International Journal of Innovative
Computing, Information and Control
Volume **12**, Number **2**, April **2016**

ICIC International ©2016 ISSN 1349-4198
pp. 395–405

A BTC-COMPRESSED DOMAIN INFORMATION HIDING METHOD BASED ON HISTOGRAM MODIFICATION AND VISUAL CRYPTOGRAPHY

Hang-Yu Fan and Zhe-Ming Lu*

School of Aeronautics and Astronautics
Zhejiang University
No. 866, Yuhangtang Road, Hangzhou 310058, P. R. China
*Corresponding author: zheminglu@zju.edu.cn

Received September 2015; revised January 2016

Abstract. *This paper presents an information hiding method in the BTC-compressed domain based on histogram modification and visual cryptography. By using the visual cryptography algorithm, the secret image is divided into several transparencies, which are called shared images. Then, one shared image is embedded into the BTC-compressed data through histogram modification which is a reversible information hiding algorithm. When there is any need to recover the secret image, the shared image which formerly embedded into the BTC-compressed data can be extracted. Stacking the embedded shared image and the shared image we have already known, the secret image can be recovered. Experimental results demonstrate the feasibility of the proposed method, and the tampering location ability is quite good.*

Keywords: Block truncation coding, Histogram modification, Visual cryptography, Reversible information hiding

1. **Introduction.** With the development of multimedia technology, huge demands of information hiding appear to protect multimedia. The research of information hiding is booming since the 1990s. According to applications, the information hiding technology can be divided into three classes, which are secret information transmission, copyright protection (digital watermarking and digital fingerprinting) and content authentication.

During World War II, the most common way of hiding the secret information is to encrypt the original message by specific algorithms. This way may be useful, but it has obvious disadvantages. For example, if the algorithm is known by enemies, the whole system will be cracked. Later, digital signature was proposed to protect the multimedia. However, it also had a deadly disadvantage, which is easily tampered because of the known data structure. Fortunately, these problems could be solved by digital watermarks. As a kind of technology belonging to the information hiding area, secret information was embedded into the redundant space of multimedia. If the hiding information is used in military, the algorithm should pay more attention to safety. If the hiding information is used for copyright protection, the algorithm should focus on the robustness.

The information hiding algorithms can be broadly classified into two categories, which are irreversible information hiding and lossless information hiding. In irreversible information hiding algorithms, the carrier, which had been embedded information, cannot recover after the embedded information extraction. On the contrary, if we use lossless information hiding schemes, we can get the original carrier after the embedded information extraction. Compared with irreversible information hiding schemes, lossless information

hiding algorithms have more advantages, because the carrier can not only transmit secret information but also be recovered after information extraction.

Lossless information hiding algorithms can be divided into three classes, which are the spatial domain based [1], transform domains based and the compressed domain based. The spatial domain based and transform domains based hiding schemes are traditional, and they can be handled easily. On the other hand, the images are always transmitted by compressed formats. Therefore, finding appropriate information hiding methods is on the table [2-4]. The common image compression formats are JPEG, JPEG2000, BTC-compression and so on, while common lossless information hiding algorithms are difference expansion, histogram modification, prediction errors modification and so on. Combining image compression and information hiding algorithms flexibly can get a better result [5-10].

In recent years, quiet a lot scholars kept studying BTC-compression for enlarging embedded capacity. Some scholars proposed an edge-based quantization for compression of gray scale images using an Adaptive Block Truncation Coding technique (ABTC-EQ) [11] to improve the compression ratio. And many reversible data hiding schemes are proposed, too. Lin et al. proposed a reversible data hiding scheme based on AMBTC compression [12] recently.

In this paper, we proposed a new information hiding method that combines BTC-compression and histogram modification algorithm. Furthermore, we use the visual cryptography algorithm to handle the input secret information in order to enhance the security and tampering location ability [13,14]. Before we proposed this idea, we have researched the histogram modification for BTC-compression [2,8,21], and the simulation results showed that this hiding method is quite an effective scheme for embedding. As for visual cryptography, it is a safe and easily handled method to encrypt images, and we have researched earlier [20], too. Firstly, the information is embedded into the compression space, where is hard to detect. Secondly, even if the embedded information is extracted, the final information cannot be recovered without enough transparencies. Therefore, the proposed method is very safe, and it could be used in military fields.

The structure of this paper is organized as follows. In Section 2, the corresponding algorithms are described in detail. Then, the result and analysis are given in Section 3. In Section 4, we discuss the tampering location ability of our method. Finally, the conclusions are given in Section 5.

