Peer-reviewed version available at *Appl. Sci.* **2020**, *10*, 5336; doi:10.3390/app10155336

Article

Hybrid Data Hiding Based on AMBTC Using Enhanced Hamming Code

Cheonshik Kim *[1,‡], **Dong-Kyoo Shin**[1], **Ching-Nung Yang** [2,‡] **and Lu Leng** *[3,4]

1 Department of Computer Engineering, Sejong University, Seoul 05006, Korea; mipsan@sejong.ac.kr
2 Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien 97401, Taiwan; cnyang@gms.ndhu.edu.tw
3 Key Laboratory of Jiangxi Province for Image Processing and Pattern Recognition, Nanchang Hangkong University, Nanchang 330063, People's Republic of China; drluleng@gmail.com
4 School of Electrical and Electronic Engineering, College of Engineering, Yonsei University, Seoul 120749, Republic of Korea
* Correspondence: L. Leng, drluleng@gmail.com; C. Kim, mipsan@sejong.ac.kr
‡ These authors contributed equally to this work.

Abstract: We present a method of effectively concealing data in two quantization levels representing blocks based on Absolute Moment Block Truncation Coding (AMBTC) using Hamming code. The reason for choosing AMBTC as the cover media is that AMBTC is not only easily compressible by simple arithmetic operations, but is also efficient for exchanging data in small portable terminals, such as smart devices, because compression efficiency and image quality are satisfactory. AMBTC is expressed in the form of a $trio(a, b, BM)$, where a and b are quantization levels and BM is a bitmap. The quantization level is a pixel value used as a representative value when dividing an image into 4×4 blocks. Bai & Chang introduced a method of hiding data using Hamming code. However, this method presented a problem with efficiency compared to the existing methods. In this paper, in order to improve the problem of this method, we minimize the error that occurs when concealing the data using the optimized Hamming code lookup table. We verified that the performance of the proposed method is satisfactory in terms of embedding capacity and quality of the image through sufficient experimental results.

Keywords: Data Hiding; AMBTC; BTC; Hamming Code; LSB

1. Introduction

Recently, the Internet space is like a single trading world where almost all digital contents are distributed because every trading system is connected to high speed Internet like 5G. Many people distribute digital contents in this space and are constantly consuming digital contents. The problem with this digital space is that a copyright protection problem occurs because digital contents are easily redistributed, copied, and modified by illegal users. There are various solutions to this problem, but the commonly used method is digital watermarking [1–3], which is used to protect the integrity and reliability of digital media.

Besides the watermarking technology, the data hiding (DH) technology is the most commonly used method of concealing information in digital media. The DH [4–6] technique can be used to various fields, such as digital signatures, fingerprint recognition, authentication, and secret communication. It is proved various times that DH could be used for secret communication as well as protection of the copyright of digital contents. The people who use Internet communication know that Internet is not a fully protected communication channel due to a lot of attackers. But secret communication using DH can safely protect secret messages in digital cover media from incomplete Internet channel.

DH may achieve the role of a secret communication only when it satisfies two important criteria. First, the quality of the cover image (including data) should not be significantly different from the quality of the original image, since the cover image must not be detected by attackers during it is transmitted. Second, it must have an ability to transmit a lot of secret data to the receiver securely.

The DH method is mainly conducted in two domains, namely spatial domain and frequency domain. In spatial domain, a secret bit is concealed into the pixels of a host image directly. In case DH based on spatial domain is applied to a grayscale image, even though the four least significant bits (LSBs) [7–10] of each pixel are used for information hiding, they may not be detected by the human visual system (HVS).

In frequency domain, a cover image is converted into a frequency form and then the data are concealed into the coefficients of the frequency. The two most common methods based on frequency domain are discrete cosine transform (DCT) [11,12] and discrete wavelet transform (DWT) [13]. Since changing the coefficient adversely affects the image quality, it is necessary to find and change the positions of the coefficient that have a relatively small influence on the image quality during data insertion. Spatial domain methods have a merit as an ability of concealing a lot of secret data rather than that of frequency domain methods and the quality of an image is better. While, they have a vulnerable demerit from the compression, noises, and filtering attacks compared to frequency domains. Meanwhile, the excellent compression image like JPEG is preferred as a digital media, because the file size is small compared to raw images and is good at transmission. For this reason, many researchers closely studied the watermarking and DH methods based on JPEG compression image long time ago.

Block Truncation Coding (BTC) [14] is one of the compression methods and the configuration of the BTC is very simple compared to conventional JPEG. Thus, the computation time of BTC is much shorter than that of JPEG, and the quality of an image based on BTC is not significantly deteriorated compared to that of the original image. For this reason, it seems many researchers are interested in DH based on Absolute Moment Block Truncation Coding (AMBTC) [15–17] originated from BTC recently. Chuang and Chang [18] proposed a DH method based on AMBTC replacing the bitmaps of smooth blocks with the secret bits after dividing blocks of an image into smooth blocks and complex blocks directly. It is called direct bitmap substitution (DBS) method. The merit of this method may control the quality of stego image through adjusting threshold $T(= b - a)$ because the number of blocks using DH is decided according to the threshold value T. Here, the a and b are quantization levels for each block in AMBTC. According to increasing the threshold T, the embedding capacity will be increased, while the image quality will be worse. In case of decreasing the threshold T, the quality of the image will be improved, but the embedding capacity may be reduced.

Ou and Sun [19] introduced a way to embed data in the bitmaps of smooth blocks and proposed a method to reduce distortions of the image by adjusting two quantization levels through the re-computation, but the original image is required for re-calculation. Bai and Chang [20] proposed a way to embed secret data by applying a Hamming code, i.e. HC(7,4) [7] to two quantization levels and bitmaps of AMBTC, respectively. When HC (7,4) is used for a complex block of AMBTC, it may be undesirable for high image quality. Kumar et al [21] used two threshold values to increase the capacity of DH without significantly improving the image quality. Chen et al. [22] proposed lossless DH method using the order of two quantization levels in *trio*. This method is named Order of Two Quantization Level (OTQL) method, which can conceal 1-bit per a block. For example, to store the bit '1', the order of two quantization levels, a and b, are reversed as $trio(b, a, BM)$. This method does not change the coefficients of both quantization levels, so it does not affect the quality of the image.

