Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

MBE, 17(1): 160–178
DOI: 10.3934/mbe.2020009
Received: 13 May 2019
Accepted: 29 August 2019
Published: 26 September 2019

Research article

High capacity data hiding with absolute moment block truncation coding image based on interpolation

Cheonshik Kim[1*], **Dongkyoo Shin**[1*] **and Ching-Nung Yang**[2]

[1] Department of Computer Engineering, Sejong University, 209, Neungdong-ro, Gwangjin-gu, Seoul 143-747, South Korea

[2] Department of Computer Science and Information Engineering, National Dong Hwa University, No. 1, Sec. 2, Da Hsueh Rd., Shoufeng, Hualien 97401, Taiwan

* **Correspondence:** Email: mipsan@paran.com; shindk@sejong.ac.kr.

Abstract: Data hiding is a way of hiding secret data on cover-media and it is used for a variety of applications. An important of the data hiding is to conceal the data in a secret way without loss of cover-media. Until now, continuous research on absolute moment block truncation coding based data hiding methods have improved a performance on data concealment and image quality. However, the current absolute moment block truncation coding based data hiding technology has a limitation in deriving a method that significantly surpasses existing performance. In this paper, we propose a new method to overcome this problem. To do this, first the original image is transformed to the cover image using absolute moment block truncation coding and is expanded the image using neighbor average interpolation algorithm. The proposed three data hiding methods are based on the generated cover image. The first method is to directly replace the pixel value, which is a component of each block, with the same secret value. The second method is to replace the pixels to match the secret bits only for the extended pixels in each block of the cover image. The third method is to apply Hamming code to each block to minimize the number of replacement pixels for data hiding. Experimental results show that our method is superior in terms of efficiency compared to traditional absolute moment block truncation coding based data hiding methods.

Keywords: Data Hiding; AMBTC; BTC; Neighbor Mean Interpolation; Hamming code

1. Introduction

With the advent of smartphones and the emergence of social network services (SNS), many users often create a lot of multimedia data and upload it to SNS via the Internet to exchange it with other users. However, the more the openness of the Internet is high, the more it may cause shared

multimedia to exposed to the risk of tampering from illegal users. The data hiding (DH) [1, 2, 3] function may be used for various purposes such as intellectual property protection, content authentication, annotation etc and confidential communication using images and video. The most important aspect of data hiding is that it hides the existence of data, whereas watermarking technology must be able to protect the information against any attack, even if it may be revealed that there exists copyright information. DH method begins from identifying a cover image's redundant bits and the DH process creates a stego image by replacing these redundant bits with the secret bits, where the cover image means an image that will contain secret data.

Meanwhile, reversible data hiding (RDH) [4, 5] is a technique that reconstruct the cover image after extracting a secret message embedded in the stego image. This technique is useful in medical and military fields in which the defects of the cover image are not acceptable. However, there is a drawback in that an additional payload is usually required to restore the original image. For the DH based on the spatial domain, the researchers focus mainly on finding an efficient way to embed secret bits into the least significant bits of the pixel (LSB) [6, 7] by flipping the LSB. The advantage of DH based on spatial domain is that it provides high embedding capacity (EC) and good image quality. On the other hand, it has the disadvantage of being vulnerable to the attacks such as image compression, image transformation and salt & pepper.

Meanwhile, frequency-based data concealment is particularly strong against a variety of image attack methods. In the compression methods based on frequency domain, there are discrete cosine transform (DCT) [8], discrete wavelet transform (DWT) [9], singular value decomposition (SVD) [10], and so on. Block truncation coding (BTC) [11] is an image compression coding having moderate bit-rate due to its low computation, and the absolute moment BTC (AMBTC) [12] proposed by Lema and Mitchell uses the most widely in one of BTC. AMBTC is composed of bitmap and two quantization levels and it represents as $trio(a, b, BM)$ where a and b are two quantization levels and BM is a bitmap. Until now, various DH methods based on AMBTC have been introduced. First, Chuang and Chang [13] proposed a method to hide secret bits into each BM of the cover image using direct bitmap substitution (DBS). The merit of this method may control the image quality by adjusting threshold T. However, for a certain threshold ($b - a \leq T$), it may not be able to provide satisfactory EC. This is because the number of blocks for the threshold value T varies depending on the characteristics of the cover image.

Chen et al. [14] introduced a lossless DH using two quantization levels of the *trio*. This method is called the Order of Two Quantization Levels (OTQL) method and it may embed 1-bit per block. For example, to store '1'-bit, the order of a and b is replaced as follows; $trio(b, a, BM)$. This method does not change the coefficients of the two quantization levels, so that the original cover image can be completely restored without any harmful effect on the image. Hong et al. [15] proposed hybrid method that is a combined lossless and lossy hiding with considerations on acceptable distortions. The secret data was first embedded lossless in the complex image blocks and any amounts exceeding would be lossy embedded based on optimal bitmap replacement on the smooth image blocks. It may maintain the image quality as this method.

Ou and Sun [16] introduced a method of DH using DBS and OTQL together to improve data hiding performance. Here, they adjusted the errors happened from embedding procedure by their optimal algorithm. Bai and Chang [17] proposed a method for applying matrix encoding to both the quantization levels and the BM on smooth blocks as well as using OTQL for complex blocks. In

2017, Huang et al. [18] also proposed a scheme to embed data using DBS, OTQL, and modification of pixels difference (hidden bits = log_2T: derived from difference expansion method) on the two quantization levels. Here, they adjust the difference between the quantization levels to preserve reasonable visual quality. W Hong [19] proposed a DH using pixel pair matching (PPM) [20], DBS, and OTQL, where the PPM is applied to the quantization levels to increase EC. In [21], Jung and Yoo introduced new simple algorithm of a scale-up neighborhood mean interpolation (NMI) with low complexity. An expanded cover image is generated from the original image using NMI.

In this paper, we intend to improve EC performance by using expanded AMBTC cover image and three DH methods. The cover image is that combines the AMBTC generated from the original image and the original image expanded by NMI. The larger the difference between the two quantization levels in AMBTC, the larger the noise is when the DBS algorithm is applied. Therefore, our method can generate less noise for the cover image because it provides enough blocks to store more data in the same threshold condition. For the same reason, since data concealment performance has been greatly improved, various applications based on the proposed method are expected to be developed.

The main contributions of this paper include: (i) To ensure high EC and image quality, we devised a way to expand AMBTC using the NMI algorithm. The quality of the enlarged AMBTC showed the same quality as that of the original AMBTC. (ii) In our proposed methods, to restore the original cover image is possible by handling the expanded pixels of the cover image. (iii) High quality of cover image can be maintained after hiding data while minimizing distortion of the bitmap using Hamming code. (iv) We present a method of increasing data hiding performance up to two times or more while maintaining image quality.

The remainder of this paper is organized as follows. Section 2 briefly describes the AMBTC compression technique, the extension of an image usng neighbor mean interpolation (NMI), the optimal quantization level adjustment, and the Hamming code method. In Section 3, we introduce three our proposed DH methods. In Section 4, we compare and analyze the experimental results using the conventional DH and the proposed DH. We will prove that the proposed method is better performance than that of previous methods. Finally, this paper concludes in Section 5.

2. Preliminaries

2.1. AMBTC compression technique

The AMBTC [12] is a method to compress the original grayscale image by preserving two quantization levels and a bitmap of blocks, and it obtains better reconstructed image quality than BTC [11]. In addition, this technique is effective in saving the bandwidth of a portable device as well as it is also easier to implement than BTC because it does not have square root and square calculations.

