



Data hiding method for AMBTC compressed images

Yung-Yao Chen¹ · Chih-Hsien Hsia² · Sin-Ye Jhong¹ · Hsin-Ju Lin¹

Received: 26 February 2018 / Accepted: 16 September 2018 / Published online: 24 September 2018
© Springer-Verlag GmbH Germany, part of Springer Nature 2018

Abstract

With the current growth of multimedia applications, a substantial amount of visual data, including images and videos, is transmitted daily, making information forensics and security essential. Thus, protecting secret data from attackers is critical. In this work, we propose a AMBTC-based data-hiding method, in which secret data are concealed in individual image blocks that are compressed using absolute moment block truncation coding technique. By placing high or low mean values at the corner of each block, the secret data are then extracted reversibly and the original host image can be recovered at the decoding end. In addition, the quality of stego images is maintained, which protects the embedded visual data from the suspicion of attackers. From the experimental results, we show that the proposed method outperforms other state-of-the-art methods.

Keywords Information security for image data · Absolute moment block truncation coding (AMBTC) · Reversible data hiding

1 Introduction

Due to the fast development of network-related technologies, visual data communication through the Internet becomes prevalent. Consequently, to prevent data security problems, e.g. illegal interception and duplication, through the Internet, information forensics is essential. In this paper, we present a data-hiding method for absolute moment block truncation coding compressed images.

In the past few years, many image-based data-hiding techniques have been proposed to protect hidden data and guarantee secrecy of the visual data. On the basis of their applications, most of them can be classified into two categories: (1) steganography/encryption, e.g. Li et al. (2015b), Suri and Vijay (2018) and Shaheen et al. (2018), and (2) and watermarking, e.g. Han et al. (2016), Zhou et al. (2017), Zeng et al. (2017) and Martino and Sessa (2018). In the method of Zhou et al. (2017), steganography is achieved by

scattering the hidden data in each pixel, and the secret message has no connection to the stego image. The purpose of steganography is to protect the secret data only, not the cover image. Therefore, recovering the host image is not necessary for this type of applications.

The second category of application is called watermarking. In Bhowmik and Oakes (2016), the hidden data provide additional information about the host image. Although embedding data produces distortion inevitably, it is desired that the embedding distortion is controlled within a negligible level. Preserving image quality of the host images through the procedure of watermarking is essential for some applications, e.g. artwork preservation. Therefore, reversible data hiding schemes provide a solution to this problem because the embedding distortion can be totally removed, and the host image is recovered.

Moreover, embedding data in a compressed image is highly desirable. Most transmitted images are compressed first to reduce the file size and increase the network transmission efficiency. The commonly used compression methods typically are divided into four groups: lossy compression, lossless compression, predictive compression, and transform compression.

In lossy compression methods, gradual changes of color are maintained and sudden changes of color are removed. Although lossy compression introduces some distortion, it can achieve high compression rates. Therefore, lossy compression

✉ Yung-Yao Chen
yungyaochen@ntut.edu.tw

¹ Graduate Institute of Automation Technology, National Taipei University of Technology, No. 1, Sec. 3, Zhongxiao E. Rd. Daan Dist., Taipei 106, Taiwan, ROC

² Department of Computer Science and Information Engineering, National Ilan University, No. 1, Sec. 1, Shennong Rd., Yilan City 260, Yilan County, Taiwan, ROC

schemes are commonly used in streaming and Internet multimedia applications. In lossless compression methods, the quality of images or videos is not degraded. Moreover, after decoding the stego images, the original images can be recovered. Lossless compression has several applications for static image communication, such as scanning bank transactions. In addition, Portable Network Graphics (PNG) is a famous lossless graphics file format commonly used on the Internet. However, in lossless compression, because additional keys are required for recovering original images, achieving a high compression rate is difficult. Compared by the decoding time, lossless compression usually takes more time than lossy compression. Predictive compression involves high correlation between local space and time limit in image signals. Typically, in this technique, neighboring pixels are used to predict the value of current pixel and the error of predictive compression is quantized. Predictive compression is widely used in pictures with motion, digital television and videophone. Transform compression transforms the image data described in the spatial domain into the frequency domain through orthogonal transformation, such as Discrete Fourier Transform (DFT), Discrete Wavelet Transform (DWT), Discrete Cosine Transform (DCT), and so on. In this work, the proposed data hiding method is built based on the block truncation coding (BTC) format proposed in Delp and Mitchell (1979), which is a lossy compression scheme.

This paper presents a reversible AMBTC-based data hiding scheme. Different from other reversible data hiding method, the proposed method not only maintains high image quality, but also has the property of blind decoding. That is, the original host images can be recovered without additional information. In Sect. 2, we briefly introduce BTC method and review some previous reversible data hiding schemes. In Sect. 3, the proposed method is introduced. In Sect. 4, the proposed method is compared with other existing methods. Finally, Sect. 5 concludes this paper.

