** Example of extraction and reconstruction using stego AMBTC: **a** stego grayscale block, **b** bitmap (restore original bitmap from stego bitmap), and **c** reconstructed image block



$$\gamma = \left\lfloor \frac{(\alpha + \beta)}{2} \right\rfloor \tag{9}$$

$$g_i = \begin{cases} \alpha, & \text{if } (b_i = 1), \\ \beta, & \text{if } (b_i = 0), \\ \gamma, & \text{if } (b_i = 2), \end{cases} \tag{10}$$

To reconstruct Fig. 6c, the bit planes for “1’s” and “0’s” change into  $\alpha$  and  $\beta$ , respectively, using Eqs. (5), (6), and (10). In the case of the bit “2’s” values, it is changed into  $\gamma$  using Eqs. (9) and (10).

Figure 7 shows an example of extracting the hidden bits from the stego bit planes, reconstructing the original bit planes from the stego bit planes, and then restoring the original grayscale image. That is, if a pixel in a stego block is a “1,” the extracted hidden bit is “0.” Otherwise, in the case a “2,” the hidden bit is “1,” and it could be reconstructed by changing it into a “1.” Figure 7b shows an original block of reconstructed bit planes. Figure 7c shows a grayscale image block based on AMBTC that is restored using Eqs. (4), (5), and (6).

In Sects. 3.1 and 3.2, we explain our proposed scheme, including the embedding and extraction algorithms.

### 3.1 Embedding method

This section shows the embedding procedure based on the bit planes of AMBTC using histogram modification, according to the hidden messages.

**Input:** Original grayscale image **GI** with  $M \times N$  pixels and secret data  $\delta = [b_1, b_2, \dots, b_n]$

**Output:** Stego image **SI**, length of secret data  $|\delta|$

- Step 1: First, divide the image to be coded into small non-overlapping image blocks (typically the blocks of size  $4 \times 4$  pixels to achieve reasonable quality). Each pixel block can be treated as a vector with a size of  $M \times N$ . Then, let  $x_i$  be the pixel values of each vector. Next, we compute the mean value  $\bar{x}$  for the pixels of each pixel block using Eq. (3). If a pixel is greater than or equal to the block mean, the corresponding pixel position of the bitmap will have a value of “1”; otherwise, it will have a value of “0.” As a result of this procedure, we could obtain a bitmap image (**BI**), which is composed of “0’s” and “1’s.”
- Step 2: Read a block from **BI** using Eq. (11), where  $i$  and  $j$  are indexes of an image, and  $T$  is a block. Count the number of “0” and “1” bits using Eq. (12).  $S_0$  and  $S_1$  are the sums of all the “0’s” and “1’s” in  $T$ , respectively.

$$T = \sum_{i=1}^{M/4} \sum_{j=1}^{N/4} BI_{i+3,j+3} \tag{11}$$

$$S_0, S_1 = \begin{cases} S_0 + 1, & \text{if } (T_{m,n}^{4 \times 4} = 0), \\ S_1 + 1, & \text{if } (T_{m,n}^{4 \times 4} = 1), \end{cases} \tag{12}$$

- Step 3: If  $S_0$  or  $S_1$  is all “0’s” or “1’s,” go to step 2.
- Step 4: Scan every element in a block  $T$ . If the scanned value is equal to “1” and hidden bit  $\delta_\varphi$  is “0<sub>b</sub>,” it does not need to move. If the scanned value is equal to “1” and hidden bit  $\delta_\varphi$  is “1<sub>b</sub>,” move to “2” using Eq. (13).

$$SI_{ij} = \begin{cases} 1, & \text{if } (T_{m,n}^{4 \times 4} = 1 \cap \delta_{\varphi++} = 0), \\ 2, & \text{if } (T_{m,n}^{4 \times 4} = 1 \cap \delta_{\varphi++} = 1), \end{cases} \tag{13}$$

- Step 5: Go to step 2 to continue the embedding processes until all of  $\delta$  is scanned.
- Step 6: If the process of data hiding is complete, it has produced the bit planes of image **SI**.

### 3.2 Extraction and reconstruction method

The extraction and reconstruction procedures are shown below:

**Input:** Stego image **SI** and the length of secret data  $|\delta|$   
**Output:** Original image **BI** and secret data  $\delta$

- Step 1: Scan image **SI** and divide it into a  $4 \times 4$  group with non-overlap.
- Step 2: Read a block from **SI** using Eq. (14), where  $i$  and  $j$  are indexes of an image, and  $T$  is a block, and then count the number of “0” and “1” bits using Eq. (12).

$$T = \sum_{i=1}^{M/4} \sum_{j=1}^{N/4} SI_{i+3,j+3} \tag{14}$$

- Step 3: If  $S_0 = 16$  or  $S_1 = 16$ , then go to step 2.
- Step 4: Scan every pixel in a block  $T$ . If the scanned pixel value is equal to “1,” the hidden bit is “0,” which is combined with a secret message  $\delta$  using Eq. (15). In contrast, if the scanned pixel value is “2,” the extracted bit value is “1,” which is combined at  $\delta$  using Eq. (15). In addition, if the pixel is “2,” move to “1” to reconstruct the original bit planes using Eq. (16).

$$\delta_{\varphi++} = \begin{cases} \delta + '0', & \text{if } (T_{m,n}^{4 \times 4} = 1), \\ \delta + '1', & \text{if } (T_{m,n}^{4 \times 4} = 2), \end{cases} \tag{15}$$