2. **Related Works and Proposed Method.**

2.1. **The BTC compression algorithm.** BTC compression is a lossy compression method first proposed by Delp and Mitchell in 1979 [15]. Later, Lema and Mitchell proposed the improved BTC compression method called absolute moment BTC (AMBTC) [16], which guarantees the same absolute moment after the compression for each block. The main idea of this algorithm is dividing the image into several blocks, and every block $\boldsymbol{x}$ contains $n \times n$ pixels (n typically equals 4). Assume x_i represents the grey value of each pixel in the block $\boldsymbol{x}$, m means the total number of pixels in each block, $\overline{\eta}$ means the average value over all pixels in $\boldsymbol{x}$, and $\overline{\alpha}$ means the first absolute central moment of $\boldsymbol{x}$, and we have

$$\overline{\eta} = \frac{1}{m} \sum_{i=1}^{m} x_i \tag{1}$$

$$\overline{\alpha} = \frac{1}{m} \sum_{i=1}^{m} |x_i - \overline{\eta}| \tag{2}$$

Here, $\overline{\eta}$ is used as the threshold, and p represents the amount of pixels whose values are not less than the threshold $\overline{\eta}$ in $\boldsymbol{x}$. And q represents the amount of pixels whose values are less than $\overline{\eta}$ in $\boldsymbol{x}$. Next, we use a to represent the average value over all the pixels whose values are not less than the threshold, and b represents the average value over all the pixels whose values are less than the threshold, that is

$$a = \frac{\sum_{for\ x_i \geq \overline{\eta}}^{m} x_i}{p} \tag{3}$$

$$b = \frac{\sum_{for\ x_i < \overline{\eta}}^{m} x_i}{q} \tag{4}$$

In order to label the pixel positions whether corresponding pixels are less or not less than $\overline{\eta}$, the compression algorithm uses a so-called bit-plane BM which has the same size with the block. If a pixel value is not less than $\overline{\eta}$, the corresponding value in BM is 1; otherwise the value is 0. Thus, the compressed data for each block can be replaced by (a, b, BM). The process for recovering each block can be performed as follows: if the value in BM table is 1, then the corresponding reconstructed pixel value is a; otherwise the reconstructed pixel value is b. Obviously, the decoding process will introduce distortion; thus, AMBTC is a lossy compression scheme. Figure 1 shows a concrete example for a 4×4 image block whose average value $\overline{\eta} = 119.06$. The pixels in the first row are all larger than $\overline{\eta}$, and thus the corresponding values in the BM block are all 1; while the pixels in the third row are all less than $\overline{\eta}$, and thus the corresponding values in the BM block are all 0. We can compute the average value of all pixels whose value is less than $\overline{\eta}$, i.e., $b = 118$. We can also compute the average value of all pixels whose value is not less than $\overline{\eta}$, i.e., $a = 120$. $BM1$ consists of the upper 8 binary numbers in the BM block with $(11110011)_2 = 243$, and $BM2$ consists of the rest 8 binary numbers with $(00001000)_2 = 8$. After compression, the original block can be described by 4 bytes: (a, b, $BM1$, $BM2$) = (120, 118, 243, 8).