Hong [23] proposed a DH using Pixel Pair Matching (PPM) [24], where PPM is applied to the quantization level; while the existing OTQL and DBS are used together for complex and smooth blocks, respectively. In 2017, Huang et al [25] proposed a scheme for hiding data using pixel differences (hidden bits $=log_2 T$: derived from the difference expansion method) at two quantization levels, and besides introduced a method to adjust the differences in quantization levels to maintain image quality. This method is also a hybrid method by using OTQL and DBS additionally. Chen and Chi [26]

sub-divided less complex block and highly complex block. In 2016, Malik et al [27] introduced a DH based on AMBTC using 2-bit plane and 4 quantization levels. The merit of this method is high payload, and the demerit is to decrease the compression rates.

In this paper, we intend to propose a method to increase the image quality and DH capacity by minimizing the distortion of the cover image by improving the method of Bai and Chang (2016) [20]. Their DH method applies Hamming code to the codeword, where the 7-bit codeword is constructed by using two quantization levels. When the Hamming code method is used for the quantization level, image distortion can increase significantly compared to the existing methods. To solve this problem, we had studied Hamming code theories. As a result, we found an optimized method of the Hamming code to reduce errors using look up table. Our proposed method is to find the codeword with the smallest error in the code word list and applies the Hamming code to the both quantization levels. To this end, our method calculates the codeword corresponding to the minimum distance from the standard array of the (7,4) Hamming code table, and then extracts the corresponding code. The method has little effect on program performance and can be easily conducted. The minimum distance of our proposed Hamming coding may improve the performance of DH and is expected to be actively used in other studies. Experimental results show that the proposed method achieves higher payload and better image quality compared to the existing DH methods.

The rest of this paper is organized as follows. Section 2 gives the introduction of related works. The proposed method is described in detail in Section 3. The experimental results are analyzed in Section 4. Section 5 draws the conclusion.

2. Related Works

2.1. AMBTC

Absolute Moment Block Truncation Coding(AMBTC) [15] efficiently improves the computation time of Block Truncation Coding (BTC) and improves the image quality over BTC. The basic configuration of one block in AMBTC is two quantization levels and one bitmap, while one block is compressed by preserving the moment. Here, the two quantization values are obtained by calculating the higher mean and the lower mean of each block. For AMBTC compression, the gray image is first divided into $(k \times k)$ blocks without overlapping, where k can determine the compression level by (4×4), (6×6), (8×8), etc. AMBTC adopts block-by-block operations. For each block, the average pixel value is calculated by

$$\bar{x} = \frac{1}{k \times k} \sum_{i=1}^{k^2} x_i \tag{1}$$

where x_i represents the i^{th} pixel value of this block with the size of $k \times k$. All pixels in this block is quantized into a bitmap b_i (0 or 1). That is, if the corresponding pixel x_i is greater than or equal to the average ($\bar{x}$), it is replaced with '1', otherwise it is replaced with '0'. Pixels in each block are divided into two groups of '1' and '0'. The symbol t and $k^2 - t$ refer to the numbers of pixels in the '0' and '1' groups respectively. The means a and b of the two groups indicate the quantization levels of the groups '0' and '1'. The two quantization levels are calculated by Equations (2) and (3).

$$a = \left\lfloor \frac{1}{t} \sum_{x_i < \bar{x}} x_i \right\rfloor \tag{2}$$

$$b = \left\lfloor \frac{1}{k^2 - t} \sum_{x_i \geq \bar{x}} x_i \right\rfloor \tag{3}$$

where a and b are also used to reconstruct AMBTC.

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

$$g_i = \begin{cases} a, & \text{if } b_i = 0, \\ b, & \text{if } b_i = 1. \end{cases} \tag{5}$$

The bitmap is obtained from Equation (4) and the compressed block is simply uncompressed by using Equation (5). That is, the compressed code unit, $trio(a, b, BM)$, may be obtained by using Equations (2)-(5). The image block is compressed into two quantization levels a, b, and a bitmap BM, and can be represented as a $trio(a, b, BM)$. A bitmap BM contains the bit-planes that represent the pixels, and the values a and b are used to decode the AMBTC compressed image by using equation (5).

Example 1:
Here, we describe the encoding and decoding procedure of one block of a grayscale image using an example. Figure 1(a) is a grayscale block and the mean value of pixels is 106. By applying Equations (2),(3) and (4) on (a), we can obtain the bitmap as shown in (b) and two quantization levels ($a = 102; b = 107$). The basic unit of each block is $trio(a, b, BM)$ = (102, 107, 0101111111011001). Using the information of the *trio* and Equation (5), the decoded grayscale block in (c) is reconstructed.

102	107	104	110
109	106	107	106
110	112	104	106
106	101	103	107

(a) a natural block

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

(b) bitmap

102	107	102	107
107	107	107	107
107	107	102	107
107	102	102	107

(c) a reconstructed block

Figure 1. An example of AMBTC: (**a**) a natural block. (**b**) a bitmap block. (**c**) a reconstructed block.

2.2. Hamming code

Hamming code (HC) [28] is a single error-correcting linear block code with a minimum distance of 3 for all the codewords. In HC(n, k), n is the length of the codeword, k is the number of information bits and $(n - k)$ is the number of parity bits.

Let x be a k bit information word. The n-bit codeword y is created by using $y = xG$, where G is the $k \times n$ generator matrix. Let $e = y - \tilde{y}$ be the error vector that determines whether an error occurred while sending y. If $e = 0$, no error occur and $\tilde{y} = y$.

Otherwise, the weight of e represents the number of errors. Let **H** be a $(n - k) \times n$ parity matrix with the relation of $G \cdot H^T = [0]_{k \times (n-k)}$. Let we assume that the codeword $\tilde{y}$ has an error like $e = (y - \tilde{y})$. In this case, we could correct 1-error ($e = y \oplus \tilde{y}$) from the codeword $\tilde{y}$ by using the syndrome $S = \tilde{y} \cdot H^T$, where the syndrome denotes the position of the error in the codeword. As show in Equation (6), the error e can be obtained.

$$\begin{cases} \tilde{y} \cdot H^T = (e \oplus y) \cdot H^T = e \cdot H^T + y \cdot H^T \\ (y \cdot H^T) = (x \cdot G) \cdot H^T = x \cdot (G \cdot H^T) = 0) \\ \qquad = e \cdot H^T + 0 = e \cdot H^T \end{cases} \tag{6}$$

Consider HC(7,4) code with the following parity matrix.