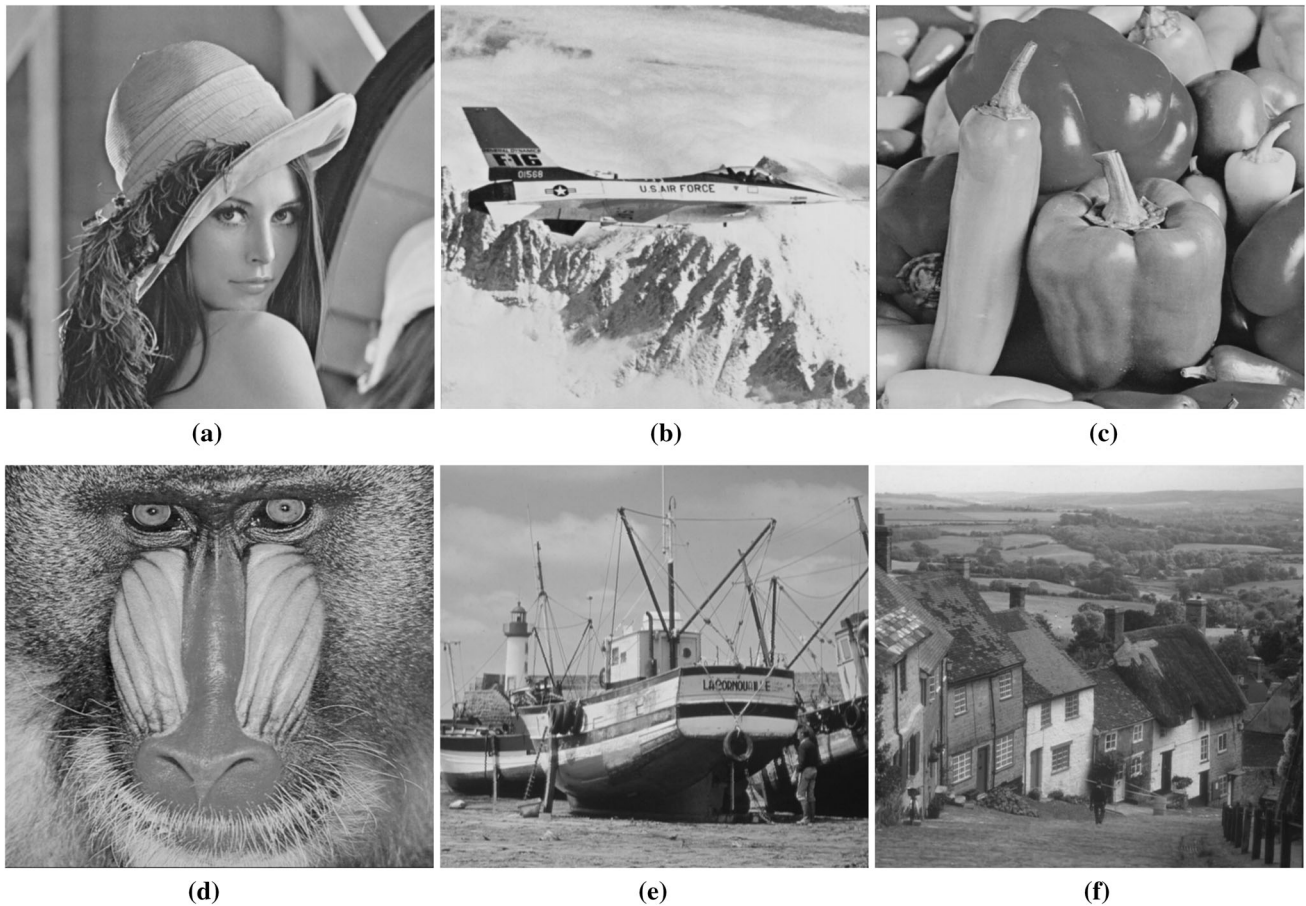
$$BI(m, n) = \begin{cases} \text{don't move} & \text{if } (T_{m,n}^{4 \times 4} = 1), \\ 1 & \text{if } (T_{m,n}^{4 \times 4} = 2), \end{cases} \tag{16}$$

- Step 5: Repeat step 2 until  $\varphi = |\delta|$ .
- Step 6: At this step, most of the coefficient values are restored, and the bitmap image (**BI**) could be reconstructed.

## 4 Experimental results

In this paper, we propose a novel reversible data hiding scheme using a histogram modification method. In order to prove the performance of our proposed scheme, we need to compare its performance with those of previous methods. Six  $512 \times 512$  images (“Lena,” “Airplane,” “Boat,” “Pepper,” “Baboon,” and “F16”) [30] (Fig. 8) were used in the experiment. In addition, the block size for the AMBTC compression method was set at  $4 \times 4$ .

The simulation environment for the experiments was a 2.66 GHz, Core 2, Quad system with 8 GB of RAM and a MATLAB compiler.