2 Related works

2.1 Block truncation coding (BTC)

This subsection presents the concept of BTC compression. In BTC, the input grayscale image is first divided into non-overlapped $n \times n$ blocks. Subsequently, the mean value (μ) and standard deviation (σ) of each block are calculated. For the purpose of compression, each block is stored only as one bitmap and two corresponding quantization levels. For the case of $n = 4$, the quantization levels A and B are obtained by

$$A = \mu - \sigma \times \sqrt{\frac{q}{m-q}}, \quad (1)$$

and

$$B = \mu + \sigma \times \sqrt{\frac{q}{m-q}}, \quad (2)$$

where $m = 16$ is the number of pixels in this block, and q indicates the number of pixels whose pixel value is larger than μ . To create the approximation, the bitmap is defined as follows. When a pixel value is smaller than the mean value of the block, it was set to "0" at that position; and otherwise, it was set to "1". Accordingly, each bit value "0" of bitmap is set as A , and each bit value "1" of bitmap is set as B when approximating the original (i.e. uncompressed) input image. Later, authors in Lema and Mitchell (1984) proposed an improving method of BTC to preserve high image performance, which is called AMBTC.

2.2 Reversible data hiding methods

Data-hiding methods usually can be divided into two types: (1) irreversible methods, e.g. Coltuc (2011), Zareian and Tohidypour (2013), and Qin et al. (2014), and (2) reversible methods, e.g. Fallahpour et al. (2011), Zhang et al. (2013), Coatrieux et al. (2013), and Wang et al. (2013). For the methods in the irreversible category, the hidden data are embedded into original host images to generate stego images. Although the secret data can be extracted, the host image cannot be recovered from the stego image. By contrast, reversible data-hiding schemes can not only extract the hidden data but also recover the original host images.

Typically, irreversible data-hiding methods outperform reversible schemes in terms of embedding capacity, whereas reversible data-hiding schemes have higher image quality. The reversible data-hiding schemes usually require additional information to restore the original images. Therefore, the compression ratio of the reversible schemes is often lower than that of the irreversible schemes. Moreover, reversible schemes are more difficult to design than irreversible schemes.

Several reversible data-hiding methods are proposed for AMBTC images. Lin et al. (2015) computed the mean value and variation of each block in AMBTC images to determine whether the current block can embed data. Next, the scheme was used to create four situations in individual blocks to embed data. Four data-hiding strategies have been proposed to manage the four cases specifically in the embeddable AMBTC blocks. Later, the method of Lin et al. (2015) is improved by the authors in Lin et al. (2017). For the improved data-hiding version, it can embed more secret data by combining a bitmap with the secret data. However, if the secret data are 0 or 1 consecutively, more pixels are required to store these positions. Although the latter method increases the embedding capacities of the former, it reduces the image performance as a compromise.

On the other hands, histogram shifting is also a popular scheme to achieve reversible data-hiding. Ni et al. (2006) proposed an embedding method, which is done by modifying this scheme and shifted the peak/zero points of the histogram. However, this histogram-shifting scheme suffers from the overflow and the underflow problems. One of the major concerns associated with this type of methods is that the peak/zero points must be embedded along with the data while the data are embedded. However, the histogram-shifting schemes is not suitable to be directly applied in AMBTC images. Lo et al. (2014) proposed a histogram-shifting reversible data-hiding method to divide the difference between the high and low mean values in the AMBTC-compressed block into three groups. In this technique, after selecting the group, histogram shifting is performed to embed the hidden data by modifying the pairs of peak and zero points. Kim et al. (2016) further proposed a simple histogram-shifting method for AMBTC-compressed images. In this method, the data are embedded in the position of a bitmap with value 1. If the secret bit is 0, the value 1 (bit 1) is maintained. By contrast, if the secret bit is 1, the bit value changes to 2.

Some other AMBTC reversible data-hiding methods exploit the parity property. Li et al. (2015a) proposed a hiding method by changing the high and low mean values of the AMBTC-compressed block. If the to-be-embedded secret data is 0, the two mean values will remain the same. If the to-be-embedded secret data is 1, the two mean values will change. To achieve a higher payload and good image quality, Huynh et al. (2018) proposed a novel least-significant-bit (LSB) algorithm based on the AMBTC method. The LSB-substitution method cannot be used to restore the original image. To realize reversible property, the minima-maxima preserving (MMP) algorithm embeds secret data into the AMBTC image blocks by maintaining the first high and the first low mean values for each block of the AMBTC compressed image; the secret data are then embedded in the remaining positions by replacing one or two LSBs.

Therefore, the total embedding capacity is considerably enhanced and the stego images still have high visual quality.

3 Proposed method

In this section, we propose a reversible data hiding method for AMBTC compressed image, as shown in Fig. 1. In Sect. 3.1, the overflow/underflow problem is solved by histogram modification. In Sect. 3.2, a block classification scheme is proposed by analyzing the corner values of each block. In Sect. 3.3, four embedding strategies are proposed according to different block types. Section 3.4 describes the embedding for type 4 blocks and the decoding procedure, in which the original host image can be restored.

3.1 Histogram modification

For an 8-bit grayscale image, the range of pixel value is from 0 to 255. To avoid overflow/underflow problem occurring after embedding secret data, a histogram modification step is performed using

$$I(x, y) = \begin{cases} 1, & \text{if } I(x, y) = 0 \\ 254, & \text{if } I(x, y) = 255 \\ I(x, y), & \text{otherwise} \end{cases} \quad (3)$$

where $I(x, y)$ represents the AMBTC image. After applying (3), for each block, the low mean (A) and high mean (B) values are replaced by

$$\begin{cases} A \rightarrow A' : A' = A - 1 \\ B \rightarrow B' : B' = B + 1 \end{cases} \quad (4)$$

where A' and B' represent the substituted low mean and the substituted high mean values, respectively. Equations (3) and (4) are applied to guarantee that the pixel values of the stego image are not duplicated after embedding data.