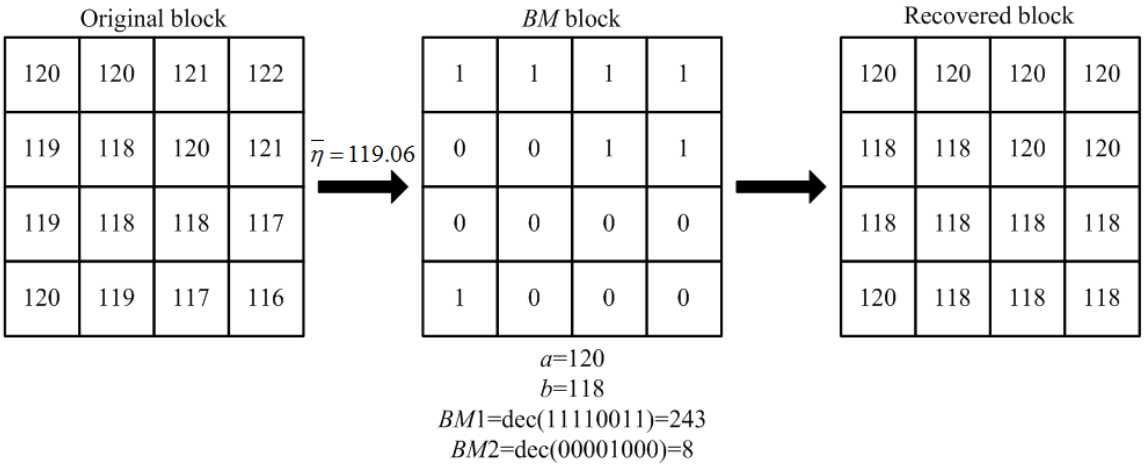


FIGURE 1. AMBTC compression scheme

2.2. **Related information hiding algorithms.**

2.2.1. *Lossless information hiding methods in the BTC-compressed domain.* As we all know, the BTC-compressed data for each block contain a, b and BM. Hong et al. proposed an information hiding method based on BTC-compressed data in 2008 [17]. The main idea is to switch the positions of a and b if necessary. For each block, if the secret bit which is going to be embedded is "1", then we switch a and b then reverse each binary bit in BM. If the secret bit which is going to be embedded is "0", we just skip the current compressed data of this block. If a equals b, we skip the compressed data of this block, too. This method did not really change the data in each block, so the decoding mechanism can recover the image completely. This method operates easily but wastes some space. Thus, there is an improved method proposed by Chen et al. in 2010 [18]. This method solved the space waste problem for the case that a equals b. For the case that a equals b, we clear BM block and embed $n \times n$ bits information in it. During the decoding process, when we detect the case that a equals b, we just extract all the $n \times n$ bits information from BM, then set 1 for every cell in BM block. This method can hide more information but lower the robustness.

2.2.2. *Histogram modification based algorithms.* The histogram modification algorithm [2,19] is an effective method to hide information. Assume x represents the pixel value and $H(x)$ represents the occurrence frequency of x. Find the maximum $H(x_{H\max})$ and minimum $H(x_{H\min})$. Shift all the bars between $x_{H\max}$ and $x_{H\min}$ ($x_{H\max}$ and $x_{H\min}$ are not included) to $x_{H\min}$, and the shift distance is 1. Assume $x_{H\max} < x_{H\min}$. The meaning of the shifting operation is just adding 1 to all the pixels whose value x_i is between $x_{H\max}$ and $x_{H\min}$. After the shifting operation, $H(x_{H\max} + 1) = 0$. Traverse all the pixels in order. If $x_i = x_{H\max}$, check the secret to be embedded. If it is 0, do not change x_i; otherwise, x_i add 1, then search the next pixel whose value equals $x_{H\max}$ and embed next secret data. So we can know the hiding capacity of this scheme is $H(x_{H\max})$. During the extracting process, just check the pixel values in order. If $x_i = x_{H\max}$, then the hiding data is 0; otherwise, if $x_i = x_{H\max} + 1$, then the hiding data is 1. After the extraction, all the pixel values can be recovered. If $x_{H\max} > x_{H\min}$, the shifting operation means that 1 should be subtracted from all the pixels with values $x_i \in (x_{H\max}, x_{H\min})$. The mechanism of the histogram modification algorithm is shown in Figure 2, where we take the case $x_{H\max} < x_{H\min}$ as an example.

2.3. **Visual cryptography algorithm.** Visual cryptography was first proposed by Noar and Shamir in 1995 [13]. It is a method for sharing a secret. Visual cryptography can be implemented in different ways [20]. The main idea of sharing a secret is that n sharers share one secret and k or more sharers can recover the secret. Such a scheme is called the (k, n) scheme. The visual cryptography algorithm can divide a secret binary image into n shared images. When k of n shared images are stacked together, the secret image can be recovered. Taking the $(2, 2)$ scheme as an example, one secret image is divided into two shares. The method to build the shared images can be explained in Figure 3. As we can see, when two transparencies are stacked, a white pixel in the secret image becomes a half white and half black block, and a black pixel in the secret image becomes a black block. In the (k, n) scheme, less than k shares stacked can only get half white and half black blocks, and there will not be completely black blocks. Obviously, a half white and half black block can be distinguished from a totally black block, and it can be seen as a white block.