For transforming the original image into AMBTC, it is divided into non-overlapped blocks of size $m \times m$. The bitmap and quantization levels are obtained through the following computations. The total number of pixels in the block is $n = m \times m$. The mean value (μ) of each block is calculated as Eq 2.1. The pixel x_i denotes the i^{th} pixel in the block.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.1}$$

Every pixels x in the $m \times m$ block are quantized into a bitmap, $\mathbb{b}_i$ (0 or 1). That is, if the corresponding pixel x is greater than or equal to the μ, then the replaced bitmap pixel is '1' or '0'. The pixels in each block is classified into two group consisting of '1' and '0'. p_n and $n - p_n$ mean the number of pixels in the group of '1' and '0', respectively. The mean a and b for two group are used to preserve the quantization levels of the '0' and '1' group. The two quantization levels are computed by Eqs 2.2 and 2.3:

$$a = \frac{1}{p_n} \sum_{x_i < \mu} x_i \tag{2.2}$$

$$b = \frac{1}{n - p_n} \sum_{x_i \geq \mu} x_i \tag{2.3}$$

where a and b are also used to reconstructed AMBTC.

$$\mathbb{b}_i = \begin{cases} 1, & \text{if } (x_i \geq \mu) \\ 0, & \text{otherwise} \end{cases} \tag{2.4}$$

$$x_i' = \begin{cases} a, & \text{if } (\mathbb{b}_i = 0) \\ b, & \text{otherwise} \end{cases} \tag{2.5}$$

The bitmap obtains simply computed by Eq 2.4. The encoded block is simply decoded by using the calculation of Eq 2.5, and x_i is reconstructed into grayscale pixels by the quantization level. The compressed code, *trio(a,b,BM)*, may be obtained by using Eqs 2.2–2.4.

165	175	176	188
175	185	193	166
156	106	100	123
147	167	178	143

(a) Natural image block

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

(b) AMBTC bitmap

176	176	176	176
176	176	176	176
129	129	129	129
129	176	176	129

(c) Reconstructed image block

Figure 1. An example of AMBTC.

Figure 1 shows an example of encoding and decoding a block of grayscale image. Here, assuming that (a) is a block of an grayscale image. In this block, the mean of block is 158. Applying Eq 2.4 to the (a) yields the bitmap block shown in (b). The two quantization levels ($a = 129, b = 176$) of the decoded block (c) are obtained using Eqs 2.2 and 2.3. The $trio(a, b, BM)$ denotes a pattern of compressed block. The restoration from *trio* to the block Figure 1(c) is simply doing by Eq 2.5.

2.2. Extension of an image using neighbor mean interpolation (NMI)

For NMI [21], the details of the scale-up procedure are introduced using examples. We here introduced a way how to extend a 2×2 sized block composed 4-pixel to a 3×3 block composed of 9-pixel. In Figure 2, the 4-pixel block (a) has four element pixels such as $(x_{i,j}, x_{i,j+2}, x_{i+2,j}, x_{i+2,j+2})$.

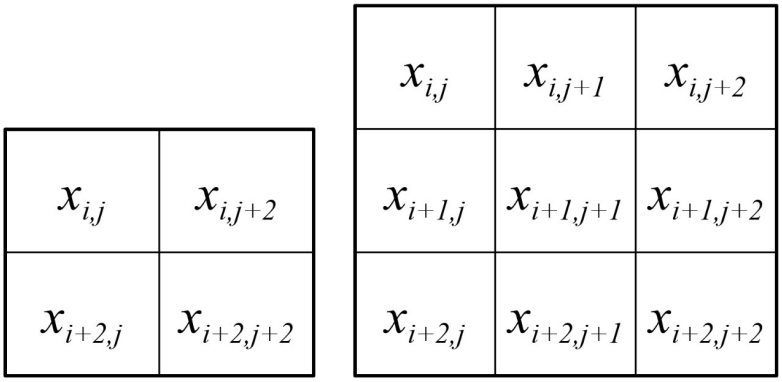

Figure 2. An example of an extension of the block.

The 9-pixel block (see Figure 2(b)) is composed of original 4-pixels of Figure 2(a) and extra five inserted pixels. Five inserted pixels $(x_{i,j+1}, x_{i+1,j}, x_{i+1,j+1}, x_{i+2,j+1}, x_{i+1,j+2})$ are determined from $(x_{i,j+1}, x_{i+2,j}, x_{i+2,j}, x_{i+2,j+2})$ via Eq 2.6:

$$\begin{cases} x_{i,j+1} = (x_{i,j} + x_{i,j+2})/2 \\ x_{i+1,j} = (x_{i,j} + x_{i+2,j})/2 \\ x_{i+1,j+1} = (x_{i,j} + x_{i,j+1} + x_{x+1,j})/3 \\ x_{i+2,j+1} = (x_{i+2,j} + x_{i+2,j+2})/2 \\ x_{i+1,j+2} = (x_{i+1,j+2} + x_{i+2,j+2})/2 \end{cases} \tag{2.6}$$

2.3. Optimal quantization level adjustment

W. Hong [19] proposed optimal quantization level adjustment to improve quality of stego image and it proved to make an image's quality improving some degree. In case that one pixel in Figure 1(b) is changed from '0' to '1', the difference between the two quantization levels become 47 ($b - a = 47$). On average, by doing about 50% of pixels per block may be flipped, it will have a bad effect on the quality of the image. Thus, it is necessity to adjust the two quantization levels to reduce the noises. The re-calculation of quantization levels is Eq 2.7, where n_0 and n_1 denote the number of the unchanged bits and changed bits, respectively. Re-calculated two quantization levels are a' and b'.

$$a' = \frac{a_i n_0 + b_i n_1}{n_0 + n_1},\ b' = \frac{a_i n_1 + b_i n_0}{n_0 + n_1} \tag{2.7}$$

2.4. Hamming Code

The Hamming code HC $(n, n-k)$ of $d_{min} = 3$ is a single error-correction linear block code, where k is the number of parity bits and $(n-k)$ is the number of data bits. The Hamming code is the code $C : \{0,1\}^4 \to \{0,1\}^7$ such that for every $v \in \{0,1\}^4$. We have $C(v) = vG$ (the product of the vector v and the maxtrix G), where $G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$. G is called the code generating matrix. Let $\mathbb{F}_2^n$ be n-bit binary column vectors $y = (y_1, \ldots, y_n)^T$. The codeword y is obtained from data word $x \in \mathbb{F}_2^{n-k}$ via $y = x \cdot G$. H is a $k \times n$ parity matrix, $H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$. For codeword y, the vector $S = Hy$ is called to the syndrome. Suppose that the codeword $\hat{y}$ has an error like a pattern $e = (y \oplus \hat{y})$. The position of the error can be gained using Eq 2.8.

$$\begin{cases} \hat{y} \cdot H^T = (e \oplus y) \cdot H^T = e \cdot H^T + y \cdot H^T \\ = e \cdot H^T + 0 = e \cdot H^T \end{cases} \tag{2.8}$$

For example, suppose that it happened that the fifth (from left) bit of codeword $\hat{y}$ has an error, i.e., $e = (e_1, e_2, \ldots, e_7) = (0000100)$. Using the Eq 2.8, we may find out the syndrome ($S = (101)$), which denotes the position of the error is 5^{th}. This may solve easily the error by flipping the fifth bit.