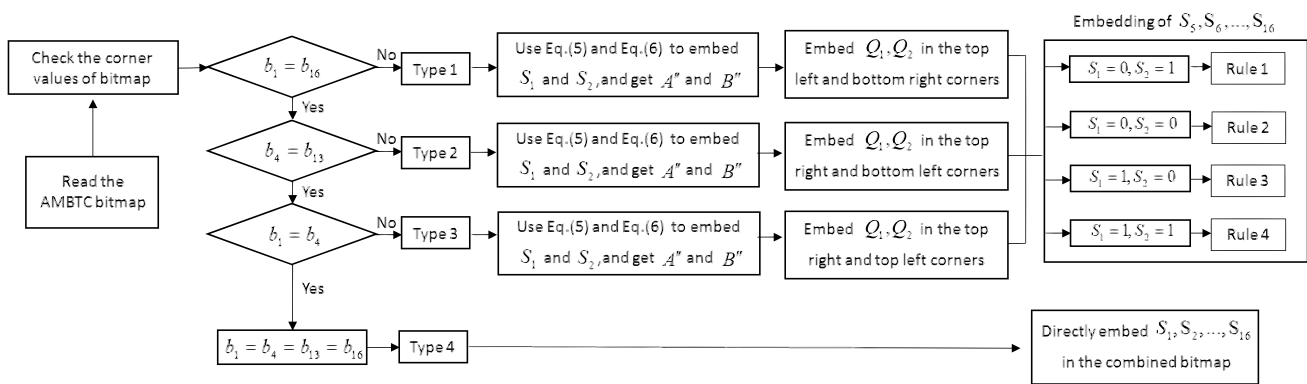


Fig. 1 Flowchart of the secret bits embedding stage

3.2 Block classification and corner storage

This subsection describes a block classification scheme in which all AMBTC blocks are classified into four types. The block classification is achieved by analyzing the corner values of each bitmap. As shown in Fig. 2, b_1, b_4, b_{13} and b_{16} represent the corner bit values. The corner storage situation is determined by orderly checking the property of corner value inequality as follows: If $b_1 \neq b_{16}$, this block belongs to type 1; otherwise, if $b_4 \neq b_{13}$, this block belongs to type 2. If $b_1 \neq b_4$, this block belongs to type 3; otherwise, the block belongs to type 4, that is, $b_1 \neq b_4 \neq b_{13} \neq b_{16}$. On the basis of the block classification results, we present a data-hiding strategy that enables the embedding 16-bit information (i.e., S_1, S_2, \dots, S_{16}) in each block. In this strategy, for block types 1, 2, and 3, S_1 and S_2 are embedded by modifying the values of A' and B' using

$$A'' = \begin{cases} A' - \left\lfloor \frac{S-Q}{2} \right\rfloor, & \text{if } Q \leq S \\ A' - \left\lfloor \frac{S-Q}{2} \right\rfloor, & \text{if } Q > S \text{ and } D + Q - S > T, \\ A' - \left\lfloor \frac{S-Q+T}{2} \right\rfloor, & \text{otherwise} \end{cases} \tag{5}$$

and

$$B'' = \begin{cases} B' + \left\lfloor \frac{S-Q}{2} \right\rfloor, & \text{if } Q \leq S \\ B' + \left\lfloor \frac{S-Q}{2} \right\rfloor, & \text{if } Q > S \text{ and } D + Q - S > T, \\ B' + \left\lfloor \frac{S-Q+T}{2} \right\rfloor, & \text{otherwise} \end{cases} \tag{6}$$

where D is the difference between A' and B' , T is the predefined threshold, and S is the binary-to-decimal result of the binary numbers S_1S_2 , that is,

$$S = S_1 \times 2^1 + S_2 \times 2^0, \tag{7}$$

The parameter Q is the key for the extraction stage (used to extract S_1S_2), which is defined as

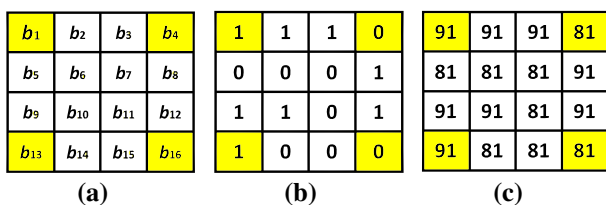


Fig. 2 Illustration of block classification. **a** Order of bit values. **b** Example of a type 1 bitmap. **c** Corresponding AMBTC block after histogram modification, where $A' = 81$ and $B' = 91$

$$Q = D \bmod T, \tag{8}$$

By applying decimal-to-binary conversion, Q is stored as the binary numbers Q_1Q_2 , and then these two binary numbers are placed in the corners of a block separately. Following the example shown in Fig. 2, if the secret bits are $S_1S_2 = 01$, after applying (5)–(8), four values are obtained, i.e. $A'' = 81, B'' = 91, Q_1 = 1$, and $Q_2 = 0$. To reinforce data protection, we propose a combined bitmap that stores two bits at each pixel position. The key binary numbers, Q_1 and Q_2 , are placed in the corners according to the block type. If the block belongs to type 1 (i.e., $b_1 \neq b_{16}$), Q_1 and Q_2 are placed at the top left and bottom right corner positions, respectively. The first four secret bits are placed as the second bit of the combined bitmap, as shown in Fig. 3. Finally, the corresponding pixel values of the stego block are computed according to the rules shown in Table 1.