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{7}$$

For example, assuming that one error bit occurred in y (e.g., the second bit from the left in $e = (e_1, e_2, \ldots, e_7) = (0\underline{1}00000)$, we may obtain the error position and recover the 1-bit error from the codeword y by using calculation of syndrome $S(= y \cdot H^T = (010))$.

2.3. Bai and Chang's method

For DH, AMBTC algorithm is applied to the original cover image to obtain a low mean, a high mean and a bitmap for every block. Then, the secret message is concealed into the AMBTC compressed $trio(a, b, BM)$. The merit of AMBTC is to achieve a higher payload compared to other data hiding schemes performed on the compression domain. Here, it is performing AMBTC DH as two phases . The method proposed by Bai and Chang is composed of two stages. One of them is to embed 3-bit in two quantization levels in $trio(a, b, BM)$ by using HC(7, 4). The detailed process of this method is as follows.

Step 1: Column vector $y = (y_7, y_6, y_5, y_4, y_3, y_2, y_1)$ is constructed by extracting 4 LSBs of the low mean a and 3 LSBs of the high mean b. Three secret message bits $(m = (m_1, m_2, m_3))$ are read from the secret bit set M.

Step 2: Compute the syndrome $S(= Hy \oplus m)$ of the codeword y and then the value is changed into decimal value and assigned to the variable i. For obtaining the stego codeword $\hat{y} = (y_1, y_2, y_3, y_4, y_5, y_6, y_7)$, flip i^{th} bit of the codeword y.

Step 3: To reconstruct two quantization levels with the codeword y, (y_7, y_6, y_5, y_4) is replaced with 4 LSBs of the low-mean value a and (y_3, y_2, y_1) is replaced with 3 LSBs of the high-mean value b.

Step 4: It is possible to hide additional 1-bit by using the order of two quantization levels and the difference between them. In this case, it may be accepted to embed additional 1-bit when the criterion $(b - a \geq 8)$ is satisfied. Otherwise, it is not accepted to embed additional 1-bit. If the bit to be embedded is '1', swap the order of two quantization levels as $(trio(b, a, BM))$, otherwise none change is conducted.

In Step 4, it is possible to embed additional 1-bit only under the given condition $(b - a \geq 8)$. The reason for the condition is necessary, if the difference between the values of a and b is small, the order of the two values may be reversed as a result of the computation of the Hamming code. It is able to happen ambiguous result in the decoding procedure.

3. The Proposed Scheme

In this section, we introduce a DH to embed secret data in bitmaps and quantization levels of AMBTC using optimized Hamming code and DBS method. First, compressed blocks, trios, are classified into smooth blocks and complex blocks. Then, DBS is applied to the bitmaps of the smooth blocks, meanwhile Hamming code may be applied to the quantization levels regardless of the block characteristics. The method proposed by Bai and Chang has very large distortion of the cover image. In Section 3.1, we introduce the way to solve this problem.

3.1. Embedding Procedure

We introduce a way of DH using Hamming code, DBS, and OTQL based on AMBTC, and explain the detail procedure step by step as follows. Additionally, the flowchart of the embedding process is described in Figure 2.

Input: Original grayscale image G with a size of $N \times N$, threshold T and secret data $M = (m_1, m_2, \ldots, m_n)$.

Output: Stego AMBTC *trios*

Step 1: The original image G is divided into 4×4 non-overlapping image blocks.

Step 2: The $trio(a, b, BM)$ of the AMBTC, i.e. the compressed codes, is obtained according to Equations (1)-(4), where $\boldsymbol{a}$ and $\boldsymbol{b}$ are the lower and the higher quantization levels, respectively, and BM is the bitmap.

Step 3: The quantization levels are $\boldsymbol{a} = (a_8 a_7 \dots a_1)$ and $\boldsymbol{b} = (b_8 b_7 \dots b_1)$, where a_8 is the most significant bit (MSB) of $\boldsymbol{a}$ and a_1 is the LSB of $\boldsymbol{a}$. Similarly, b_8 is the MSB of $\boldsymbol{b}$ and b_1 is the LSB of $\boldsymbol{b}$. The rearranged 7-bit codeword is obtained by

$$y = (a_4 a_3 a_2 a_1 || b_3 b_2 b_1) \tag{8}$$

where symbol $||$ denotes concatenation. Note that, a_4 and b_1 are the MSB and LSB of the rearranged pixel y, respectively.

Step 4: Codewords matching m_i^{i+2} bits can be obtained from the lookup table (LUT) by comparing in order (see Figure 3). In other words, it may get the codeword by converting m to decimal d and then searching for the coset reader that matches d in Figure 4 (b).

$$\epsilon = min((a - a')^2 + (b - b')^2) \tag{9}$$

To compute minimum distance between two codewords, y and searched codewords, Equation (9) may be used. When a codeword matched minimum distance is obtained, new codeword h is constructed like $(a'||b')$. After then, two quantization levels, a and b are constructed as follows:

$$\begin{cases} a = (a_8 a_7 a_6 a_5 || h_7 h_6 h_5 h_4) \\ b = (b_8 b_7 b_6 b_5 b_4 || h_3 h_2 h_1) \end{cases} \tag{10}$$

Before next step, 3 is added to the index variable i.

Step 5: If $|a - b| \leq T$, it may be a smooth block. For the smooth block, we use DBS. The m_i^{i+15} bits replace the pixels of the BM. 15 is added to the index variable i before next step. If $|a - b| > T$ and $|a - b| \geq 8$, OTQL is launched. If $m_i^i = 1$, transpose the order of two quantization levels, a and b, of the *trio*, otherwise, put the *trio* as original state.