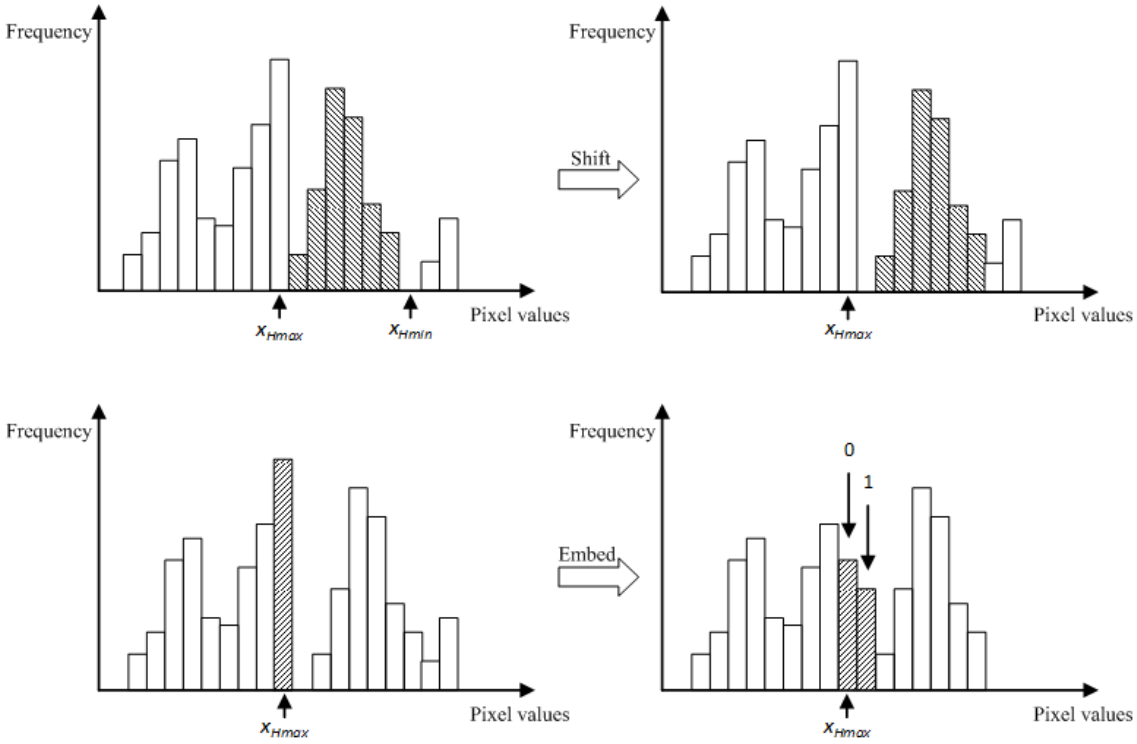


FIGURE 2. The histogram modification algorithm for the case $x_{H\max} < x_{H\min}$

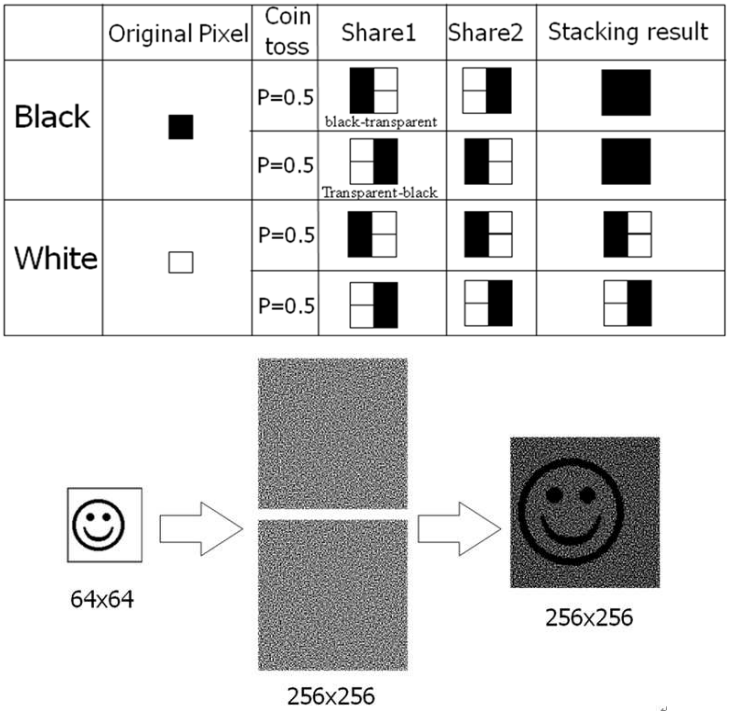


FIGURE 3. The $(2, 2)$ visual cryptography scheme

2.4. **The proposed hiding algorithm.** In this paper, we proposed a lossless information hiding scheme performed in the BTC compression domain, which combined the visual cryptography scheme. In our method, we adopt the histogram modification algorithm to