3.3 Embed the remaining 12 bits for block types 1, 2 and 3

For block types 1, 2, and 3, we propose four hiding rules to embed the remaining 12 secret bits S_5, S_6, \dots, S_{16} according to the first two secret bits S_1 and S_2 .

- *Rule 1* If $S_1 = 0$ and $S_2 = 1$, the 12 secret bits S_5, S_6, \dots, S_{16} are placed separately as the second bit of

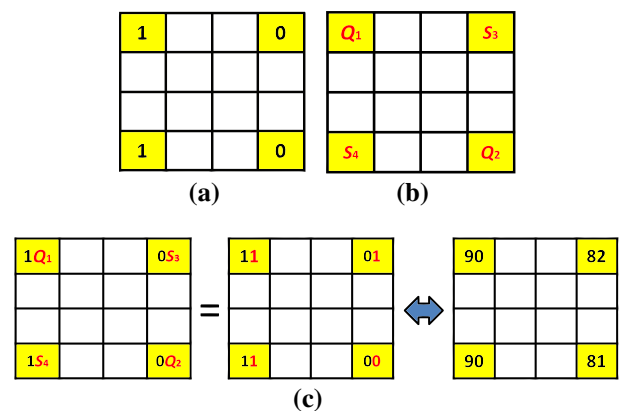


Fig. 3 Data hiding in corners (type 1 block). **a** Corner bit values of an AMBTC bitmap. **b** The embedding of secret bits $S_1S_2S_3S_4 = 0111$, following the example shown in Fig. 2. Based on the block type, the secret bits S_1 and S_2 are replaced by Q_1 and Q_2 using (5)–(8). **c** Corners of the resulting combined bitmap and stego block

Table 1 Rule of creating a stego block from the combined bitmap (types 1, 2, and 3)

Combined bitmap	00	01	10	11
Pixel value	A''	$A'' + 1$	B''	$B'' - 1$

the combined bitmap (except for the corners). For example, if the bit value in the AMBTC bitmap is 1 and the secret bit is 0, the bit value of that position is 10 in the corresponding combined bitmap. Thereafter, the corresponding pixel values of the stego block are computed according to the rules listed in Table 1. Following the example shown in Fig. 3, one example is provided as shown in Fig. 4.

- **Rule 2** If $S_1 = 0$ and $S_2 = 0$, the 12 secret bits S_5, S_6, \dots, S_{16} are placed separately as the first bit of the combined bitmap (except for the corners). For example, if the bit value in the AMBTC bitmap is 1 and the secret bit is 0, the bit value of that position is 01 in the corresponding combined bitmap. Thereafter, the corresponding pixel values of the stego block are computed according to the rules listed in Table 1.
- **Rule 3** If $S_1 = 1$ and $S_2 = 0$, the 12 secret bits S_5, S_6, \dots, S_{16} are placed separately as the second bit of the combined bitmap (except for the corners). Unlike Rules 1 and 2 which recall Table 1, in Rule 3, the corresponding pixel values (x_i) of the stego block are computed by

$$\text{if } b_i = 0, \begin{cases} x_i = A'' & (\text{in the case of } S_i = 0) \\ x_i = A'' + 1 & (\text{in the case of } S_i = 1) \end{cases} \tag{9}$$

and

$$\text{if } b_i = 1, \begin{cases} x_i = B'' & (\text{in the case of } S_i = 0) \\ x_i = B'' - 1 & (\text{in the case of } S_i = 1) \end{cases} \tag{10}$$

- **Rule 4:** If $S_1 = 1$ and $S_2 = 1$, the 12 secret bits S_5, S_6, \dots, S_{16} are placed separately as the first bit of the combined bitmap (except for the corners). Unlike Rules 1,

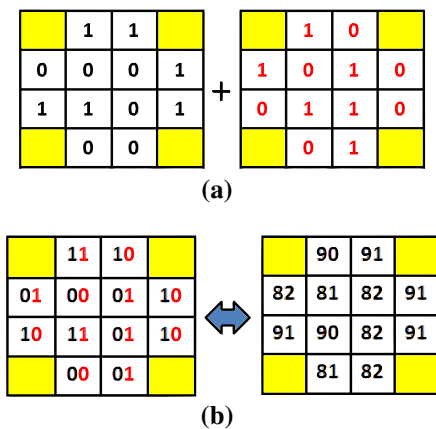


Fig. 4 Data hiding of the remaining 12 secret bits (case of $S_1 = 0$ and $S_2 = 1$). **a** Bit values of the AMBTC bitmap (black) and secret bits (red). **b** Resulting combined bitmap and the corresponding stego block

2, and 3, in Rule 4, the corresponding pixel values (x_i) of the stego block are computed by

$$\text{if } b_i = 0, \begin{cases} x_i = A'' + 1 & (\text{in the case of } S_i = 0) \\ x_i = A'' & (\text{in the case of } S_i = 1) \end{cases} \tag{11}$$

and

$$\text{if } b_i = 1, \begin{cases} x_i = B'' - 1 & (\text{in the case of } S_i = 0) \\ x_i = B'' & (\text{in the case of } S_i = 1) \end{cases} \tag{12}$$

3.4 Embedding for type 4 blocks and secret extraction

Because the four corner values are equal for a type 4 block, we can only predict the value of either A' or B' from the type 4 bitmap. Therefore, (5) and (6) are not performed, and the key binary numbers Q_1 and Q_2 are not used for the type 4 blocks. The 16 secret bits are thus all placed as the second bit to construct a combined bitmap. The rule in Table 2 is used to compute the pixel values of the corresponding stego block (Fig. 5).