**Fig. 8** Grayscale images ( $512 \times 512$ ) for the experiment. **a** Lena. **b** F16. **c** Pepper. **d** Baboo. **e** Boat. **f** Goldhill

To evaluate our proposed scheme, we used the peak signal-to-noise ratio (PSNR) [6] and hiding capacity. The PSNR is defined as Eq. (17), where the mean squared error (MSE) is computed by Eq. (18). A higher PSNR value indicates a better quality for the stego image. The hiding capacity denotes the number of bits that can be hidden in the AMBTC image.

$$\text{PSNR} = 10 \log_{10} \frac{255^2}{\text{MSE}} \quad (17)$$

$$\text{MSE} = \frac{1}{M \times N} \sum_{i=1}^M \sum_{j=1}^N (x_{ij} - x'_{ij})^2 \quad (18)$$

where  $M \times N$  is the size of the image, and  $x_{ij}$  and  $x'_{ij}$  denote the pixel values of the cover image and stego image in the same position ( $i, j$ ), respectively.

It is well known that an AMBTC image is very sensitive to flipping the bits of the pixels. Thus, in the experiments, two performance aspects were adopted to evaluate the data hiding scheme, i.e., the capacity represents the maximum number of secret bits that can be hidden and the PSNR represents the quality of the stego image.

First, we tested the performance of our algorithm under different embedding rates. Therefore, to examine the influence of flipping a large number of pixel bits, we compared the relation between the embedding rate (from 0.1 to 0.8 bpp) of the hidden bits in an image and PSNR, as listed in Table 2. As given in Table 2, the embedding rates steeply increased, while the PSNRs gradually decreased. In these cases, there were a few different PSNRs between the neighboring embedding rates, which proved that the performance of our proposed scheme was good. In addition, the PSNRs in Table 2 are relatively high values, so the quality of the stego images was very good.

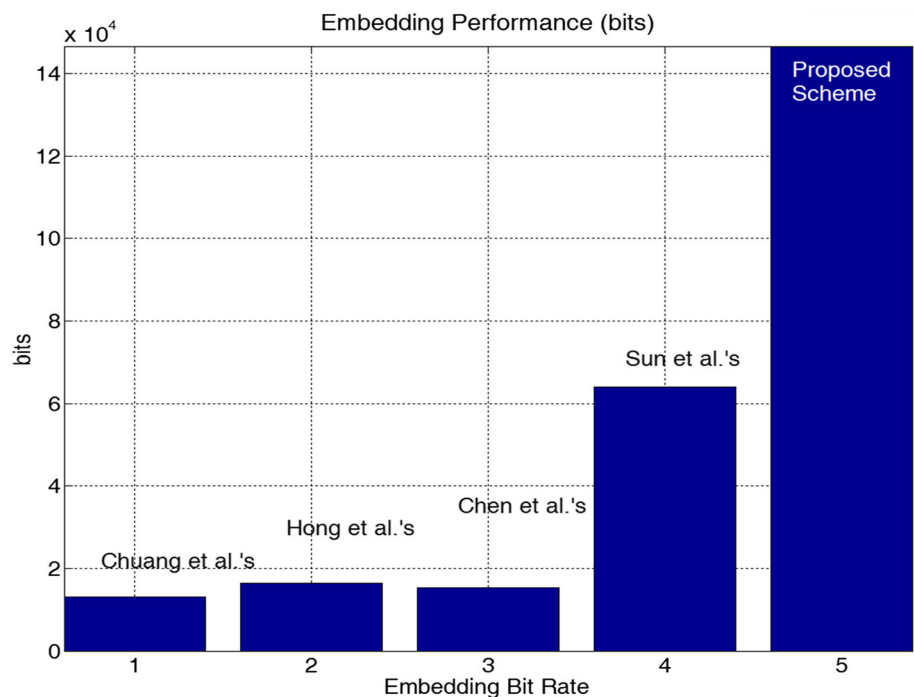
Afterward, using the same block size ( $4 \times 4$ ), we compared our algorithm with the methods of Chuang and Chang [14], Hong et al. [25], Chen et al. [26], and Sun et al. [27]. Figure 9 shows a comparison of the embedding bits with the “Lena” image using the previous schemes and our proposed scheme. According to Fig. 9, our proposed embedding scheme had the best performance. That is, the number of hidden bits in the “Lena” image based on our proposed scheme was 146,598 bits. However, the PSNR of



**Table 2** Comparison of relation between embedding rate and PSNR

Images	Performance							
	Embedding rate							
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
Lena								
Bits	14,659	29,319	43,979	58,639	65,535	87,958	102,618	117,278
dB	35.448	35.426	35.364	35.324	35.281	35.224	35.143	35.167
Baboon								
Bits	14,618	29,237	43,855	58,474	73,093	87,958	102,618	117,278
dB	28.353	28.335	28.308	28.313	28.319	28.319	28.317	28.323
Pepper								
Bits	14,910	29,321	44,731	59,642	74,552	89,463	104,373	119,284
dB	35.425	35.380	35.274	35.151	34.952	34.992	34.932	34.949
Airplane								
Bits	15,251	30,502	45,753	61,004	76,255	91,506	106,757	122,008
dB	34.148	34.145	34.130	34.056	33.693	33.571	33.467	33.369
Goldhill								
Bits	14,572	29,144	43,716	58,288	72,860	87,432	102,004	116,576
dB	34.361	34.341	34.318	34.324	34.339	34.338	34.252	34.216
Boat								
Bits	15,092	30,184	45,276	60,368	75,460	90,552	105,644	120,736
dB	33.440	33.315	33.337	33.235	33.100	32.964	32.927	32.876