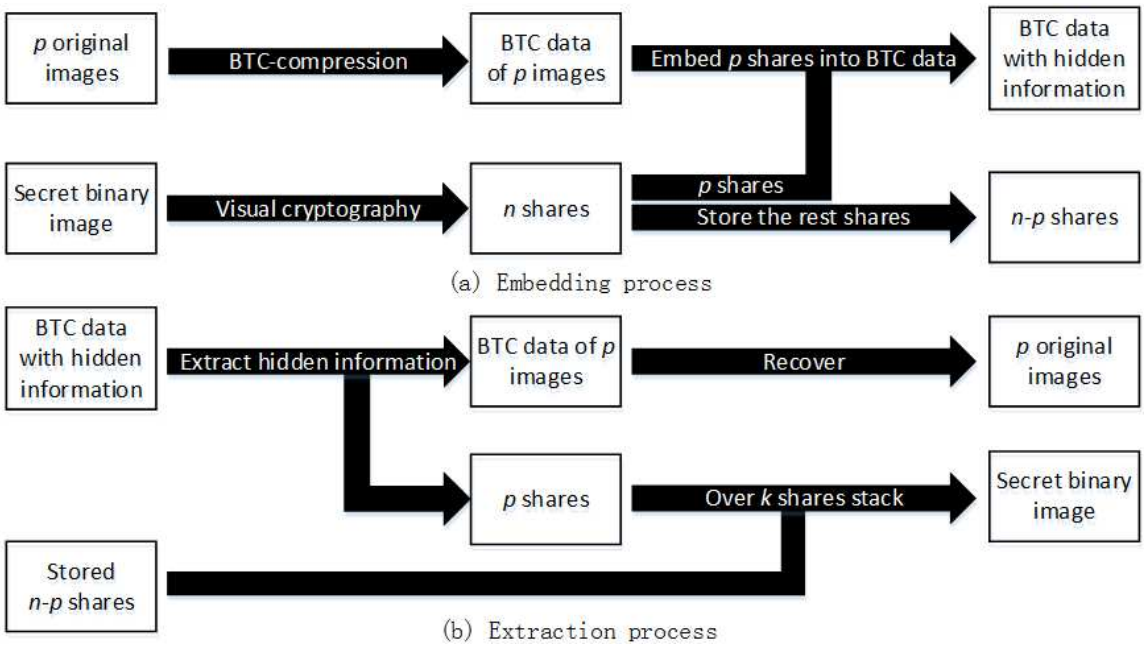


FIGURE 4. The proposed data embedding process (a) and extraction process (b)

hide information in the BTC-compressed data, where the high mean (i.e., a in 2.1) table and low mean (i.e., b in 2.1) table are thought of as two images. If we wanted to embed more information, we could use the algorithms in 2.2.1 after the using of histogram modification algorithm. On the other hand, we choose the (k, n) scheme in visual cryptography to enhance the security. We hide p share transparencies into p BTC-compressed data by the lossless hiding algorithm described in Sections 2.2.1 and 2.2.2 ($0 < p \leq n$). The process of the proposed method is shown in Figure 4. And the detail steps of the method are listed as follows.

Step 1. Use BTC-compression to compress the original images. Because one original pixel block can be compressed into 4 numbers (a, b, $BM1$, $BM2$), the compressed data of one image can be arranged as a_1, b_1, $BM1_1$, $BM2_1$, ..., a_i, b_i, $BM1_i$, $BM2_i$, ...

Step 2. Use the visual cryptography to seperate the secret binary image into n share transparencies. p shares will be embedded into the compressed data of p images, and the rest $n - p$ shares will be stored for recovering the secret image later.

Step 3. Embed p shares into the compressed data of p images. For each image, the compressed data can be split into 2 tables: high mean (a) table and low mean (b) table. And each share data will be embedded into 2 tables by using histogram modification. Then we use another hiding method which we mentioned in Section 2.2.1 to embed more data. After embedding, the BTC data become (a_1^*, b_1^*, $BM1_1^*$, $BM2_1^*$, ..., a_i^*, b_i^*, $BM1_i^*$, $BM2_i^*$, ...).

Step 4. When we got the BTC data with hidden information, we could extract hidden information. After the extraction process, the original BTC data can be recovered, because the hiding methods we used are lossless information hiding methods.