In the following, we describe the procedure used to extract the hidden data and to restore the original AMBTC image. In the beginning, the first bit values of the corners (i.e. b_1, b_4, b_{13} and b_{16}) of the combined bitmap are analyzed to determine the block type. If this block is type 1, 2, or 3, the four bit values Q_1, Q_2, S_3 and S_4 can be extracted from the second bits of the corners, as shown in Fig. 3c. The quantization levels A'' and B'' are computed by using the rule in Table 1, and the first two secret bits S_1 and S_2 can be obtained using

$$S = D' \text{ mod } T, \tag{13}$$

where $D' = B'' - A''$. By transforming the decimal value S to its binary representation, the bit values of S_1 and S_2 are obtained. To extract the remaining 12 secret bits S_5, S_6, \dots, S_{16} , the bit values of each position in the combined bitmap are collected and the four embedding rules are recalled. If this block is type 4, the second bits of the combined bitmap are collected to extract the hidden data. Moreover, the decimal value Q can be obtained by transforming the binary numbers $Q_1 Q_2$ to its decimal representation. The difference (D) between A' and B' can be obtained by

$$D = D' - (S - Q). \tag{14}$$

Thereafter, the values of A' and B' can be computed by recalling (5) and (6) reversely. For example, if Q is less than

Table 2 Rule of creating a stego block from the combined bitmap (type 4)

Combined bitmap	00	01	10	11
Pixel value	A'	$A' + 1$	B'	$B' - 1$

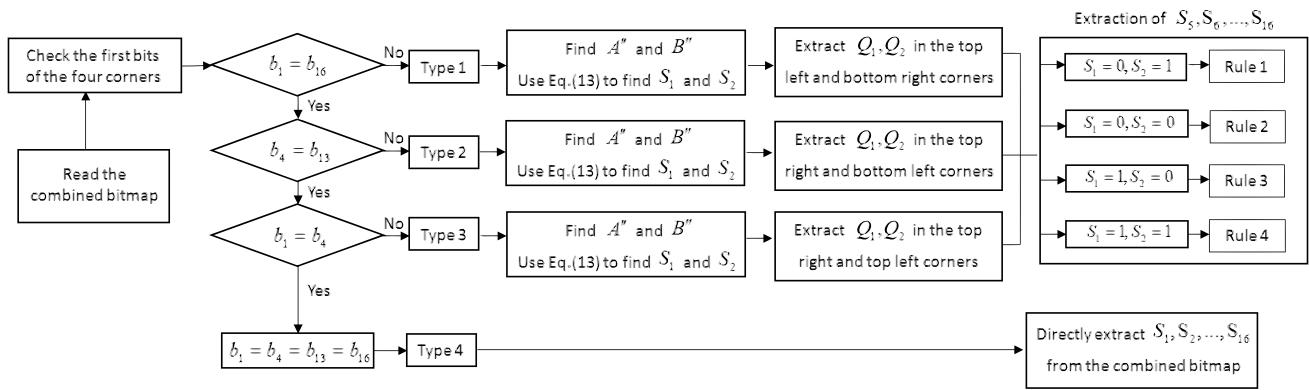


Fig. 5 Flowchart of the data extraction stage

or equal to S , $B' = B'' - \left\lfloor \frac{S-Q}{2} \right\rfloor$ and $A' = A'' + \left\lfloor \frac{S-Q}{2} \right\rfloor$. The recovery of the original AMBTC image requires obtaining the input AMBTC bitmap and the corresponding high and low mean values. To obtain the original bitmap, the four proposed hiding rules, which clearly define the values of the AMBTC bitmap, can be recalled. To obtain the corresponding high and low mean values, (4) is reversed to obtain A and B .

4 Experimental results

In this section, we list the comparison between the proposed method and the three methods we have introduced previously, e.g. the methods of Lin et al. (2015), Kim et al. (2016), and Huynh et al. (2018). The objective measures include PSNR, HPSNR, MSSIM and data capacity. Six standard 8-bit gray-scale images size of 512×512 are selected as the test images: Splash, Tiffany, Baboon, Lena, F-16 and House, as shown in Fig. 6. The secret data are all produced by the same pseudo-random generator.

4.1 Definition of image quality measures

- **Peak signal noise ratio (PSNR)** PSNR is the ratio of the maximum possible power (of a signal) to the detrimental noise power that affects its representation accuracy. As the PSNR value is higher, the quality performance of stego images is better. The PSNR value is defined by

$$10 \times \log_{10} \left(\frac{W \times H \times 255^2}{\sum_{W,H} \left[\sum_{m,n} g_{i+m,j+n} - h_{i+m,j+n} \right]^2} \right), \tag{15}$$

where (W, H) represents the image size. The variables g and h represent the input host image and the output stego AMBTC-compressed image, respectively.