**Fig. 9** Comparison of embedding capacities of data hiding schemes



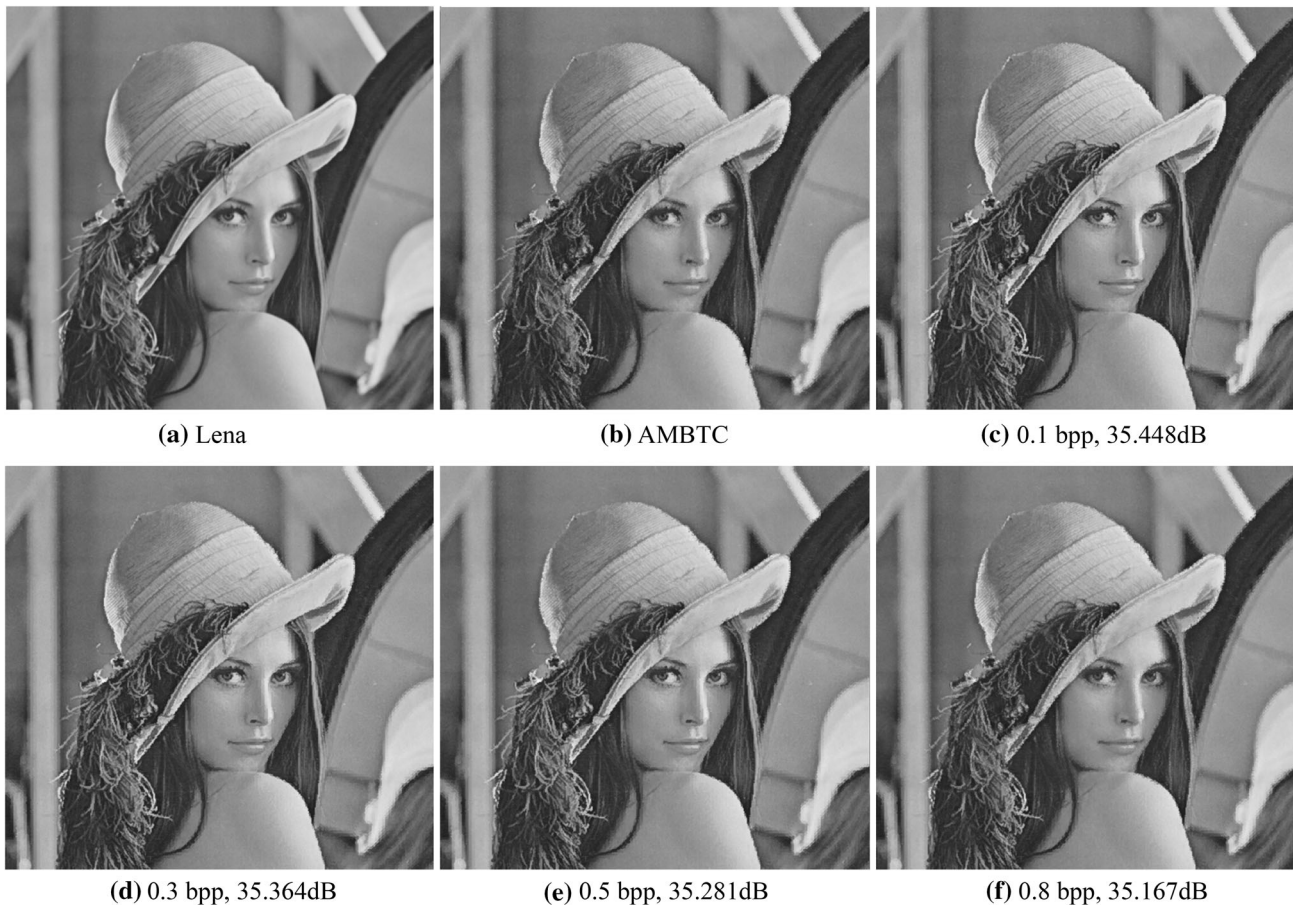
Lo et al.'s method [28] is slightly low and its embedding capacity is lower than that of the other schemes. With respect to the embedding capacity and PSNR, our proposed scheme achieved the highest embedding capacity and good quality. The scheme of Sun et al. [27] embedded 2 bits in each mean value, including the low mean table and high

mean table. Thus, it was possible to embed 4 bits in each 4 × 4 pixel block. The capacities of the schemes of Hong et al. [25] and Chen et al. [26] were normally 1 bit in each 4 × 4 pixel block.

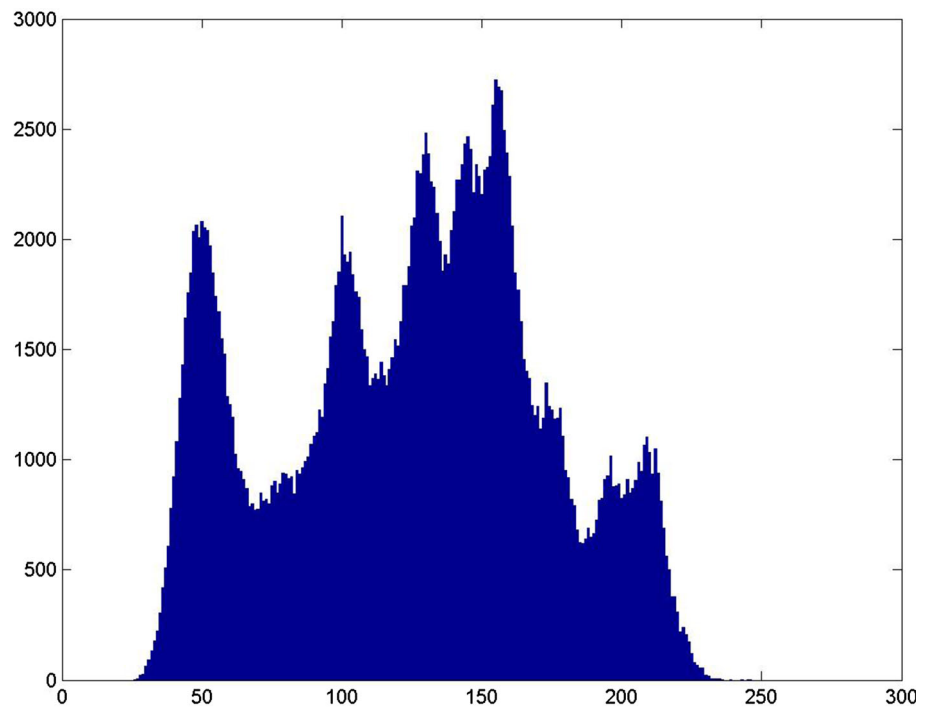
As listed in Table 3, the capacity of our proposed scheme is about 9 times higher than those of the other