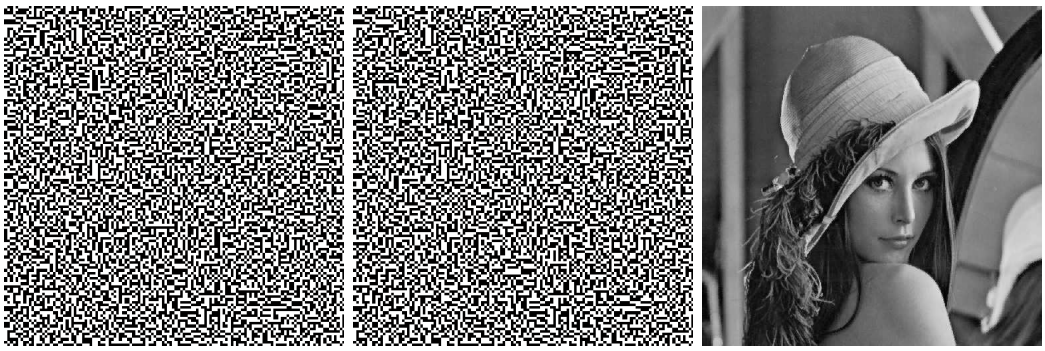
Step 5. When we got the data of p shares, we can recover the secret image by using these shares and the stored $n - p$ shares. In fact, we only need k (or more) shares to recover the secret image by stacking them.

The research of information hiding usually focuses on creating new algorithms, and the advantages are obvious. However, once the algorithm was cracked, all the secret information will be exposed. However, in our method, we combined the information hiding

and visual cryptography. If the hiding algorithm was cracked, the secret information is still safe for lacking of enough shares. What is more, the histogram modification is a lossless hiding method, which keeps the original data clean and usable.

(a) Original image (b) Secret image (c) BTC-compressed image

FIGURE 5. The original image, the secret image and the obtained BTC-compressed image without hidden information



(a) Share 1 (b) Share 2 (c) Image with embedded data

FIGURE 6. Two share transparencies and the image with embedded data

(a) Stacked secret image (b) Recovered secret image

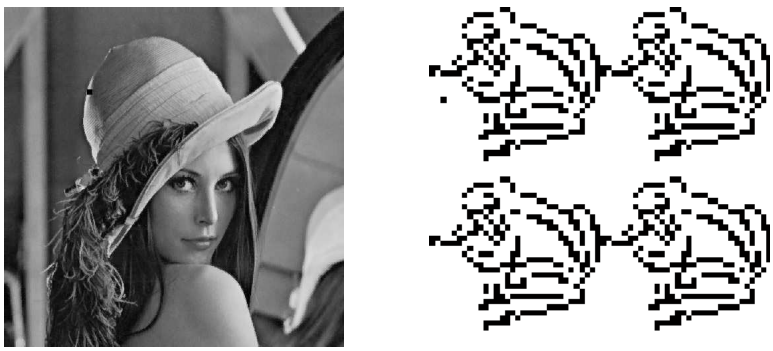
FIGURE 7. The stacked result and recovered secret image

3. **Results and Analysis.** We use Matlab to simulate and test our method. We adopt the Lena image as the original image, as shown in Figure 5(a). And we adopt the binary frog image as the secret image, as shown in Figure 5(b). The BTC-compressed image without hidden information is shown in Figure 5(c). The secret image is divided into two shares as shown in Figure 6(a) and Figure 6(b). We embed Share 1 into BTC-compressed data. We can then get the BTC-compressed image with embedded data as shown in Figure 6(c).

After extracting the data of Share 1 in the BTC-compressed data and recovering Share 1, we stack Share 1 and Share 2 to get the reconstructed secret image as shown in Figure 7(a). In this paper, we recover the secret image by counting the Hamming weight in each 2×2 block in the stacked image as shown in Figure 7(b). If the Hamming weight in a 2×2 block equals 2, the corresponding pixel color in the recovered secret image is white. If the Hamming weight in a 2×2 block equals 4, the corresponding pixel color is black. In order to enhance the robustness of our method, we can use more flexible criterion as follows: if the Hamming weight is not larger than 2, the corresponding pixel color is white; otherwise it is black.

4. **Tampering Location Ability Testing.** Now we start to discuss the robustness of the proposed method. The cover image size is 512×512, and the block size is 4×4. Each block is represented by 4 8-bit integers, which are the high mean (a), the low mean (b) and the bit-plane BM (2 8-bit integers). We adopt the tamper localization ability to evaluate our method in two different situations.