3. The Proposed Schemes

DBS is an efficient DH method, which is widely used in AMBTC for high EC without time complexity. The DBS adjusts the quality of the image by limiting the difference between the two quantization levels ($b - a = T$). In order for more EC, it is possible to increase the limit of the threshold T. Then, the quality of the image may deteriorate.

The proposed method adopts the expanded AMBTC based on NMI, so the EC is very high. Moreover, for a given threshold T, our proposed method provides higher image quality than existing methods. This is because more data can be embedded for the same T. Another reason is that an optimal adjustment technique for the quantization level is used to maintain good image quality. In this paper, we propose three Methods; 1) Method 1: For a DH technique based on interpolation to embed high capacity; 2) Method 2: For a RDH technique based on interpolation except for fixed blocks; and 3) Method 3: For a RDH technique based on interpolation (using Hamming coding).

In Figure 3, after the original image is converted to *trios*, we apply the proposed schemes to the *trios* to embed secret data. First, we transform the original image into the expanded original image and *trios* $(a, b, BM, mean)$, respectively ① ②. The bitmap is generated by comparing the mean values derived from the *trio* with the extended original image using equation 2.4 ③. The *trios* for the cover image are done by combining the generated BM and *trios*(a, b) ④. The stego *trios* are generated by applying methods 1, 2, and 3 to the *trios* ⑤. The stego *trios* are decoded ⑥ and compared with the expanded cover image ⑦. Lastly, if you remove the generated pixels by NMI from the expanded pixels, original AMBTC will be generated ⑧.

In this paper, we propose an improved DH based on AMBTC, and we adopt and use the cover

image derived from the Figure 3. However, since AMBTC is derived from the original image, the quality of the stego image is measured by comparing with the original image.

3.1. Embedding procedure (Method 1)

Figure 4a shows a simple diagram of the procedure of creating a cover image. For Method 1 (DBS), if the threshold T is satisfied for the first $trios(a, b, BM)$ list, it is possible to embed bits in the BM, otherwise moves to the next *trio*. For *trio* that meet the threshold T, it takes a secret bit equal to the size of the bitmap and replaces it directly in the bitmap BM (Figure 4b). Details of proposed Method 1 are as follows:

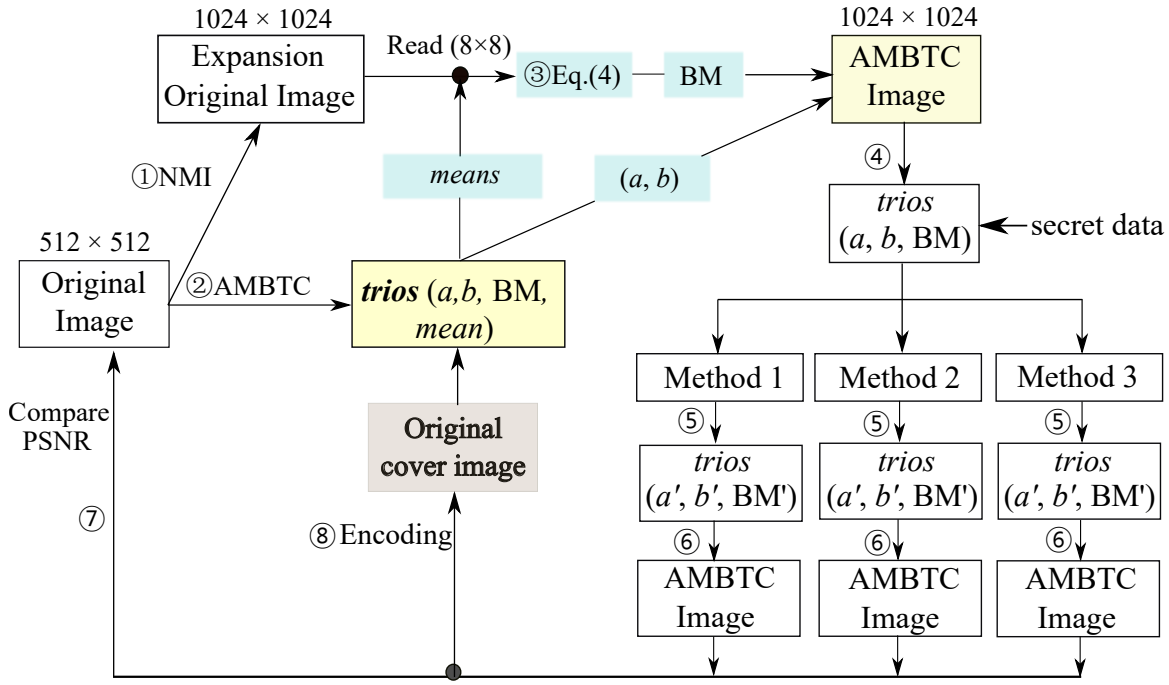

Figure 3. Diagram of the procedure to generate *trios*.

Input: AMBTC codes $trios(a, b, BM)$, secret data κ, block size $m \times m$.
Output: AMBTC stego code $trios(a, b, BM)$, threshold T.

Step 1. Decide threshold T as optimum value.

Step 2. Set B = $trio(a_i, b_i, BM)_{i=1}^{N}$ (Notes: assign bitmap).

Step 3. If $(b_i - a_i \leq T)$, obtain secret bits of $m \times m$ from κ and assign B_i' into BM. Re-calculate the quantization pair (a_i, b_i) to (a_i', b_i') by using Eq 2.7.

$$B_i' = \begin{cases} \kappa_1^{m \times m}, & \text{if } (b_i - a_i \leq T) \\ no\ action, & \text{otherwise} \end{cases} \tag{3.1}$$

Step 4. Steps 2 and 3 are repeated until every *trios* are scanned.

3.2. *Embedding procedure (Method 2)*

Method 2 embeds the secret bits only in the expanded pixels as shown in Figure 4c to reconstruct original cover image. After extracting the data, we can reconstruct the original cover image by removing the expanded pixels in every blocks.