Step 6: Repeat Steps 2∼5 until all image blocks are processed. Then, the stego AMBTC compressed codes *trio* is constructed.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 July 2020 doi:10.20944/preprints202007.0067.v1

Peer-reviewed version available at Appl. Sci. 2020, 10, 5336; doi:10.3390/app10155336

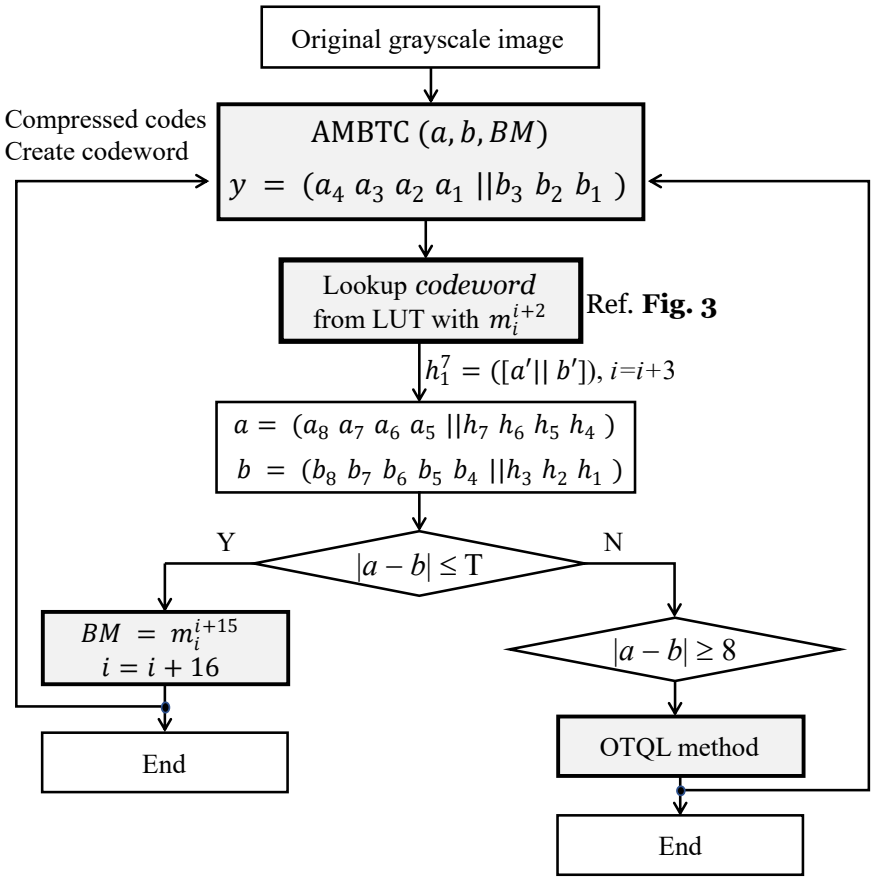

Figure 2. The flowchart of data embedding process.

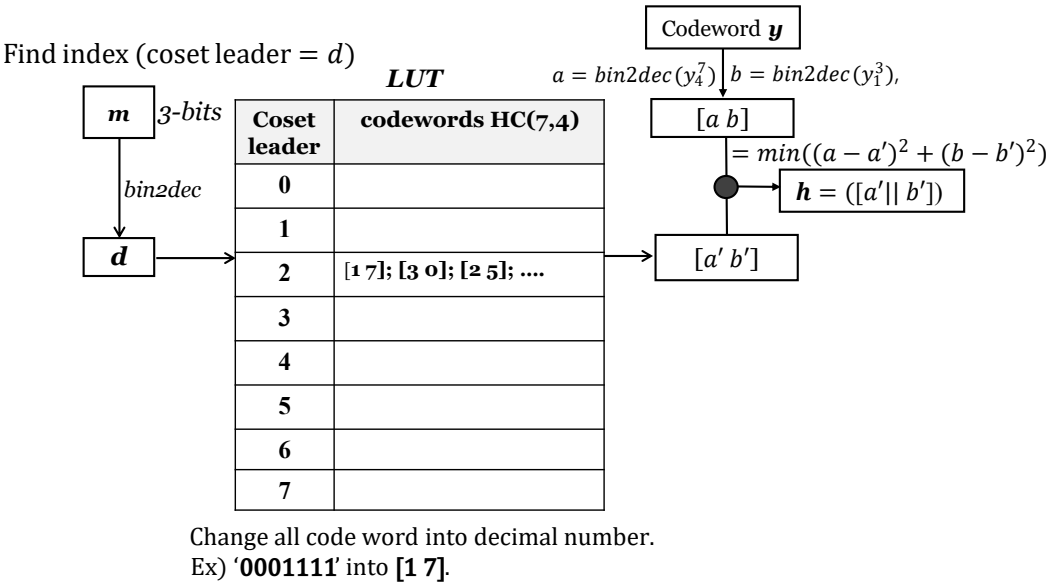

Figure 3. The flowchart of lookup codeword with m.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 July 2020 doi:10.20944/preprints202007.0067.v1

Peer-reviewed version available at Appl. Sci. 2020, 10, 5336; doi:10.3390/app10155336

Standard array of a (7,4) Hamming code for DH								
Coset leader								
0000000	(0001101)	(0011010)	(0010111)	(0110100)	(0111001)	(0101110)	(1000011)	(1101000)
	(1100101)	(1110010)	(1111111)	(1011100)	(1010001)	(1000110)	(1001011)	
0000001	(0001100)	(0011011)	(0010110)	(0110101)	(0111000)	(0101111)	(0100010)	(1101001)
	(1100100)	(1110011)	(1111110)	(1011101)	(1010000)	(1000111)	(1001010)	
0000010	(0001111)	(0011000)	(0010101)	(0110110)	(0111011)	(0101100)	(0100001)	(1101010)
	(1100111)	(1110000)	(1111101)	(1011110)	(1010011)	(1000100)	(1001001)	
0000011	(0001110)	(0011001)	(0010100)	(0110111)	(0111010)	(0101101)	(0100000),	(1101011)
	(1100110)	(1110001)	(1111100)	(1011111)	(1010010)	(1000101)	(1001000)	
0000100	(0001001)	(0011110)	(0010011)	(0110000)	(0111101)	(0101010)	(0100111)	(1101100)
	(1100001)	(1110110)	(1111011)	(1011000)	(1010101)	(1000010)	(1001111)	
0000101	(0001000)	(0011111)	(0010010)	(0110001)	(0111100)	(0101011)	(0100110)	(1101101)
	(1100000)	(1110111)	(1111010)	(1011001)	(1010100)	(1000011)	(1001110)	
0000110	(0001011)	(0011100)	(0010001)	(0110010)	(0111111)	(0101000)	(0100101)	(1101110)
	(1100011)	(1110100)	(1111001)	(1011010)	(1010111)	(1000000)	(1001101)	
0000111	(0001010)	(0011101)	(0010000)	(0110011)	(0111110)	(0101001)	(0100100)	(1101111)
	(1100010)	(1110101)	(1111000)	(1011011)	(1010110)	(1000001)	(1001100)	

(a)