4.1. **Tampering specific blocks.** We use (x, y) to represent the position of block at the xth row and the yth column. We run 9 times in total, and 4 blocks data are tampered for each time. The tampering positions are $(32, 32)$ $(32, 33)$ $(33, 32)$ $(33, 33)$ for the first time, and we tamper the compressed data $(a_i^*, b_i^*, BM1_i^*, BM2_i^*)$ into 0 in these blocks, then we extract the embedded data and recover the secret image. The tampered BTC compressed image and recovered secret image are shown in Figure 8. We can find that this tampering does small effect, and one pixel is changed in the recovered secret image. The remaining results can be found in Figure 9. From Figure 10, we can see that there are 5 recovered secret images unchanged, and 4 recovered secret images are changed but still recognizable.



(a) Tampered BTC image (b) Recovered secret image

FIGURE 8. Tampering example and recovered secret image

FIGURE 9. The remaining results (TP means tampered position.)

FIGURE 10. The results of 20 8-bit compressed data tampered

4.2. **Random tampering.** In this test, we tamper the BTC-compressed data randomly. Here, we randomly change 20 8-bit compressed data and 50 8-bit compressed data for 5 times each. The results are shown in Figure 10 and Figure 11 respectively.

As shown in Figure 10, three recovered secret images are unchanged, while two recovered secret images are destroyed. In Figure 11, one recovered secret image is unchanged, one is changed but can be recognized, two are destroyed and one cannot be recovered.

The reason of above phenomena is that some important data, such as $x_{H\max}$, is changed. The change of these important data leads to complete mess of decoding, so we can explain why the recovered secret images are destroyed. There are still some data which means the resolution of Share 1, and once they are tampered, the recovery of secret images will fail. If the end point of the hiding data is tampered, the image cannot be recovered, too.

As we can see in Figure 11, the more data we tampered, the more possibility changed the important data. And we can say the proposed method has some robustness, but still cannot bear heavily attacking. Assume that we know the important data (such as

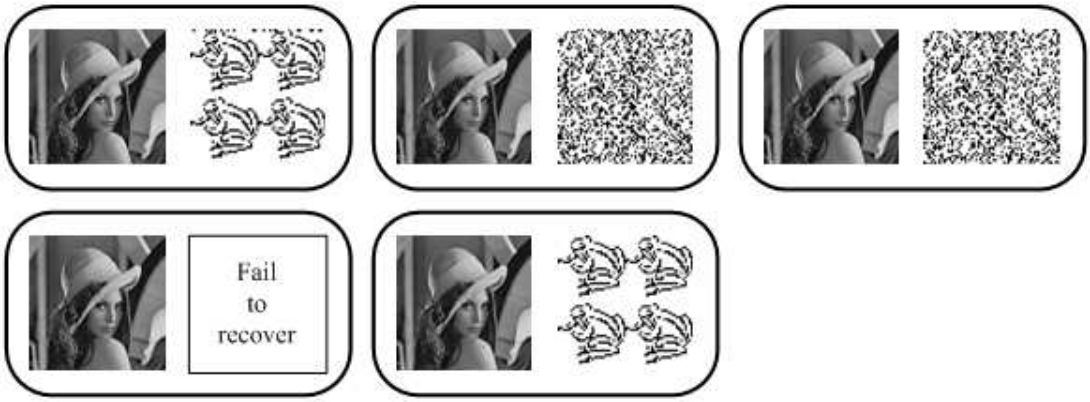


FIGURE 11. The results of 50 8-bit compressed data tampered

resolution of the secret image), and the hidden information is just the pixel values of Share 1, the robustness of the proposed method can be strengthened more.

5. **Conclusions.** In this paper, we have proposed a lossless information hiding method performed in the BTC-compressed domain, which combines visual cryptography. Embedding one transparency share into BTC-compressed data only changes a little visual effect. After the visual cryptography division, the security is improved. Even though the embedded share is intercepted, the secret image cannot be recovered for lacking of enough transparencies. It is confirmed that it is a feasible and stable way after our simulation. And the proposed method can bear some attacks and have good tampering location ability.

In addition, there is still much room to improve. There are many other lossless information hiding methods worth to be tried. If we choose different information hiding methods, the attacking results can be totally different. And visual cryptography can be improved to enhance the security. More works need to be done in the future.

Acknowledgment. This work is supported by the National Natural Scientific Foundation of China under Grant No. 61171150.

REFERENCES

[1] W.-M. Zheng, Z.-M. Lu and H. Burkhardt, Color image retrieval schemes using index histograms based on various spatial-domain vector quantizers, *International Journal of Innovative Computing, Information and Control*, vol.2, no.6, pp.1317-1326, 2006.