**Table 3** PSNR and capacity comparisons between previous schemes and proposed scheme

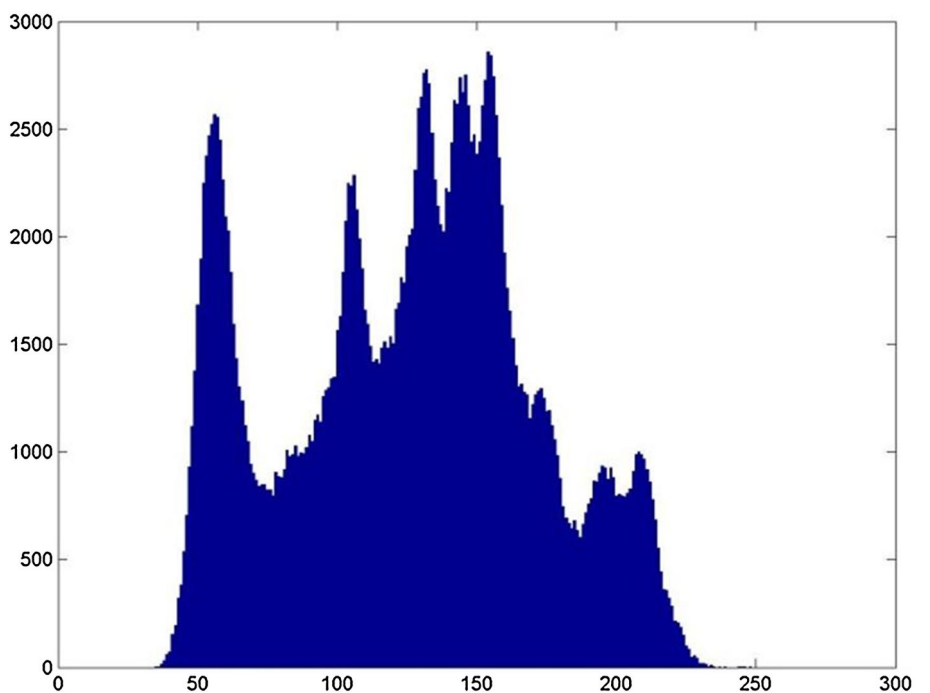
Method	Performance					
	Images					
	Lena	Peppers	Baboon	Boat	Goldhill	F16
AMBTC						
PSNR (dB)	33.971	33.400	25.995	31.147	33.161	31.949
Proposed						
Bits	146,598	146,186	149,105	150,920	145,721	152,510
PSNR (dB)	34.455	35.788	28.353	33.444	34.371	34.155
Hong et al.'s [25]						
Bits	16,384	16,099	16,384	16,384	16,384	16,384
PSNR (dB)	33.278	33.324	25.187	31.129	33.128	31.390
Chen et al.'s [26]						
Bits	16,384	20,944	16,384	16,394	16,384	16,384
PSNR (dB)	33.165	33.301	25.891	31.135	33.101	31.435
Sun et al.'s [27]						
Bits	64,008	64,008	64,008	64,008	64,008	64,008
PSNR (dB)	33.971	33.400	25.995	31.147	33.161	31.949
Lo et al.'s [28]						
Bits	4025	4882	1201	5487	2410	7073
PSNR (dB)	33.364	33.873	28.132	31.751	33.435	31.751