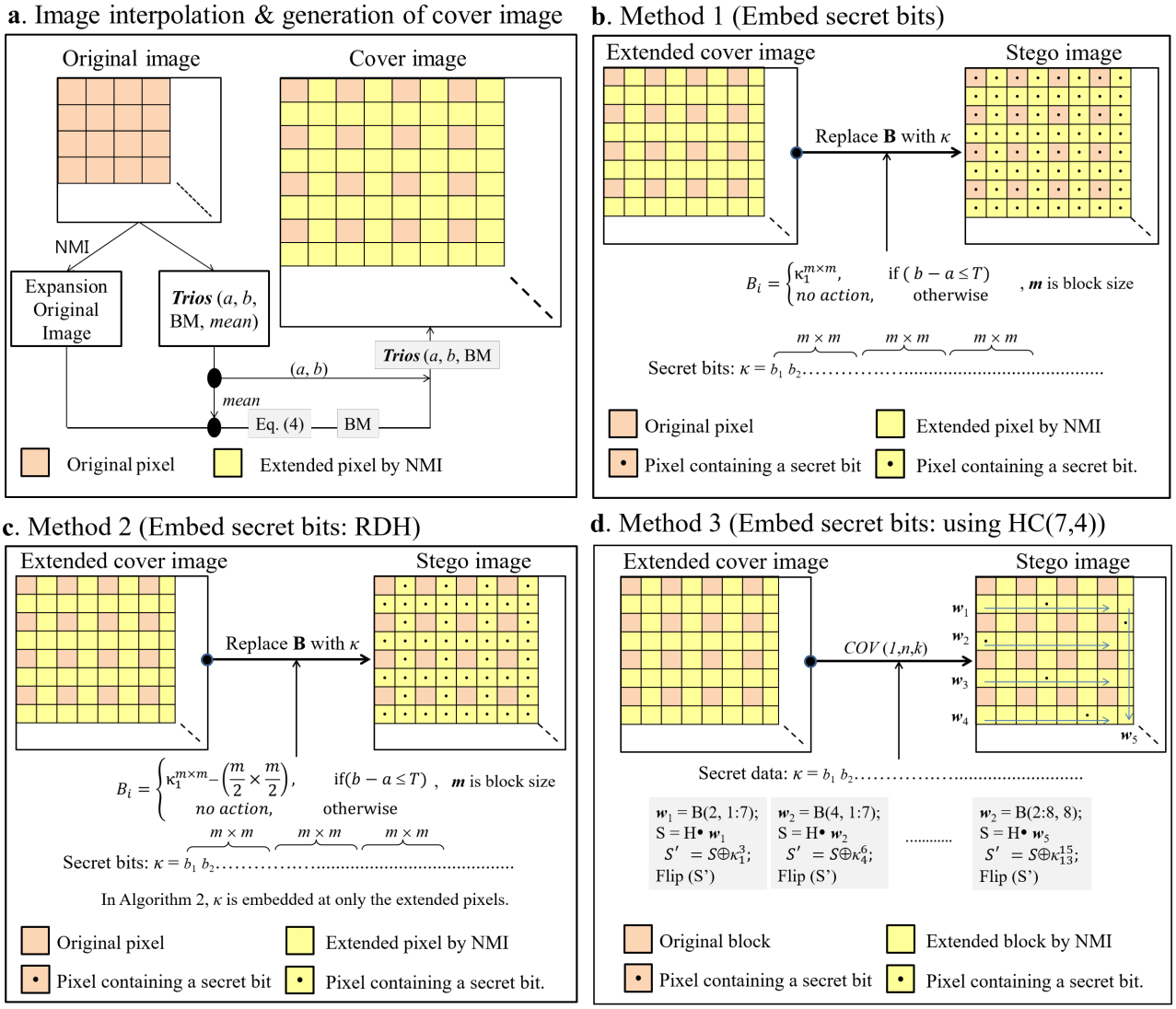

Figure 4. Diagrammatical representation of the proposed scheme.

Input: $trios(a, b, BM)$, secret data κ, block size $m \times m$.
Output: $trios(a, b, BM)$, threshold T.

Step 1. Used the same as threshold T of (Method 1).

Step 2. Set $B = trios(a, b, BM)_{i=1}^{N}$.

Step 3. If $(b - a \le T)$, fetch secret bits of $m \times m - (m/2 \times m/2)$ from κ, and replace B with the secret bits using Eq 3.2, otherwise, no action. Re-calculate the quantization pair (a_i, b_i) to (a_i', b_i') by

using Eq 2.7. Assign B_i' into BM (Notes: Except fixed pixels in B_i).

$$B = \begin{cases} \kappa_1^{m\times m} - \left(\frac{m}{2} \times \frac{m}{2}\right), & \text{if } (b_i - a_i \le T) \\ no\ action, & \text{otherwise} \end{cases} \tag{3.2}$$

(Note: Secret bits are embedded at only the extended pixels.)

Step 4. If $(b - a) \ge T$ and $\kappa_1^1 = 1$, replace $trios(a, b, BM)$ with $trios(b, a, BM')$.

Step 5. Steps 2–4 are repeated until every *trios* are scanned.

3.3. Embedding procedure (Method 3)

Method 3 applies the cover function $COV(1, 7, 3)$ to the bitmap BM of *trio*, and it may conceal 3 secret bits at each 7-pixels of BM. Let codeword w be 7-bits binary vector, the syndrome of the codeword w is $S = Hw^T$, and κ is secret bits. The distance between the syndrome S and secret bits κ is $S' = (S \oplus \kappa)$. Assuming that error pattern e is equal to S', it is possible to recover the codeword w by changing one bit of w. The Eq 3.3 is to explain such a concept as well as it can be able to extract 3-secret bits from the codeword w.

$$\begin{cases} w' \cdot H^T = (w \oplus e) \cdot H^T = w \cdot H^T \oplus e \cdot H^T \\ = S \oplus S' = \kappa \end{cases} \tag{3.3}$$

The embedding procedure is briefly described below.
Input: $trios(a, b, BM)$, secret data κ, block size $m \times m$.
Output: $trios(a, b, BM)$, threshold T.

Step 1. Used the same as threshold T of Method 1.

Step 2. Set $B = trios(a, b, BM)_{i=1}^N$. // Read a block and assign to B.

Step 3. If $(b - a \le T)$, repeat 5-times applying HC (7,4) to B. (Notes: In each block, the expanded all pixels are divided 7 pixels and assigned to w_1–w_5, and HC (7,4) applies to them.)

1. Read 7-bit from the B and assign to variable w;
2. Calculate syndrome, $S = Hw^T$;
3. $S' = S \oplus k_1^7$; Flip $w_{S'}$;
4. Replace 7-bits w with B and then assign B to BM.

Step 4. If $(b - a) \ge T$ and $\kappa_1^1 = 1$, Replace $trios(a, b, BM)$ with $trios(b, a, BM')$.

Step 5. Steps 2–4 are repeated until every *trios* are scanned.

4. Examples

The detailed procedure of our proposed DH explains using simple data. As shown in Figure 5a, $trio(103, 107, BM)$ and secret bit κ are given first. Since $b - a = 107 - 103 = 4 \le T$, the *trio* is classified as a smooth block. In Figure 5b, secret bits κ = '1100 1001 0110 1010' is embedded into BM using (Method 1), which may perform by replacing κ with B directly. B_1 appears the block after

embedding the secret bits. After then, the re-calculation for two quantization levels is performed, i.e., $a' = 105$ and $b' = 104$ for B_1. The MSEs between $trio(103, 107, B_1)$ and $trio(105, 104, B_1)$ are 10 and 4.3125, respectively, thus it seems that the re-calculation of two quantization levels is effective.