[2] Z.-F. Zhao, H. Luo, Z.-M. Lu and J.-S. Pan, Reversible data hiding based on multilevel histogram modification and sequential recovery, *AEU-International Journal of Electronics and Communications*, vol.65, no.10, pp.814-826, 2011.

[3] J.-X. Wang and Z.-M. Lu, A path optional lossless data hiding scheme based on VQ joint neighboring coding, *Information Sciences*, vol.179, no.19, pp.3332-3348, 2009.

[4] Z.-M. Lu, J.-X. Wang and B.-B. Liu, An improved lossless data hiding scheme based on image VQ-index residual value coding, *Journal of Systems and Software*, vol.82, no.6, pp.1016-1024, 2009.

[5] Y.-J. Hu, K. Wang and Z.-M. Lu, An improved VLC-based lossless data hiding scheme for JPEG images, *Journal of Systems and Software*, vol.86, no.8, pp.2166-2173, 2013.

[6] K. Wang, Z.-M. Lu and Y.-J. Hu, A high capacity lossless data hiding scheme for JPEG images, *Journal of Systems and Software*, vol.86, no.7, pp.1965-1975, 2013.

[7] Z.-M. Lu, J.-S. Pan and S.-H. Sun, An efficient BTC image compression algorithm with vector quantization, *Chinese Journal of Electronics*, vol.9, no.4, pp.453-456, 2000.

[8] Y. Zhang, S.-Z. Guo, Z.-M. Lu and H. Luo, Reversible data hiding for BTC-compressed images based on lossless coding of mean tables, *IEICE Trans. Communications*, vol.96, no.2, pp.624-631, 2013.

[9] F.-X. Yu, H. Luo and Z.-M. Lu, Colour image retrieval using pattern co-occurrence matrices based on BTC and VQ, *Electronics Letters*, vol.47, no.2, pp.100-101, 2011.

[10] W. Sun, Z.-M. Lu, Y.-C. Wen, F.-X. Yu and R.-J. Shen, High performance reversible data hiding for block truncation coding compressed images, *Signal, Image and Video Processing*, vol.7, no.2, pp.297-306, 2013.

[11] J. Mathews and M. S. Nair, Adaptive block truncation coding technique using edge-based quantization approach, *Computers & Electrical Engineering*, vol.43, pp.169-179, 2015.

[12] C.-C. Lin, X.-L. Liu, W.-L. Tai and S.-M. Yuan, A novel reversible data hiding scheme based on AMBTC compression technique, *Multimedia Tools and Applications*, vol.74, no.11, pp.3823-3842, 2013.

[13] N. Moni and A. Shamir, Visual cryptography, *Advances in Cryptology – EUROCRYPT'94*, 1995.

[14] Z.-F. Zhao, K.-Y. Chau and Z.-M. Lu, High capacity data hiding in reversible secret sharing, *International Journal of Innovative Computing, Information and Control*, vol.7, no.11, pp.6411-6421, 2011.

[15] E. J. Delp and O. R. Mitchell, Image compression using block truncation coding, *IEEE Trans. Communications*, vol.27, no.9, pp.1335-1342, 1979.

[16] M. D. Lema and O. R. Mitchell, Absolute moment block truncation coding and its application to color images, *IEEE Trans. Communications*, vol.32, no.10, pp.1148-1157, 1984.

[17] W. Hong, T.-S. Chen and C.-W. Shiu, Lossless steganography for AMBTC-compressed images, *Congress on Image and Signal Processing*, vol.2, pp.13-17, 2008.

[18] J. Chen, W. Hong, T.-S. Chen and C.-W. Shiu, Steganography for BTC compressed images using no distortion technique, *Imaging Science Journal*, vol.58, no.4, pp.177-185, 2010.

[19] Z.-C. Ni, Y.-Q. Shi, N. Ansari and W. Su, Reversible data hiding, *IEEE Trans. Circuits and Systems for Video Technology*, vol.16, no.3, pp.354-362, 2006.

[20] H. Luo, F.-X. Yu, S.-C. Chu and Z.-M. Lu, Hiding multiple watermarks in transparencies of visual cryptography, *International Journal of Innovative Computing, Information and Control*, vol.5, no.7, pp.1875-1881, 2009.

[21] H. Luo, F.-X. Yu, Z.-L. Huang, H. Chen and Z.-M. Lu, Reversible data hiding based on hybrid prediction and interleaving histogram modification with single seed pixel recovery, *Signal Image & Video Processing*, vol.8, no.5, pp.813-818, 2014.