- **Human visual peak signal noise ratio (HPSNR)** Different from PSNR, HPSNR takes the property of human visual system into account. That is, HPSNR must through a Gaussian low-pass filter to simulate the low-pass characteristics of human vision. The HPSNR value is defined by

$$10 \times \log_{10} \left(\frac{W \times H \times 255^2}{\sum_{W,H} \left[\sum_{m,n} q_{m,n} (g_{i+m,j+n} - h_{i+m,j+n}) \right]^2} \right), \tag{16}$$

where q represents the HVS coefficients.

- **Structural similarity (SSIM)** SSIM is an index of image evaluation, which evaluates the similarity of image by brightness, contrast, and structure, respectively. The SSIM value is defined by

$$\frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \tag{17}$$

where μ_x and μ_y represent the mean values of images X and Y , respectively. σ_x and σ_y represent the variance of images X and Y , respectively. σ_{xy} represents the covariance of images X and Y .

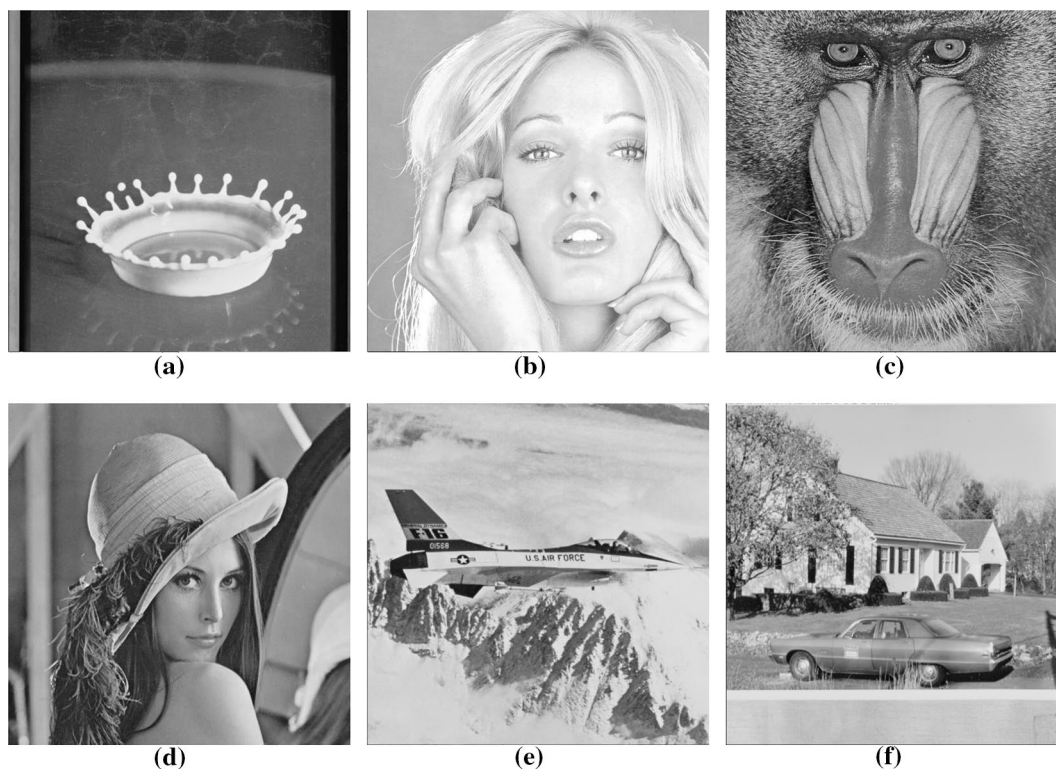


Fig. 6 The test images. a Splash. b Tiffany. c Baboon. d Lena. e F-16. f House

- *Mean structural similarity (MSSIM)* The MSSIM value is defined by averaging the SSIM values of M different window sizes. As the MSSIM value is higher, the quality performance of stego images is better. The MSSIM value can be computed by

$$\frac{1}{M} \sum_{j=1}^M \text{SSIM}(g_j, h_j), \tag{18}$$

where the variables g and h represent the input cover image and the output stego AMBTC-compressed image, respectively.

4.2 Performance comparison

The general property among different methods is discussed as follows. In the methods proposed in Lin et al. (2015) and Kim et al. (2016), not all AMBTC blocks are available for embedding data. According to their individual rules, the AMBTC blocks are classified into embeddable and nonembeddable blocks, and then different data-hiding approaches have been applied in Lin et al. (2015) and Kim et al. (2016). However, when extracting hidden data, the users in the extraction end must first distinguish the embeddable blocks.

Thus, it requires additional memory to save the locations of embeddable blocks, which results in considerable memory size because the AMBTC image is large. In the method proposed in Huynh et al. (2018), although the bitmap values are 0 and 1, up to four values appear in a stego block. Because the hidden data are extracted by comparing the stego block values with the high and low mean values of the original AMBTC block, these two values (or, the positions which corresponds to these two values) must be stored. This increases the complexity of data extraction. Compared with these three methods, the proposed method has the advantage of a stable embedding capacity (16 secret bits per block). Moreover, the memory size is fixed such that for each stego block, only high/low mean values and one combined bitmap is sufficient to extract hidden data. At the extraction end, no additional information is required to extract hidden data, and the original AMBTC image can be restored.