Standard array of a (7,4) Hamming code for DH							
Coset leader	MSB 4-BIT to DECIMAL NUMBER, LSB 3-BIT to DECIMAL NUMBER Ex) (a_1 = 1, b_1 = 5) ∈ {(1 5) in coset leader 0}						
0	(1 5)	(3 2)	(2 7)	(6 4)	(7 1)	(5 6)	(8 3)	(13 0)
	(12 5)	(14 2)	(15 7)	(11 4)	(10 1)	(8 6)	(9 3)	
1	(1 4)	(3 3)	(2 6)	(6 5)	(7 0)	(5 7)	(4 2)	(13 1)
	(12 4)	(14 3)	(15 6)	(11 5)	(10 0)	(8 7)	(9 2)	
2	(9 7)	(3 0)	(2 5)	(6 6)	(7 3)	(10 4)	(4 1)	(13 2)
	(12 7)	(14 0)	(15 5)	(11 6)	(10 3)	(8 4)	(9 1)	
3	(1 6)	(3 1)	(2 4)	(6 7)	(7 2)	(10 5)	(4 0),	(13 3)
	(12 6)	(14 1)	(15 4)	(11 7)	(10 2)	(8 5)	(9 0)	
4	(1 1)	(3 6)	(2 3)	(6 0)	(7 5)	(5 2)	(4 7)	(13 4)
	(12 1)	(14 6)	(15 3)	(11 0)	(10 5)	(8 2)	(9 7)	
5	(9 0)	(3 7)	(2 2)	(6 1)	(7 4)	(5 3)	(4 6)	(13 5)
	(12 0)	(14 7)	(15 2)	(13 1)	(10 4)	(8 3)	(9 6)	
6	(1 3)	(3 4)	(2 1)	(6 2)	(7 7)	(5 0)	(4 5)	(13 6)
	(12 3)	(14 4)	(15 1)	(11 2)	(10 7)	(8 0)	(9 5)	
7	(1 2)	(3 5)	(2 0)	(6 3)	(7 6)	(5 1)	(4 4)	(13 7)
	(12 2)	(14 5)	(15 0)	(11 3)	(10 6)	(8 1)	(8 4)	

(b)

Figure 4. Standard array of HC(7,4) for DH; (a) binary presentation and (b) decimal presentation.

3.2. Extraction and Reconstruction Method

The procedure of extracting the hidden secret bits shows in Figure 5. The detailed process is explained in detail according to the following procedure.

Input: Stego AMBTC compressed codes *trios*, matrix $H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$, and threshold T.

Output: Secret data $M = (m_1, m_2, \ldots, m_n)$.

Step 1: Read one block of $trio(a, b, BM)$ from a set of *trio* as a defined order, where the *trio* consists of two quantization levels and one bitmap. Here, one codeword may be obtained from two quantization levels by Equation 8.

Step 2: The quantization level are $a = (a_8 a_7 \ldots a_1)$ and $b = (b_8 b_7 \ldots b_1)$, where a_8 is the MSB of a and a_1 is the LSB of a. Similarly, b_8 is the MSB of b and b_1 is the LSB of b. The rearranged 7-bit codeword is $y = (a_4 a_3 a_1 a_1 || b_3 b_2 b_1)$. Note that, a_4 and b_1 are the MSB and LSB of the rearranged pixel y, respectively.

Step 3: Obtain the syndrome $S = y \cdot H^T$. Then, assign S into the m_i^{i+2}, and add 3 to i.

Step 4: If $|a - b| \leq T$, it is a smooth block *trio*. In this case, it means that the hidden bits were embedded in BM as a form of pixels. Therefore, by assigning the pixels of BM to m in order, all the values concealed in the BM can be obtained. That is, $m_i^{i+15} = BM_1^{16}$ and $i = i + 15$. If $|a - b| > T$ and $|a - b| \geq 8$, one bit was hidden in *trio* by using the order of two quantization levels. If the order of two quantization levels is $trio(b, a, BM)$, it means that $m_i^i = 1$, otherwise $m_i^i = 0$.

Step 5: Repeat Steps $1 \sim 4$ until all the *trios* are completely processed, and the extracted bit sequence constitutes the secret data m.

3.3. Examples

The detailed procedure of our proposed DH is explained by the instance shown in Figure 6. $trio(103, 109, 0000010001110111)$ and secret bits, $m = (101101011100001100)$. Since $b - a = 109 - 103 = 6 \leq T(7)$, the *trio* is classified as a smooth block. Here, we will show how to minimize errors in the encoding process through an optimized method than the existing method.

First, a codeword is constructed by using two quantization levels, a and b. That is, the codeword $y(= (0111101)$ is made by 4LSB of a and 3LSB of b, where a and b are equals to $103 = (01100111)_2$ and $109 = (01101101)_2$ respectively. The constructed codeword is possible to hide 3-bits in two quantization levels. The 3 secret bits of m is $(101)_2 (d = 5$ in decimal). The value d is searched in the coset leader of standard array of HC (7,4) to find the matching index of the array. Then, using Equation

(9), as shown in the example in Figure 6, the codeword with the smallest distance from $([a_1\ b_1] = [7\ 5])$ is found, so the codeword is $(h = (7||4) = (0111||100)_2 = (0111100)$. Two quantization levels to embed 3 secret bits are recovered by using the codeword h.

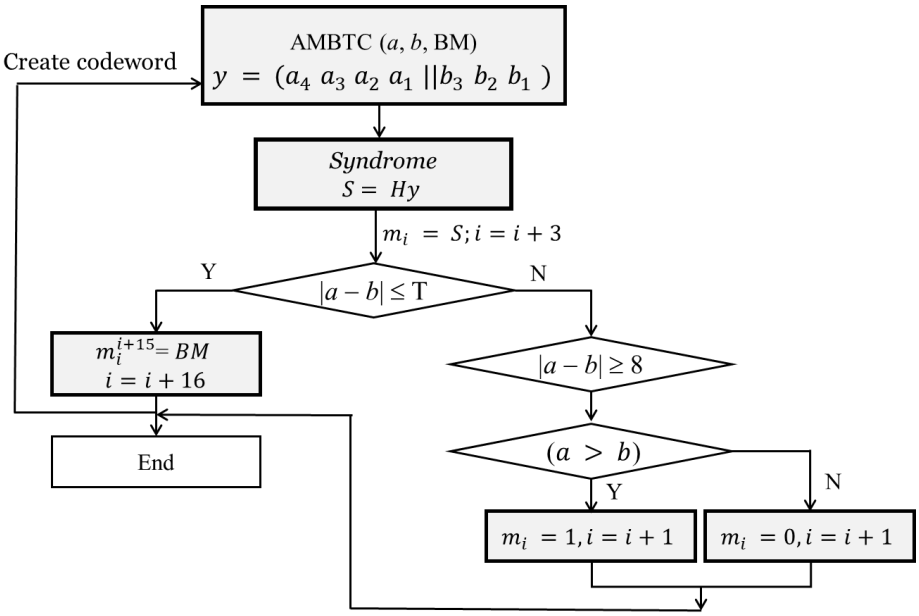

Figure 5. The flowchart of extracting procedure.