**Fig. 10** Quality comparison of Lena images, including (a) original Lena, b AMBTC, c–f stego AMBTC

**Fig. 11** Histogram of original Lena image



**Fig. 12** Histogram after Gaussian filtering

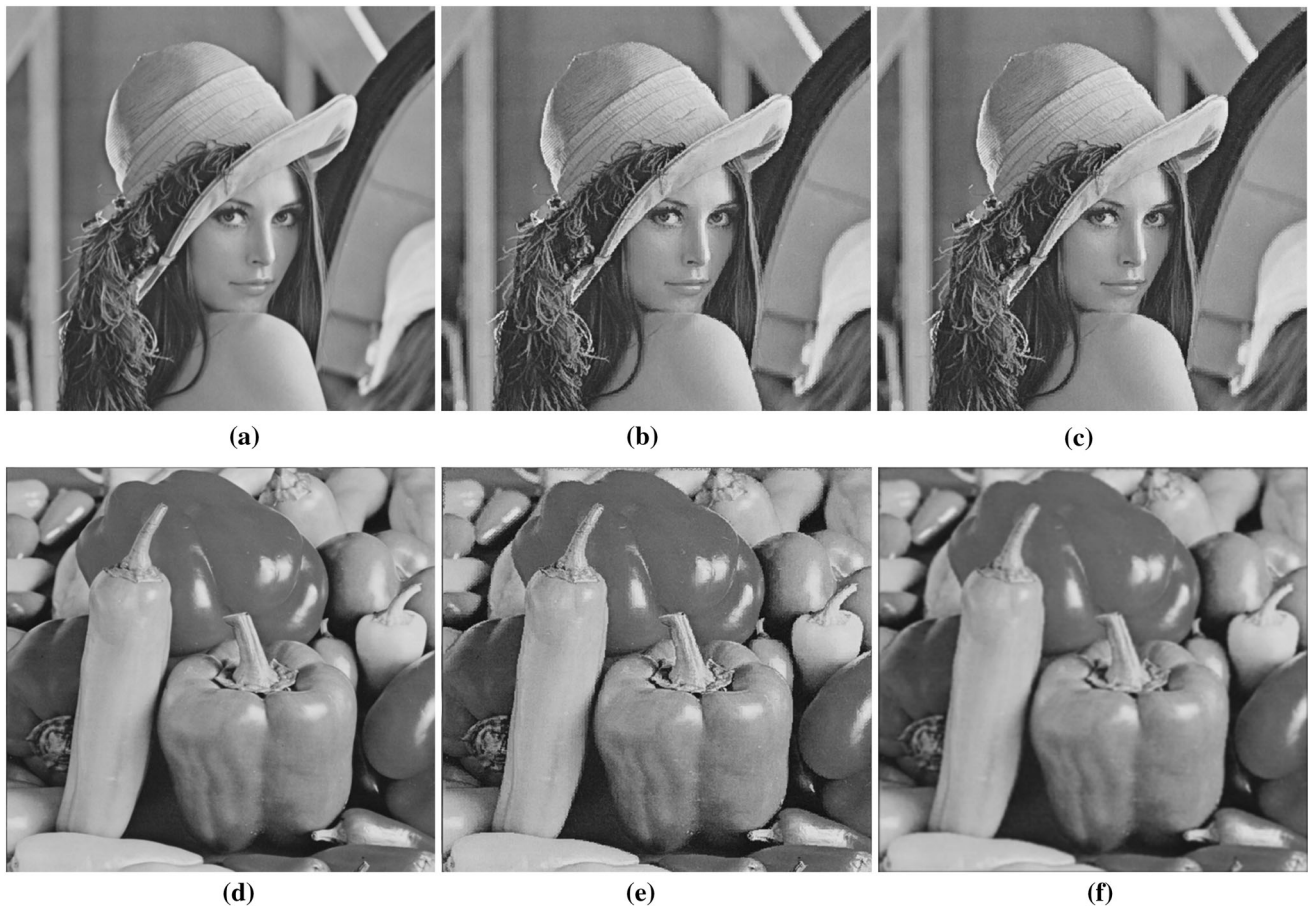


schemes (Sun et al., Hong et al., and Chen et al.). Its significant dominance in relation to the neighboring bins is important in terms of the embedding capacity.

The AMBTC  $512 \times 512$  “Lena” image with Hong et al.’s method can hide a total of 16,384 bits at one time. On the other hand, it is possible to hide 146,598 bits at one time with our method. Figure 10 shows a quality comparison of the images, including the original (a) Lena,

(b) AMBTC, and (c)–(f) stego images. As shown in Fig. 10, it would be difficult to detect which is original image or stego image. Therefore, it is shown that the performance of our proposed scheme is good enough for data hiding based on AMBTC.

After extracting the secret data from the stego image, the original AMBTC image could be reconstructed. However, this does not mean that the reconstructed AMBTC image is



**Fig. 13** “Lena” and “Peppers” images after Gaussian filtering. **a** Original Lena. **b** Stego 0.1 bpp, 35,448 dB. **c** Gaussian, 37,610 dB. **d** Original Peppers. **e** Stego 0.1 bpp, 35,425 dB. **f** Gaussian, 37,132 dB

the original grayscale image, because there is a quality difference between them. To improve the quality of an AMBTC image, we apply a Gaussian filter to this image. It has two variations, involving the filter size and standard deviation. We performed numerous experiments to determine the size of the filter and standard deviation. In this work, the size of the filter is fixed at  $7 \times 7$  with a standard deviation  $\delta = 1.3$ , which is used to simulate the human visual response.

Figures 11 and 12 show histograms of the original “Lena” image and stego “Lena” image, respectively. We are interested in comparing them, because this makes it possible to determine how much the stego is similar to the original image. If they are similar, it is not easy for an attacker to detect the stego image.

Therefore, the usefulness of our proposed scheme has been amply proven. Figure 13 shows the stego “Lena” and “Peppers” images after Gaussian filtering. Figure 13b shows a quality improvement of about 2 dB. In addition, our method has low complexity and achieves a high embedding capacity with good perceptual quality compared to the prior arts.

The complexity of our proposed scheme mainly lies on generating histogram, determining minimum and maximum (and possibly subminimum and submaximum) points, scanning pixels, and adding or subtracting pixel grayscale values by one in the spatial domain. Hence, the execution time of the algorithm is rather short. Assume the image height is  $M$  and the width is  $N$  and a block height is  $H$  and the width is  $W$  and number of blocks is  $k$ . We need to scan the whole image one time in the embedding. Hence, the computational complexity is  $O(MN + kWH)$ .