Figure 7 shows the overall comparison results among different methods, using the image quality measures defined in Sect. 4.1. As shown in Fig. 7, the proposed method outperforms the other methods with respect to averaged PSNR, averaged HPSNE, and averaged MSSIM. Furthermore, the average data capacity of the proposed method is also higher than those of the comparative methods. In the method in Kim et al. (2016), the secret bits are embedded by adding the third value in the stego block, which is computed by

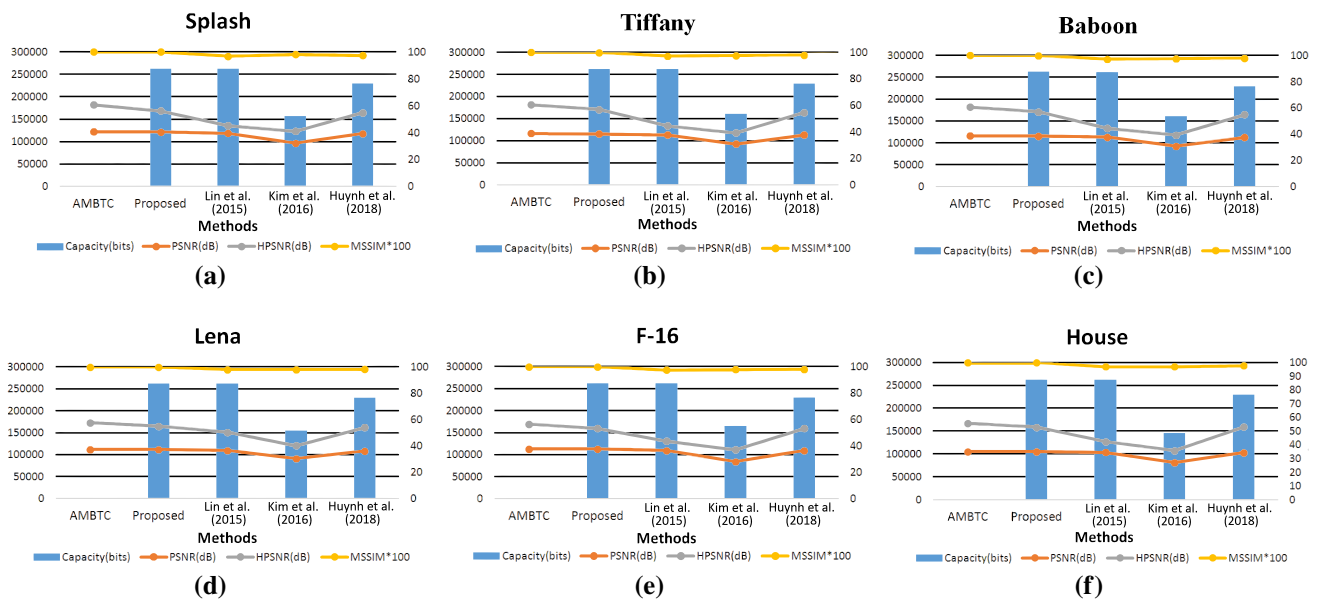


Fig. 7 Overall comparison results among different methods



Fig. 8 Visual comparison. **a** The original grayscale image. **b** The AMBTC image. **c** The stego AMBTC image produced by the proposed method

averaging the high and low mean values. However, if the two mean values are too far from each other, the third value results in unnatural noise in a stego block and degrades the image quality. In the method in Huynh et al. (2018), the secret bits are embedded by applying the least significant bit (LSB) substitution to modify the stego block values. However, if these two mean values are too close, the modified stego block value might be equal to the mean value. The duplicated problem might cause an error in the decoded results. In the proposed method, because the histogram modification step is implemented, the duplicated problem of a stego block is avoided. In addition, four hiding rules are separately applied according to the block types, enabling embedding of more secret bits while preventing degradation of the image quality.

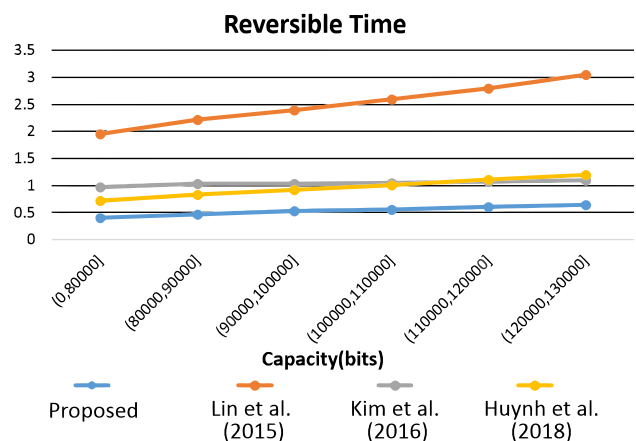


Fig. 9 Comparison of the reversible time (in units of seconds)

Figure 8 shows the visual comparison of the original grayscale image, the AMBTC compressed image, and the resulting stego image using the proposed method. As shown in the right side of Fig. 8a–c (the enlarged versions of the red rectangles), the detail and edge differences between the AMBTC image and stego image are almost negligible, which indicates the superiority of the proposed method. Figure 9 shows the comparison of the reversible time used to extract the hidden data and restore the original AMBTC image, in which the horizontal axis indicates the size of the embedded data. On an average, the method in Lin et al. (2015) requires the most time for data extraction. In this method, the system distinguishes whether the current processing block is an embeddable block, which requires collecting and analyzing all pixel values in the stego block. Furthermore, if this block is embeddable, the pixel values are hierarchically cross-validated to extract the hidden data. Therefore, compared with other methods, this method requires more time at the data extraction end. Because the embedding rules are concise and straightforward, the proposed method requires the least data extraction time among all methods.