That is, the recovered codewords are $a = 103$ and $b = 108$. This block is a smooth block, because the difference $(|a - b|)$ between two quantization levels is less than or equal to T. Thus, it is acceptable to hide 16 bits in the BM by DBS. That is, the pixels of the BM is replaced with the secret 16 bits, m = (1010 1111 0000 1100), and the bits concealed at the BM. In this example, for hiding additional 1 bit, the method based on OTQL is not used because the criterion of $|a - b| > T$ is not satisfied. For decoding of two quantization levels, the codeword y should be constructed like decoding procedure. Here, we obtain the hidden secret bits, $\overline{m} = (101)$, by using the equation, $S = y \cdot H^T$, to the codeword. The decoding of the secret bits in the BM extracts the hidden bits by moving all pixels in the BM into a variable $\overline{m}$ of array directly.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 July 2020 doi:10.20944/preprints202007.0067.v1

Peer-reviewed version available at Appl. Sci. 2020, 10, 5336; doi:10.3390/app10155336

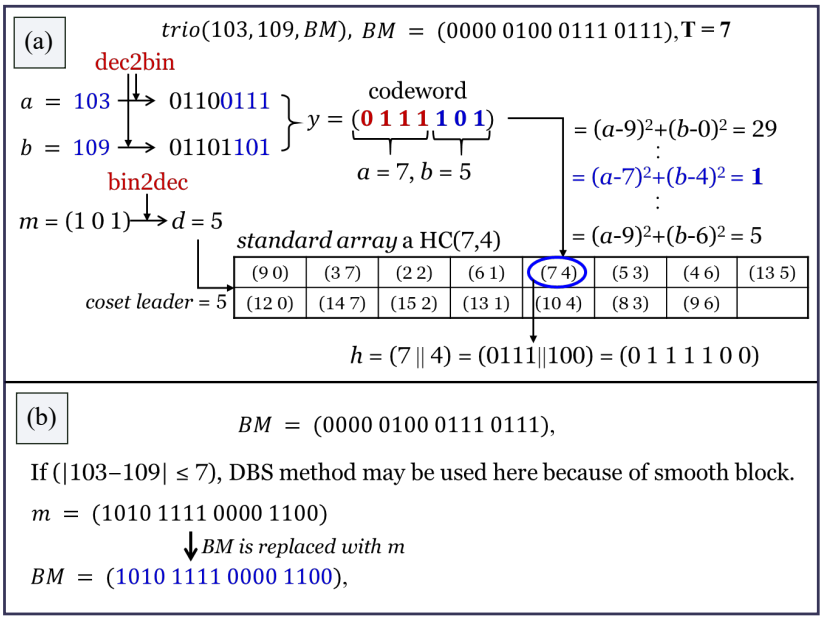

Figure 6. Illustration of data embedding.

4. Experimental Results

In this section, we prove the performance of our proposed scheme by comparing with the existing methods, such as Bai and Chang[20] , W Hong[23], and Chuang et al.[18] As shown in Figure 7, six grayscale images sized 512 × 512 are used for our experiments. In addition, the block size of AMBTC is set to 4 × 4 and the secret bits are generated by pseudo-random number generator. Embedding Capacity (EC) and Peak Signal-to-Noise Ratio (PSNR) are typically widely used as objective image evaluation indices. The relatively high PSNR value means that the quality of the stego image is good. The DH capacity is the size of the secret bit which is embedded in the cover image. The quality of the image is measured by PSNR defined as:

$$PSNR = 10log_{10}\frac{255^2}{MSE} \tag{11}$$

The MSE used in PSNR denotes the average intensity difference between the stego and reference images.

Peer-reviewed version available at Appl. Sci. 2020, 10, 5336; doi:10.3390/app10155336