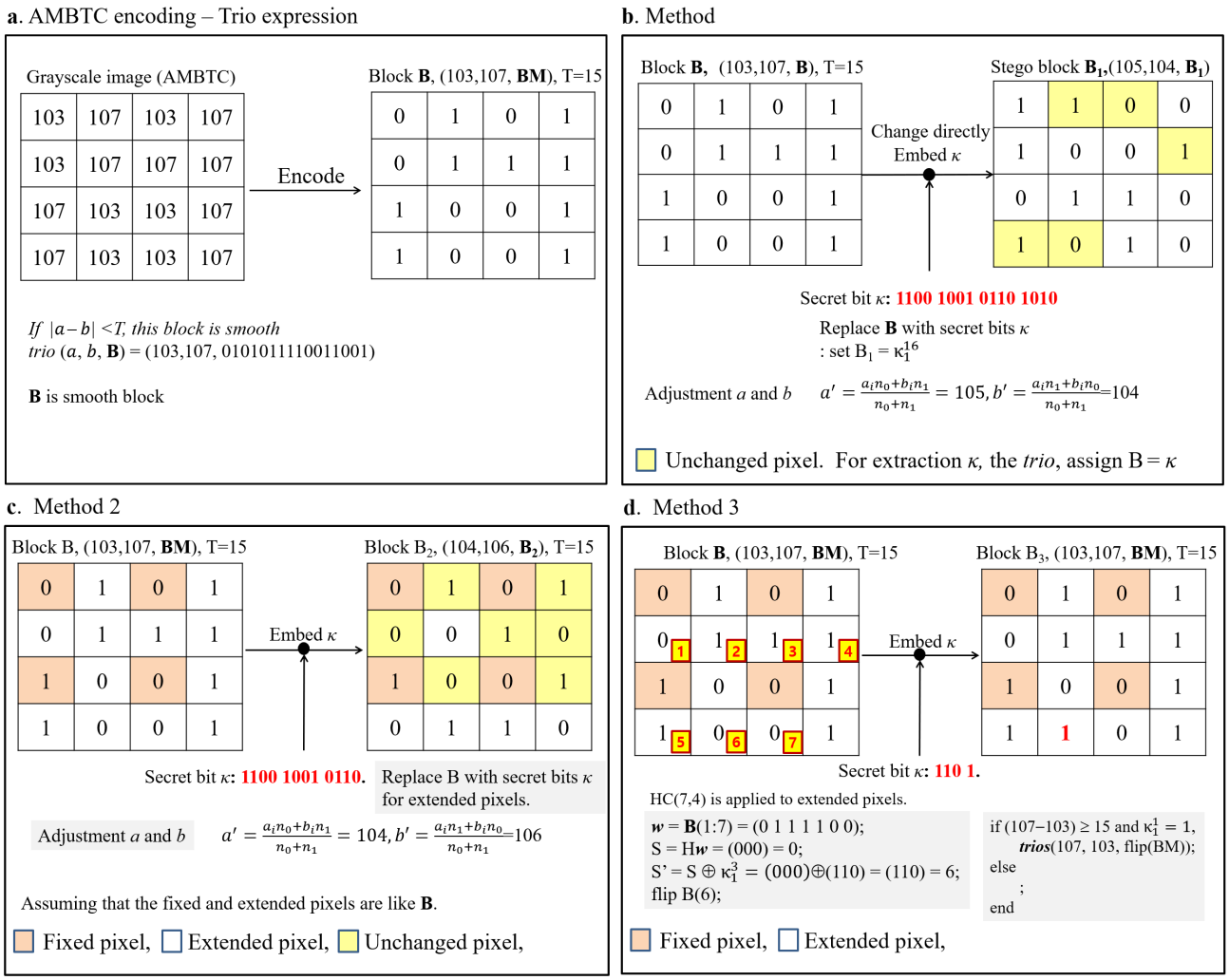

Figure 5. Examples of the proposed scheme.

Given the data and condition of the $trio(103, 107, BM)$, T=15, and secret bits κ = "1100 1001 0110" are to applied to Method 2 (see Figure 5c). The B_2 in Figure 5c is the block after embedding secret bits. Here, the number of modification pixels is 5 and the re-calculated two quantization levels for improving image quality become $(a', b') = (104, 106)$.

Figure 5d shows last example of DH using Method 3. Here, the secret bits κ = "110" are embedded into BM using the HC (7,4) function. In this example, we obtain 7 expanded pixels from each 4×4 sized BM according to the determined rules and assign this value to the variable w. Then, we calculate the syndrome $S = Hw^T = (000)$. Next, If we compute $S' = S \oplus \kappa = (000) \oplus (110) = (110)$, then we found out that there is an error of 6^{th} pixel in w. Through correcting this error, we may embed 3-secret bits in w. Finally, the value of w moves into the BM. By

repeating this procedure five times, 15 bits may be embedded in each block.

To embed 1-bit additionally, it is possible by using OTQL if the condition of two quantization levels is $(b - a) \geq T$. Here, since the secret bit κ = '1', we change the order of the two quantization values (b, a), i.e., $trios(b, a, BM')_{i=1}^{N}$.

5. Experiment and Comparison

For fairly evaluate analysis of the performance between existing schemes and our proposed scheme, we compare the performance through the experiments using nine numbers of standard 512×512 sized grayscale images (see Figure 6). These standard images are transformed into *trios* through the procedure of Figure 3. For existing schemes, we experiment using *trios* generated from original grayscale images directly. Through comparing the performance, we prove the superiority of our proposed scheme.

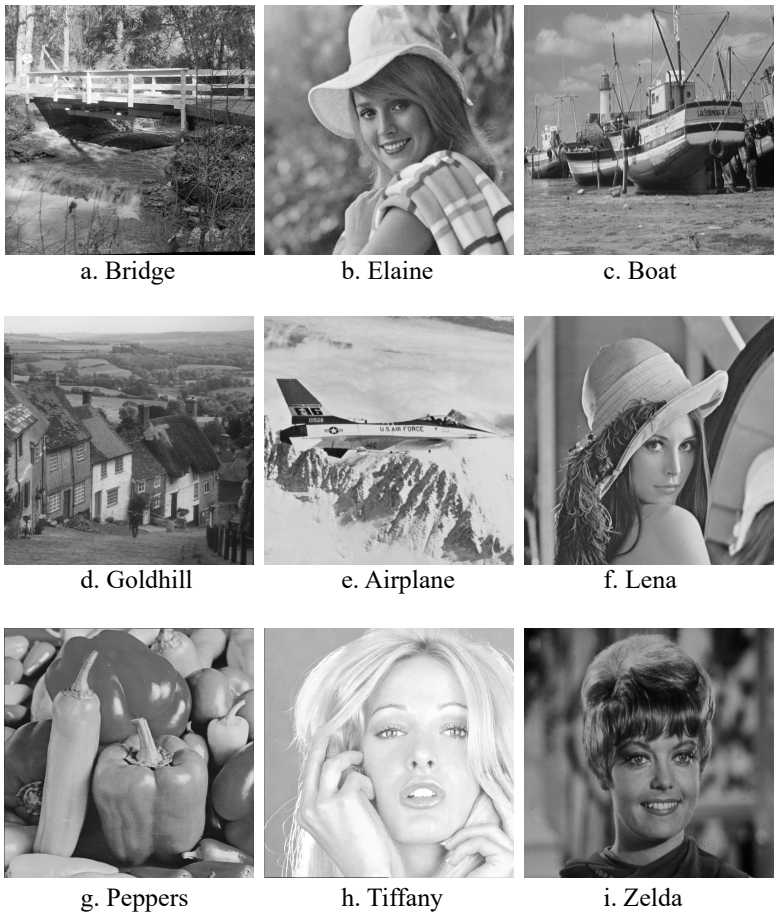

Figure 6. Nine test images for our experiments.

Figure 7 shows the types of images used in the experiment, i.e., (a) original "Lena" image, (b) expanded original image, (c) AMBTC obtained from original image, and (d) the cover image obtained from "the expanded original image" and "AMBTC image" based on the procedure of Figure 3. The

two type images, Figure 7(c,d) are used to experiment for existing schemes and our proposed scheme, respectively.

Figure 7. Original image, expansion original image, compressed image (AMBTC), and cover images: (a) original image, (b) Enlarged original image obtained from the original image by NMI, (c) AMBTC image from original image by AMBTC algorithm, and (d) cover image.

The secret bits used in the experiment are randomly generated binary number by the pseudo random number generator. An objective criteria for DH evaluation exploits measurement of the quality and the EC of stego images. Peak signal to noise ratio (PSNR) is used to measure the quality of an image and it is calculated using mean square error (MSE) calculated by averaging the squared intensity differences between the distorted image pixels and the original image pixels. When an image has a low MSE, the image regards as high quality. The MSE between the two images x and $\hat{x}$ is computed as Eq 5.1.

$$MSE(x, \hat{x}) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x})^2. \tag{5.1}$$

The error signals, $e_i = x_i - \hat{x}_i$, denotes the difference between the reference and distorted signal. The MSE is used to the measurement of PSNR;

$$PSNR = 10log_{10}\frac{255^2}{MSE}, \tag{5.2}$$

where 255^2 is the range of allowable pixel intensities. As EC means how much secret bits can be hidden in the cover image. Here, an important thing is that the increase of EC is not decreasing PSNR to a given lower limit. That is, it should be kept above 30 dB.