5 Conclusion

This paper presents a data-hiding method based on the AMBTC compression technique. The proposed method has three advantages. First, this is a reversible data-hiding scheme that does not require additional memory to achieve the reversible property. Second, this paper presents a histogram modification scheme, which prevents possible decoding errors resulting from the duplication of pixel values. Third, we design four cases by classifying the bitmap storage condition. Therefore, a large amount of secret data can be embedded, and image distortion can be considerably reduced by slight adjustment of the pixel values. Because of the proliferation of Internet, the security of visual data has become more and more crucial. In the future, we plan to extend this work to color image domain, and thus this work can provide a solution for color hardcopy copyright protection, authentication, and other forensic applications.

Acknowledgements This work was supported in part by the Ministry of Science and Technology (MOST 106-2221-E-027-063-).

References

- Bhowmik D, Oakes M (2016) Visual attention-based image watermarking. *IEEE Access* 4:80028018
- Coatrieux G, Pan W, Cuppens-Boulahia N, Cuppens F, Roux C (2013) Reversible watermarking based on invariant image classification and histogram shifting. *IEEE Trans Inf Forensics Secur* 8:111120
- Coltuc D (2011) Improved embedding for predictionbased reversible watermarking. *IEEE Trans Inf Forensics Secur* 6:873882
- Delp E, Mitchell O (1979) Image compression using block truncation coding. *IEEE Trans Commun* 27:13351342
- Fallahpour M, Megias D, Ghanbari M (2011) Reversible and high-capacity data hiding in medical images. *IET Image Process* 5:190197
- Han J, Zhao X, Qiu C (2016) A digital image watermarking method based on host image analysis and genetic algorithm. *J Ambient Intell Humaniz Comput* 7:3745
- Huynh N, Bharanitharan K, Chang C, Liu Y (2018) Minima-maxima preserving data hiding algorithm for absolute block truncation coding compressed images. *Multimed Tools Appl* 77:57675783
- Kim C, Shin D, Leng L, Yang C (2016) Lossless data hiding for absolute moment block truncation coding. *J Real Time Process* 14:114
- Lema M, Mitchell O (1984) Absolute moment block truncation coding and its application to color images. *IEEE Trans Commun* 32:11481157
- Li F, Bharanitharan K, Chang C, Mao Q (2015a) Bistretch reversible data hiding algorithm for absolute moment truncation coding compressed images. *Multimed Tools Appl* 75:1615316171
- Li X, Zhang G, Zhang X (2015b) Image encryption algorithm with compound chaotic maps. *J Ambient Intell Humaniz Comput* 6:563570
- Lin C, Liu X, Tai W, Yuan S (2015) A novel reversible data hiding scheme based on ambtc compression technique. *Multimed Tools Appl* 74:38233842
- Lin C, Liu X, Tang J, Luo B (2017) Improved reversible data hiding scheme based on ambtc compression technique. In: *Proceedings of the international conference on Industrial IoT Technologies and Applications*, pp 111–118
- Lo C, Hu Y, Chen W, Wu C (2014) Reversible data hiding scheme for btc-compressed images based on histogram shifting. *Int J Secur Appl* 8:301314
- Martino F, Sessa S (2018) Fragile watermarking tamper detection via bilinear fuzzy relation equations. *J Ambient Intell Humaniz Comput*. <https://doi.org/10.1007/s12652-018-0806-3>
- Ni Z, Shi Y, Ansari N, Su W (2006) Reversible data hiding. *IEEE Trans Circ Syst Video Technol* 16:354362
- Qin C, Chang C, Chiu Y (2014) A novel joint datahiding and compression scheme based on SMVQ and image inpainting. *IEEE Trans Image Process* 23:969978
- Shaheen AM, Sheltami TR, Al-Kharoubi TM, Shakshuki E (2018) Digital image encryption techniques for wireless sensor networks using image transformation methods: DCT and DWT. *J Ambient Intell Humaniz Comput*. <https://doi.org/10.1007/s12652-018-0850-z>
- Suri S, Vijay R (2018) A synchronous intertwining logistic map-DNA approach for color image encryption. *J Ambient Intell Humaniz Comput*. <https://doi.org/10.1007/s12652-018-0825-0>
- Wang S, Li C, Kuo W (2013) Reversible data hiding based on two-dimensional prediction errors. *IET Image Process* 7:805816
- Zareian M, Tohidypour H (2013) Robust quantization index modulation-based approach for image watermarking. *IET Image Process* 7:432441
- Zeng L, Ren W, Chen Y, Lei M (2017) LMDGW: a novel matrix based dynamic graph watermark. *J Ambient Intell Humaniz Comput*. <https://doi.org/10.1007/s12652-017-0657-3>
- Zhang W, Hu X, Li X, Yu N (2013) Recursive histogram modification: establishing equivalency between reversible data hiding and lossless data compression. *IEEE Trans Image Process* 22:27752785
- Zhou Z, Wu M, Yang C, Sun X, Pan Z (2017) Coverless image steganography using histograms of oriented gradients-based hashing algorithm. *J Internet Technol* 18:1177–1184

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.