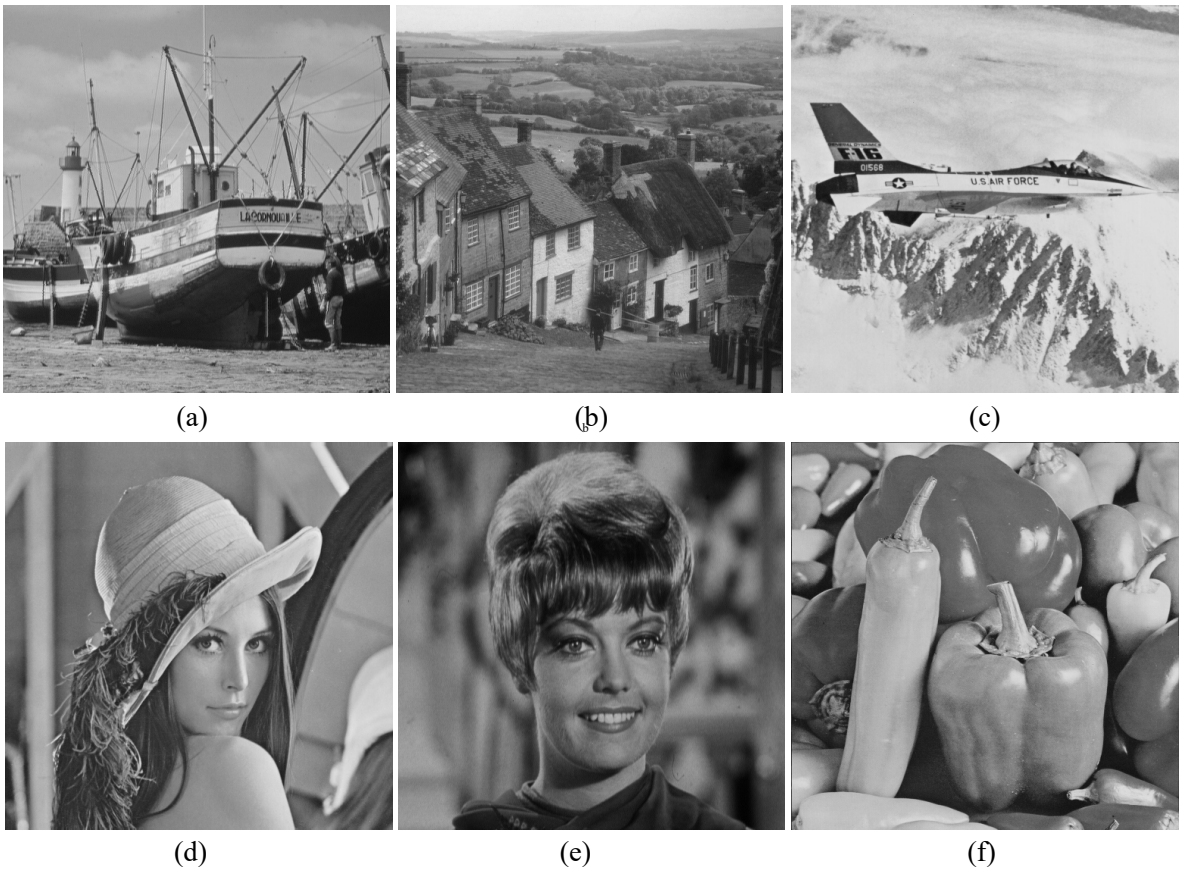

Figure 7. Test images; (a) ∼ (f): 512 × 512.

The lower the MSE value of a stego image, the better the quality of the image. MSE is calculated using the reference image p and the distorted image $p^{'}$ as follows.

$$MSE = \frac{1}{N \times N} \sum_{i=1}^{N} \sum_{j=1}^{N} (p_{ij} - p_{ij}^{'})^2 \tag{12}$$

The error value $\epsilon = p_{ij} - p_{ij}^{'}$, indicates the difference between the original and the distorted pixels. The 255^2 means the allowable pixel intensity in Equation (11). A typical value for PSNR in lossy image is from $30dB$ to $50dB$ for 8-bit depth, the higher the better. Structural SIMilarity (SSIM) [29] estimates the impact that changes image brightness, photo contrast, and other residual errors are identified as structural changes. The SSIM values is limited to a range between 0 and 1. If the SSIM value is close to 1, it means that the stego-image is similar to the cover image and high quality. The equation of SSIM is as follows:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2\mu_y^2 + c_1)(\mu_x^2\mu_y^2 + c_2)} \tag{13}$$

where, μ_x, μ_y are mean values of cover image (x) and stego-image (y), $\sigma_x, \sigma_y, \sigma_x^2, \sigma_y^2$, and σ_{xy} are standard deviation, variances and covariance of cover image and stego image, c_1, c_2, c_3 are constant values to avoid division by zero problem.

Table 1. PSNR and embedding capacity according to different thresholds T.

Images	T	Ou and Sun [19]		Bai and Chang's [20]		W Hong [23]		The proposed	
		EC	dB	EC	dB	EC	dB	EC	dB
Boats	5	129249	31.3506	64011	31.2928	149368	31.3203	166176	31.2846
Goldhill		53873	32.7028	21291	32.7076	73408	32.6373	100853	31.4917
Airplane		154545	31.7405	78477	31.6604	175203	31.7181	187268	31.2018
Lena		135089	33.1929	67400	33.1760	155889	33.1454	168498	33.1059
Peppers		100977	33.6253	48164	33.6888	121072	33.5682	138714	33.4999
Zelda		109585	35.7438	53096	35.8680	128346	35.6618	145526	35.5624
Average		113886	33.0593	55407	33.0656	133881	33.0085	151173	32.6911
Images	T	Ou and Sun [19]		Bai and Chang's [20]		W Hong [23]		The proposed	
		EC	dB	EC	dB	EC	dB	EC	dB
Boats	10	160913	31.0204	82644	31.1508	186330	30.9774	201272	30.9316
Goldhill		127409	31.6842	64721	32.2382	150349	31.6372	169667	30.7147
Airplane		194897	31.3173	102018	31.4875	221824	31.2796	232682	30.8072
Lena		193249	32.3724	101530	32.7961	220205	32.3277	231077	32.2792
Peppers		200369	32.2246	106357	32.9962	227287	32.1842	236657	32.1617
Zelda		212753	33.6013	113380	34.7771	240483	33.5452	247727	33.5333
Average		164799	31.8075	85108	32.5743	190170	31.7623	219847	31.7379
Images	T	Ou and Sun [19]		Bai and Chang's [20]		W Hong [23]		The proposed	
		EC	dB	EC	dB	EC	dB	EC	dB
Boats	20	205809	29.5664	110433	30.7138	233709	29.5557	243122	29.5228
Goldhill		212193	29.1224	117286	31.2597	240231	29.1121	249392	28.5724
Airplane		226977	30.1906	122037	31.1269	256110	30.1792	262802	29.8300
Lena		233697	30.7508	126667	32.2383	264366	30.7356	269432	30.7074
Peppers		240977	30.691	131514	32.4373	271132	30.6750	276077	30.6495
Zelda		253841	31.5579	138904	33.9124	284866	31.5299	288527	31.4777
Average		221128	29.9278	124474	31.9481	250207	29.9158	264892	30.1266

Table 1 represents the comparison of EC and PSNR between the proposed scheme and existing methods, i.e., Ou and Sun[19], Bai and Chang[20], and W Hong et al [23]. Especially, we compare the performance between our scheme and the existing methods using 6 images when the threshold value $T(= b - a)$ is 5, 10, and 20. Evaluation of EC and PSNR based on threshold values is necessary for objectivity and fairness for comparative evaluation of performance. That is, the data measured under the same threshold value may be evaluated as a more meaningful comparison. One important point for EC and PSNR is that there is a trade-off between the two assessments. That is, if EC is higher, the PSNR is reduced, and vice versa. But, in case the proposed method has very good performance, the deviation of trade-off may be not large. The EC of our proposed method is efficient in respect of EC as 151173 bits when $T = 5$.

In Table 1, Bai & Chang's PSNR (= 33.0656dB) is measured higher than that (= 32.6911dB) of our proposed method. Here, the EC of Bai & Chang [20] is 55407 bits, and the EC of our method is 151173 bits. In the end, our proposed method shows the capability to conceal about 95000 bits more than the Bai & Chang.