Structural Similarity Index Metric (SSIM) [22] is used to evaluate the resemblance between the cover image and the stego-image. SSIM is designed to deal with the quality of the whole image by constructing and combining equations with luminance, contrast, and structure that are recognized as the main contents in human vision. The average luminance, contrast and structural information are combined together to produce the quality index. The yield of SSIM value is limited in the range

between 0 and 1. If the SSIM value is near to 1 that means the stego-image is alike the cover image and it has high quality. Equation 5.3 is used to calculate the value of SSIM:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2\mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{5.3}$$

where, μ_x and μ_y are mean values of cover image x and stego-image y and σ_x and σ_y are standard deviation values of the cover image and stego image while, σ_{xy} means the covariance of both two images. c_1 and c_2 are constants to stabilize the division.

Table 1 shows The comparison of the proposed scheme (methods) with other schemes when T = 20. Here, Ou and Sun's method has the lowest performance and W. Hong's method is second low in respect of EC. Method 3 has RDH scheme and is far superior to Ou and Sun's method in respect of PSNR. In addition, Ou and Sun's method forces to make sacrifice of a part of PSNR to improve EC. While, since Method 3 is based on expanded *trios*, it is no needed to sacrifice of PSNR to increase EC. The proposed Method 3 showed that it maintains good image quality of the cover image as the average PSNR is 32.8923 dB.

Table 1. The comparison of the proposed scheme (methods) with other schemes when T = 20.

Images	Ou and Sun		Huang et al.		W. Hong	
	EC	dB	EC	dB	EC	dB
Bridge	114609	26.0652	174006	26.0435	147025	26.0570
Elaine	156993	26.4091	213741	26.2139	189166	26.3975
Boats	205809	29.5664	259506	29.2865	237577	29.5704
Goldhill	212193	29.1224	265536	29.3792	244409	29.1093
Airplane	226977	30.1906	279381	29.6765	258511	30.1813
Lena	233697	30.7508	285651	30.3321	265759	30.7268
Peppers	240977	30.6910	292476	30.4176	273364	30.6831
Tiffany	241921	31.4269	293781	31.0565	274710	31.3882
Zelda	253841	31.5579	304536	31.2446	286249	31.5050
Average	209669	29.5311	263179	29.2945	241863	29.5132
Images	Method 1		Method 2		Method 3	
	EC	dB	EC	dB	EC	dB
Bridge	392897	26.1187	294673	26.3147	108470	26.8403
Elaine	562433	26.4467	421825	26.5940	148205	26.9762
Boats	757697	29.6727	568273	30.0646	195785	30.6758
Goldhill	783425	29.2949	587569	30.0241	199997	32.0948
Airplane	842497	30.3093	631873	30.6669	213843	31.7354
Lena	869249	30.9137	651937	31.4384	220115	33.1181
Peppers	898369	30.8920	673777	31.4790	226940	33.4414
Tiffany	903937	31.6537	677953	32.3379	228217	34.7671
Zelda	949825	31.8088	712369	32.5895	239000	35.4920
Average	773370	29.6789	615697	30.6493	209013	32.2876

For EC, Huang et al.'s method shows good performance except for Methods 1 and 2. In Table 1, Method 2 showed the best performance in aspect of EC and PSNR. Therefore, Methods 1 and 2 outperform the other schemes such as Huang et al. and W Hong in terms of PSNR and EC. The method proposed by Huang et al. grows the EC according to an increase of the complexity of the image texture and threshold T. On the other than, it has a problem that the MSE of the stego image becomes larger in proportion to threshold T.

In case of Huang et al.'s method, as the threshold T is increased, the room for DH increases. On the other hand, the quality of the stego image decreases in proportion to the difference between two quantization levels (i.e., $(b-a) \times log_2 T \times \#$ of changed pixels in each block in Huang et al.'s scheme). This problem may be also happened in our proposed the methods. However, in our proposed method, when the threshold is T $\leq$ 5 (e.g., *Lena* image), EC of Method 1 is higher than that of Huang et al.'s method. As a result, we improved the performance of the two criteria for DH compare to existing schemes.

As shown in Figure 8, Methods 1 and 2 perform best in terms of EC and PSNR, and Method 3 is good that of Huang et al.'s method in terms of PSNR, because it makes the fewest pixel modifications. The PSNRs of Methods 1 and 2 are almost the same until secret bits are 320000 bits. From more than 160000 bits of x-axis, the gap of PSNRs between Methods 1,2 and Method 3 begins to wide. The reason is that Method 3 could not find resource for DH under threshold T defined in Methods 1,2. Thus, to embed the amount of secret data the same as Methods 1,2, the threshold T of Method 3 should increase.

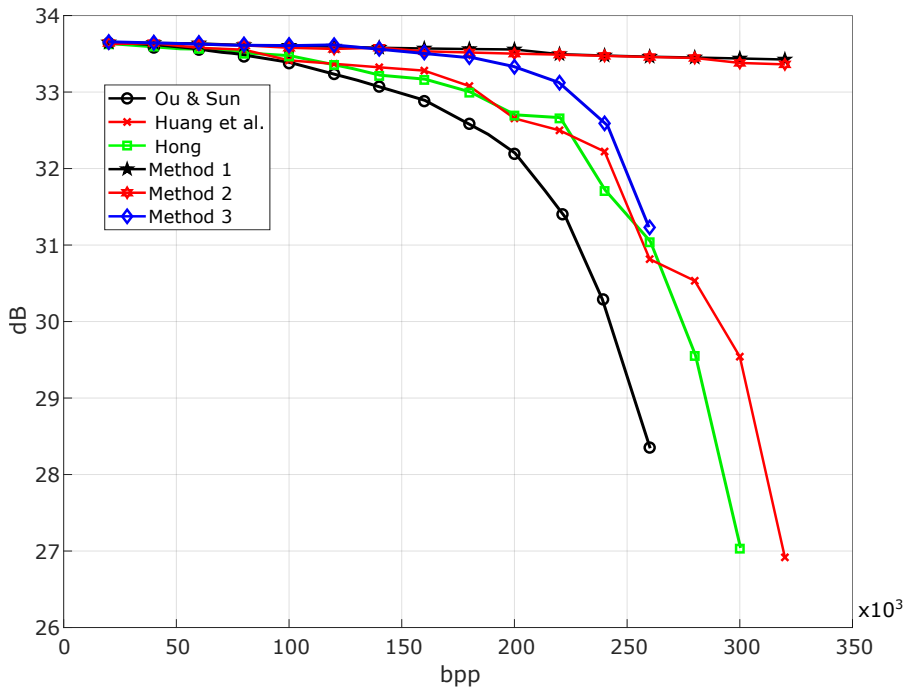

Figure 8. Comparison of performance between previous schemes and proposed scheme (Method 1) with *Lena* image.

Increasing the threshold T can increase the number of pixels used for data hiding, but adversely

affects the quality of the image. An increased threshold T means that the interval between two quantization levels is widened and it is revealing that when flipping one pixel in the bitmap BM, noises are increased as much as threshold T = 5. Methods 1 and 2 are capable of obtaining a high PSNR because it is possible to secure pixels capable of concealing more than 300000 bits or more of data under T = 5. In the case of Huang et al.'s method, the PSNR of the image shows more than 30 dB when the payload is 280000 bits, but the performance of the scheme is lower than that of our proposed method. That is because Huang et al.'s scheme does not have enough resource to embed secret data as much as our proposed scheme under the same threshold T.