## 5 Conclusion

In this paper, we presented an efficient reversible data hiding algorithm based on the histogram modification of AMBTC-compressed images. As listed in Table 2, the PSNR of our proposed scheme is higher than that of the original AMBTC-compressed image. In addition, we invented new scheme to exploit the bitmap coefficients of  $4 \times 4$  blocks of an image based on histogram modification.

Thus, the capacity of our proposed scheme is about 9 times higher than those of the other schemes. In the extraction procedure, the AMBTC-compressed image and secret messages can be reconstructed completely. As is generally known, most portable video devices do not need high-quality images. In addition, they need simple computation and high-speed time complexity. Therefore, our proposed scheme would be very appropriate for hiding data on portable devices.

**Acknowledgments** We are grateful for the support provided by the National Natural Science Foundation of China (61305010), Science and Technology Project of Education Department of Jiangxi Province (GJJ150715), Voyage Project of Jiangxi Province (201450), Open Foundation of Key Laboratory of Jiangxi Province for Image Processing and Pattern Recognition (TX201604002), and Doctoral Initiating Foundation of Nanchang Hangkong University (EA201620045). This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2015R1D1A1A01059253). This work was supported under the framework of international cooperation program managed by the National Research Foundation of Korea (2016K2A9A2A05005255, FY20\*\*).

## References

- Delp, E.J., Mitchell, O.R.: Image compression using block truncation coding. *IEEE Trans. Commun.* **27**(9), 1335–1342 (1979)
- Wu, Y.G., Tai, S.C.: An efficient BTC image compression technique. *IEEE Trans. Consum. Electron.* **44**(2), 317–325 (1998)
- Suhail, M.A., Obaidat, M.S.: Digital watermarking-based DCT and JPEG model. *IEEE Trans. Instrum. Meas.* **52**(5), 1640–1647 (2003)
- Khan, M.K., Xie, L., Zhang, J.: Robust hiding of fingerprint biometric data into audio signals. In: Lee S.W., Li S.Z. (eds.) *International Conference, ICB 2007, Lecture Notes in Computer Science*, vol. 4642, pp. 702–712. Springer, Berlin, Heidelberg (2007)
- Kim, C., Yang, C.N.: Watermark with DSA signature using predictive coding. *Multimed. Tools Appl.* **74**(14), 5189–5203 (2015)
- Kim, C.: Data hiding by an improved exploiting modification direction. *Multimed. Tools Appl.* **69**(3), 569–584 (2014)
- Yang, C.N., Ye, G.C., Kim, C.: Data hiding in halftone images by XOR block-wise operation with difference minimization. *KSII Trans. Internet Inf. Syst. (TIIS)* **5**(2), 457–476 (2011)
- Huy, P.T., Thanh, N.H., Kim, C., Yang, C.N.: Data-hiding for halftone images using an improved CPT scheme. *KSII Trans. Internet Inf. Syst. (TIIS)* **7**(2), 405–424 (2013)
- Diffie, W., Hellman, M.: New directions in cryptography. *IEEE Trans. Inf. Theory* **22**(6), 644–654 (1976)
- Neelamani, R., Queiroz, R.D., Fan, Z., Dash, S., Baraniuk, R.G.: JPEG compression history estimation for color images. *IEEE Trans. Image Process.* **15**(6), 1365–1378 (2006)
- Pfarrhofer, R., Uhl, A.: Selective image encryption using JBIG. *Lect. Notes Comput. Sci.* **3677**, 98–107 (2005)
- Guo, J.M.: Improved block truncation coding using modified error diffusion. *Electron. Lett.* **44**(7), 462–464 (2008)
- Lin, M.H., Chang, C.C.: A novel information hiding scheme based on BTC. In: *Proceedings of 4th International Conference on Computer and Information Technology*, pp. 66–71 (2004)
- Chuang, J.C., Chang, C.C.: Using a simple and fast image compression algorithm to hide secret information. *Int. J. Comput. Appl.* **28**(4), 329–333 (2006)
- Fridrich, J., Goljan, M., Du, R.: Lossless data embedding new paradigm in digital watermarking. *EURASIP J. Adv. Signal Process.* **2002**(2), 185–196 (2002)
- Kalker, T., Willems, F.M.J.: Capacity bounds and constructions for reversible data hiding. In: Delp III E.J., Wong P.W. (eds.) *Proc. SPIE 5020, Security and Watermarking Multimedia Contents V*, vol. 5020, pp. 604–611. Santa Clara, CA (2003)
- Celik, M.U., Sharma, G., Tekalp, A.M., Saber, E.: Lossless generalized-LSB data embedding. *IEEE Trans. Image Process.* **14**(2), 253–266 (2005)
- Tian, J.: Reversible data embedding using a difference expansion. *IEEE Trans. Circuits Syst. Video Technol.* **13**(8), 890–896 (2003)
- Kamstra, L., Heijmans, H.J.A.M.: Reversible data embedding into images using wavelet techniques and sorting. *IEEE Trans. Image Process.* **14**(12), 2082–2090 (2005)
- Thodi, D.M., Rodriguez, J.J.: Expansion embedding techniques for reversible watermarking. *IEEE Trans. Image Process.* **16**(3), 721–730 (2007)
- Ni, Z., Shi, Y.Q., Ansari, N., Su, W.: Reversible data hiding. *IEEE Trans. Circuits Syst. Video Technol.* **16**(3), 354–362 (2006)
- Lee, S.K., Suh, Y.H., Ho, Y.S.: Reversible image authentication based on watermarking. In: *Proceedings of the IEEE International Conference on Multimedia Expo*, pp. 1321–1324 (2006)
- Hong, W., Chen, T.S., Shiu, C.W.: Reversible data hiding for high quality images using modification of prediction errors. *J. Syst. Softw.* **82**(11), 1833–1842 (2009)
- Zhao, H., Wang, H.X., Khan, M.K.: Statistical analysis of several reversible data hiding algorithms. *J. Multimed. Tools Appl.* **52**(2), 277–290 (2011)
- Hong, W., Chen, T.S., Shiu, C.W.: Lossless steganography for AMBTC compressed images. In: *Proceedings of 1st International Congress on Image and Signal Processing*, pp. 13–17 (2008)
- Chen, J., Hong, W., Chen, T.S., Shiu, C.W.: Steganography for BTC compressed images using no distortion technique. *Imaging Sci. J.* **58**(4), 177–185 (2010)
- Sun, W., Lu, Z.M., Wen, Y.C., Yu, F.X., Shen, R.J.: High performance reversible data hiding for block truncation coding compressed images. *SIViP* **7**(2), 297–306 (2013)
- Lo, C.C., Hu, Y.C., Chen, W.L., Wu, C.M.: Reversible data hiding scheme for BTC-compressed images based on histogram shifting. *Int. J. Secur. Appl.* **8**(2), 301–314 (2014)
- Lema, M.D., Mitchell, O.R.: Absolute moment block truncation coding and its application to color images. *IEEE Trans. Commun.* **32**(10), 1148–1157 (1984)
- USC-SIPI image database website. Last Accessed 6 Apr 2016. <http://sipi.usc.edu/database/database.php>