Ou and Sun use two DBS and OTQL mixed method, thus the EC is limited and it revealed that the inflection point of Lena image is 240000 bit. W Hong improves the performance of EC and PSNR using APPM algorithm, but there is a problem that PSNR drops below 30 dB when embedding data over 280000 bits. Certainly, it is proved that the APPM algorithm can guarantee performance when applied only to the LSB. However, when the difference between the two quantization levels is big, the performance is deteriorated.

Table 2 is comparison of PSNR and SSIM for Airplane images according to increasing EC. When EC = 20000, PSNRs were measured to be higher than 31 dB in all methods. SSIM shows the result of that existing three methods are 0.84 or 0.85 respectively, while our proposed Methods 1–3 are more than 0.93. In case of EC = 200000, the PSNRs of the existing methods show more than 31 dB. SSIM shows the result of that existing three methods are more than 0.5, while our proposed Methods 1–3 are more than 0.91. Even if we may embed the secret bits into the cover image with the same amount of data, we can see that our method is higher SSIM rather than that of the other existing methods. For PSNR, existing and proposed methods show excellent results, while it is difference with the results we have expected when we are measure by SSIM. In consequence, when measured by SSIM, we can see that the stego image generated by our method is closer to the cover image. In other words, it proved a better way in terms of image quality, which is an important criterion for DH.

Figure 9 compares the quality of Airplane stego images, which are generated from existing and the proposed methods when EC = 260000. We can discern that the staircase effects appear at the upper left corner of the Airplane image generated by Ou and Sun, and W. Hong.

In this experiment, it looks that the performance of Ou and Sun are the lowest in aspect of PSNR and SSIM. Methods 1 and 2 show high PSNR and SSIM rather than that of existing methods. Method 3 has a high embedding efficiency in limited ranges, because it is possible to hide 3-bit in 7-codewords by flipping 1-bit. However, it needs enough resources more than the Methods 1 and 2. When we increase threshold T to obtain EC as the same as Methods 1 and 2, it may give the bad effect to PSNR and SSIM. When the ECs of Methods 1 and 2 are 260000, the PSNRs are 31.96 dB and 31.92 dB. Moreover, since the SSIMs are 0.92, it may very safe from the attack of steganalysis.

Table 3 summarizes comparison of the EC of the proposed scheme (Methods 1–3) with that of existing schemes. Hong et al.'s method hides 1-bit per block (of both complex and smooth) with lossless using the OTQL, thus it is possible to recover original cover image. The total EC of Hong's method is NS+NX = 16384.

Ou and Sun's method uses both DBS and OTQL on smooth blocks to obtain the EC of (16×NS)+(NS+NX). Huang et al. may obtain the EC of (16×NS)+(2×(NS+NX)+NX using DBS (on smooth blocks), OTQL (on complex blocks), and pixels difference. W. Hong's method use DBS, OTQL, and PPM for DH and the predicted EC is (16×NS)+(2×(NS+NX)+NX.

Table 2. PSNR vs. SSIM for Airplane image according to increasing EC.

EC	Ou and Sun		Huang et al.		W. Hong	
	dB	SSIM	dB	SSIM	dB	SSIM
20000	31.7846	0.8414	32.0286	0.8512	31.9705	0.8491
40000	32.0012	0.8156	32.0209	0.8336	31.9044	0.8270
80000	31.9595	0.7434	32.0060	0.7813	31.9134	0.7730
120000	31.8540	0.6610	31.8891	0.7143	31.9034	0.7198
160000	31.665	0.5761	31.8496	0.6504	31.8083	0.6416
200000	31.1367	0.5061	31.4387	0.5559	31.5517	0.5538
240000	29.2528	0.4288	31.1395	0.4954	30.8766	0.4858
EC	Method 1		Method 2		Method 3	
	dB	SSIM	dB	SSIM	dB	SSIM
20000	32.0356	0.9307	32.0317	0.9303	32.0370	0.9307
40000	31.8404	0.9291	32.0267	0.9299	32.0340	0.9303
80000	31.9931	0.9291	32.0192	0.9294	32.0262	0.9293
120000	32.0063	0.9289	32.0120	0.9290	32.0107	0.9273
160000	32.0084	0.9285	32.0034	0.9283	31.9724	0.9233
200000	32.0006	0.9279	31.9777	0.9259	31.8444	0.9153
240000	31.9718	0.9250	31.9672	0.9250	31.0439	0.8953

Table 3. Performance comparison for EC of *Lena* image when $T = 4$ (Notes: NS and NX: The number of smooth and complex blocks).

Method	Block size	(a, b)		BM	Order of (a, b)		EC	Verification
		NS	NX	NS	NS	NX		
Hong et al.	16	0		0	16384		16384	NS + NX
Ou and Sun	16	0		5906	16384		110880	(16xNS) + (NS+NX)
Huang et al.	16	$d = \log_2 T$		5906	0	10478	137742	(16xNS)+(2x(NS+NX))+NX
Wien Hong	16	$\sigma = 4$		5906	0	10839	132327	(16xNS)+(2x(NS+NX))+NX
Method 1	64	0		5906	0	0	377984	(64xNS)
Method 2	64	0		5906	0	10478	293966	(48xNS)
Method 3	64	0		5906	0	10478	109531	(15xNS)+NX