**Cheonshik Kim** received his B.S. degree in Computer Engineering from Anyang University, Korea, in 1995; his M.S. degree in Computer Engineering from Hankuk University of Foreign Studies (HUFS), Korea, in 1997; and his Ph.D. degree in Computer Engineering from HUFS in 2003. From March 2013 to February 2016, he was a professor of Department of Digital Media Engineering, Anyang University, and Republic of Korea. From March 2016, he was a professor of Department of Computer Science and Engineering, Sejong University, and Republic of Korea. He won a research award from the IEEK in 2012. Since 2012, he has served as an editor for *ICACT Transaction on Advanced Communications Technology (TACT)*, and since 2015, he has served as an editor for the *American Journal of Circuits, Systems, and Signal Processing (Public Science Framework)*. He was a program chair at two international

conferences—GPC 2013 and FutureTech 2014—and has been a vice president for the IEEK Computer Society since 2010. He is also a member of IEEE. His research fields include multimedia systems, data hiding, and watermarking. His researches were supported by NRF (2012–2015). He was a subject of biographical record in Marquis Who's Who in the World 2013–2015.

**Dongkyoo Shin** received a B.S. in Computer Science from Seoul National University, Korea, in 1986, an M.S. in Computer Science from Illinois Institute of Technology, Chicago, Illinois, in 1992, and a Ph.D. in Computer Science from Texas A&M University, College Station, Texas, in 1997. He is currently a Professor in the Department of Computer Engineering at Sejong University in Korea. From 1986 to 1991, he worked in Korea Institute of Defense Analyses, where he developed database application software. From 1997 to 1998, he worked in the Multimedia Research Institute of Hyundai Electronics Co., Korea, as a Principal Researcher. His research interests include XML-based middleware, digital right management for multimedia, mobile Internet, and ubiquitous computing.

**Lu Leng** received his Ph.D. degree from Southwest Jiaotong University, Chengdu, P.R. China, in 2012. Currently, he is a lecturer of Nanchang Hangkong University, Nanchang, P.R. China, and a visiting scholar of West Virginia University, Morgantown, USA. He has published about 40 international journals and conference papers and been granted several scholarships and funding projects in his academic research. He is the reviewer of several international journals

and conferences. His research interests include biometric security and privacy, image processing, and biometric recognition. Dr. Leng is a member of Institute of Electrical and Electronics Engineers (IEEE), Association for Computing Machinery (ACM), and China Computer Federation (CCF).

**Ching-Nung Yang** received the B.S. degree and the M.S. degree, both from Department of Telecommunication Engineering at National Chiao Tung University. He received Ph.D. degree in Electrical Engineering from National Cheng Kung University. He is presently a professor in the Department of Computer Science and Information Engineering at National Dong Hwa University and is also an IEEE senior member. He has published a number of journal and conference papers in the areas of information security, multimedia security, and coding theory. He is the guest editor of a special issue on “Visual Cryptography Scheme” for Communication of CCISA, and a coauthor of a series of articles on “Image Secret Sharing” for the Encyclopedia of Multimedia. He is the coeditor of the book “Visual Cryptography and Secret Image Sharing” published by CRC Press/Taylor & Francis. He serves as a technical reviewer for over 30 major computer science journals in the areas of his expertise and serves as editorial boards of some journals. Also, he has served as member of program committees of various international conferences committees. He is the recipient of the 2000, 2006, 2010, and 2012 Fine Advising Award in the Thesis of Master of Science awarded by Institute of Information and Computer Machinery.