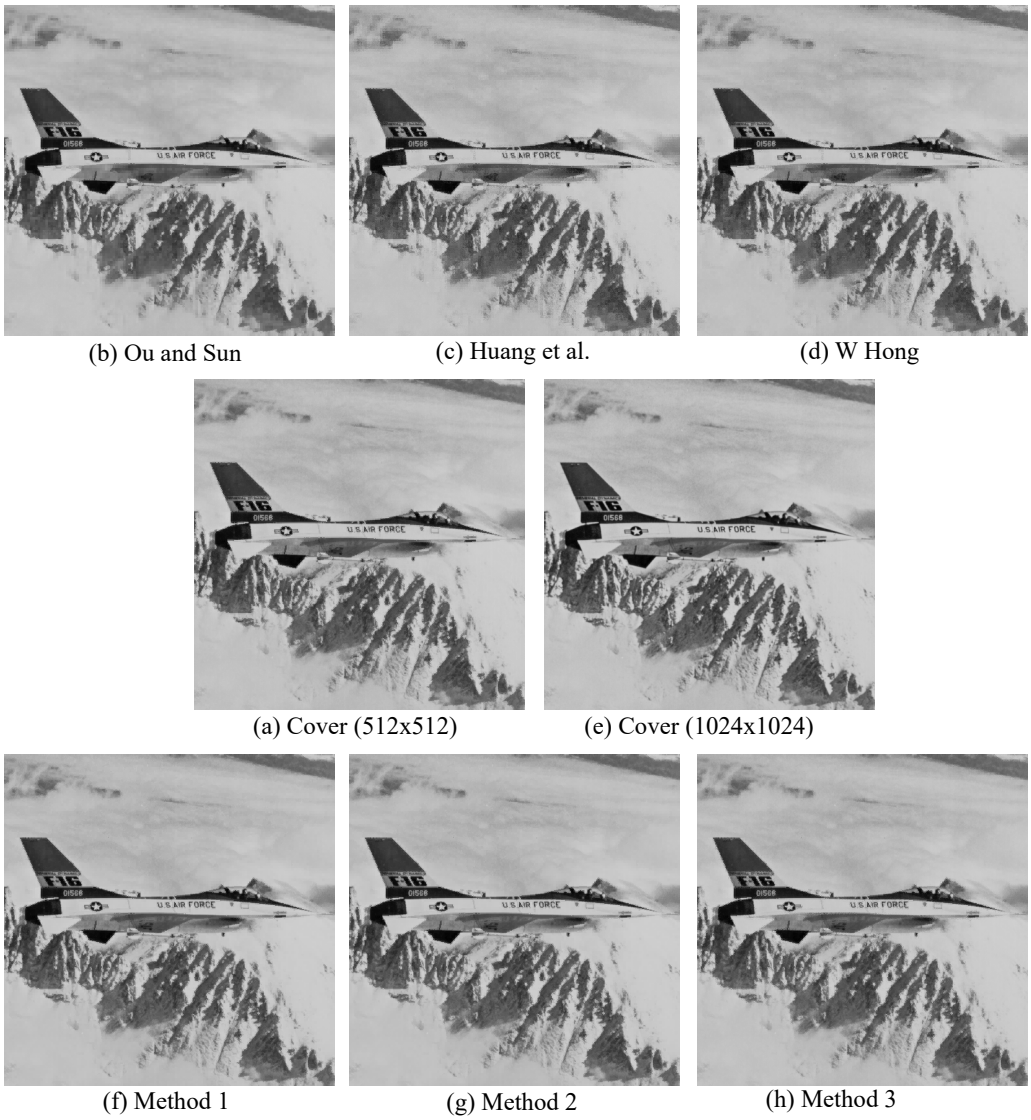

Figure 9. The quality and threshold T of stego AMBTCs when EC = 260000. (a) Original cover AMBTC, (b) Huang et al., (c) Ou and Sun, (d) W. Hong, (e) Cover AMBTC (1024×1024), (f) Method 1, (g) Method 2, (h) Method 3; (b) T = 53, 26.61 dB, SSIM: 0.37, (c) T = 16, 30.01 dB, SSIM: 0.42, (d) T = 21, 30.07 dB (σ = 4) SSIM: 0.83, (f) T = 3, 31.96 dB, SSIM: 0.92, (g) T = 4, 31.96 dB, SSIM: 0.92, (h) T = 99, 29.31 dB, SSIM: 0.87.

Method 1 embeds secret bits into the smooth blocks as much as (64×NS). Method 2 obtains the EC of (NS×48)+NX using DBS and OTQL from smooth blocks. For embedding secret bits, Method 3 uses the HC(7,4) and OTQL. When T = 4, total bits is (15×NS)+NX.

6. Conclusion

We proposed three DH methods based on AMBTC expanded by NMI with low complexity and high computation speed. The cover image used in the proposed DH was created from the original grayscale image by AMBTC and NMI. Secret data are embedded into compressed AMBTC images

(i.e., $trios(a, b, BM)$) by using Methods 1–3. The performance (EC and PSNR) is superior to existing schemes under the same threshold T. Methods 1 and 2, which may provide sufficient data and image quality, can be a very good choice.

Acknowledgments

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2018R1D1A1B07047395), National Research Foundation of Korea (NRF) funded by (2015R1D1A1A01059253), was supported under the framework of international cooperation program managed by NRF (2016K2A9A2A05005255), and was supported in part by Ministry of Science and Technology (MOST), under Grant 1082221E259009MY2.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. W. Bender, D. Gruhl, N. Morimoto, et al., Techniques for data hiding, *IBM Syst. J.*, **35** (1996), 313–336.
2. C. N. Yang, S. C. Hsu and C. Kim, Improving stego image quality in image interpolation based data hiding, *Comput. Stand. Interfaces*, **50** (2017), 209–215.
3. C. Kim, D. Shin, C. N. Yang, et al., Improving capacity of Hamming (n,k)+1 stego-code by using optimized Hamming+k, *Digital Signal Process.*, **78** (2018), 284–293.
4. Y. Q. Shi, X. Li, X. Zhang, et al., Reversible data hiding: Advances in the past two decades, *IEEE Access*, **4** (2016), 3210–3237.
5. F. Huang, X. Qu, H. J. Kim, et al., Reversible Data Hiding in JPEG Images, *IEEE Trans. Circuits Syst. Video Technol.*, **26** (2016), 1610–1621.
6. J. Mielikainen, LSB matching revisited, *IEEE Signal Proc. Let.*, **13** (2006), 285–287.
7. C. K. Chan and L. M. Cheng, Hiding data in images by simple LSB substitution, *Pattern Recognit.*, **37** (2004), 469–474.
8. C. C. Chang, C. C. Lin, C. S. Tseng, et al., Reversible hiding in DCT-based compressed images, *Inf. Sci.*, **177** (2007), 2768–2786.
9. C. C. Chang, T. S. Chen and L. Z. Chung, A steganographic method based upon JPEG and quantization table modification, *Inf. Sci.*, **141** (2002), 123–138.
10. C. Bergman and J. Davidson, *Unitary embedding for data hiding with the SVD*, Proceedings Volume 5681, Security, Steganography, and Watermarking of Multimedia Contents VII, (2005). Available from: https://doi.org/10.1117/12.587796.
11. E. Delp and O. Mitchell, Image compression using block truncation coding, *IEEE Trans. Commun.*, **27** (1979), 1335–1342.

12.W. Hong, T. S. Chen and C. W. Shiu, *Lossless steganography for AMBTC compressed images*, 2008 Congress on Image and Signal Processing, **2** (2008), 13–17. Available from: https://ieeexplore.ieee.org/document/4566259.

13.J. C. Chuang and C. C. Chang, Using a simple and fast image compression algorithm to hide secret information, *Int. J. Comput. Appl.*, **28** (2006), 329–333.

14.J. Chen, W. Hong, T. S. Chen, et al., Steganography for BTC compressed images using no distortion technique, *Imaging Sci. J.*, **58** (2010), 177–185.

15.W. Hong, J. Chen, T. S. Chen, et al., Steganography for block truncation coding compressed images using hybrid embedding scheme, *Int. J. Innov. Comput. Inf. Control*, **7** (2011), 733–743.

16.D. Ou and W. Sun, High payload image steganography with minimum distortion based on absolute moment block truncation coding, *Multimed. Tools Appl.*, **74** (2015), 9117–9139.

17.J. Bai and C. C. Chang, A high payload steganographic scheme for compressed images with hamming code, *Int. J. Netw. Secur.*, **18** (2016), 1122–1129.

18.Y. H. Huang, C. C. Chang and Y. H. Chen, Hybrid secret hiding schemes based on absolute moment block truncation coding, *Multimedia Tools Appl.*, **76** (2017), 6159–6174.

19.W. Hong, Efficient data hiding based on block truncation coding using pixel pair matching technique, *Symmetry*, **10** (2018), 1–8.

20.W. Hong, and T. S. Chen, A novel data embedding method using adaptive pixel pair matching, *IEEE Trans. Inf. Forensics Secur.*, **7** (2012), 176–184.

21.K. H. Jung and K. Y. Yoo, Data hiding method using image interpolation, *Comput. Stand. Interfaces*, **31** (2009), 465–470.

22.Z. Wang, A. C. Bovik, H. R. Sheikh, et al., Image quality assessment: From error visibility to structural similarity, *IEEE Trans. Image Process.*, **13** (2004), 600–612.

AIMS Press