If the threshold T and EC are given for faithfull measurement, the PSNR of our the proposed method may show highest. This is because the size of hidden bits affects PSNR. Apparently, a relative good method has high values both of PSNR and EC. When $T = 10$, we can see that our method's EC (= 219847 bits) is largest. Both W Hong [23] and our proposed method have the same PSNR ($31.7dB$), which is $0.1dB$ lower than that of Ou & Sun's method [19]. However, in this case as well, when considering the amount of EC, our method outperforms the other two methods.

When $T = 20$, the PSNR of the proposed method is higher than the previous two methods (Ou & Sun[19], and W Hong[23]) and EC of our method has the highest performance. It can be seen from the

simulation results in Table 1 that the proposed method has 140000 bits higher than Bai & Chang[20] in terms of EC.

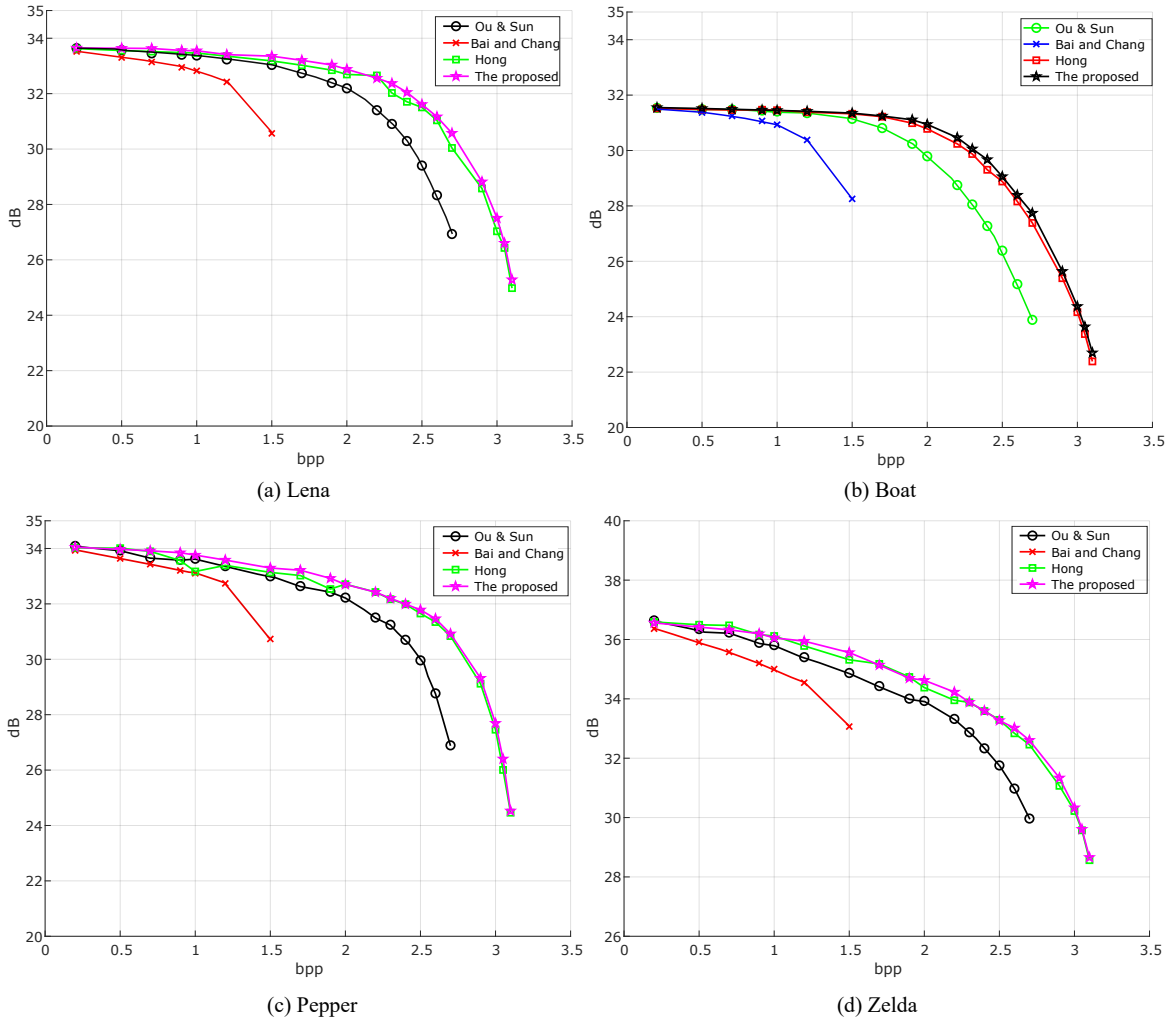

Figure 8. Performance comparisons of the proposed method and other related methods (i.e., Ou and Sun, Bai and Chang, Hong) based on four images: (a) Lena, (b) Boat, (c) Pepper, and (d) Zelda.

Figure 8 shows the performance comparison between our proposed method and the existing methods, where we measured the PSNRs while increasing the capacity of the secret bits from 20000 bits to 310000 bits in four images, ((a) Lena, (b) Boat, (c) Pepper, and (d) Zelda), by using the proposed and existing methods.

We proposed a way to improve the performance of Bai & Chang's method [20], and as shown in Figure 8, it is enough confirmed that our proposed method is superior to existing methods. On the other hand, our proposed method shows almost the same performance as W Hong's method[23], but it can be confirmed that the performance of our proposed scheme is slightly better. Ou & Sun's method [19] is superior to Bai & Chang's method [20], but the performance is not as high as that of our proposed method.

AMBTC has a difficulty in hiding enough data, because it is a compressed code, and unlike conventional grayscale images, it is not easy to exploit high embedding capacity by the constraint of compressed pixels. It is difficult to improve DH performance for images with many complex blocks, and if we exploit many pixels for high data concealment, the image quality may deteriorate.

In Figure 8, we can see that the EC of Bai & Chang's method[20] is very low. That is because this method is possible to hide only 6 bits of data while inverting up to 2 pixels in each bitmap. Thus, there

is a limit to embed enough data in the *trio*'s bitmaps. Since this method could not allow to conceal many secret bits for the threshold T of the same condition, it shows a relatively high PSNR . After all, that's why it can't mean that this method is superior to other methods. If we would like to increase the number of secret bits even at the expense of PSNR, it is able to use a way of increasing the size of the threshold T. However, it may be often the case that the PSNR gets worse without increasing enough the number of hidden bits than expected. For example, when $T \leq 4$, the three methods except Bai & Chang's method can hide about 130,000 bits, while PSNRs are slightly reduced. For such a large data to embed, they exploited the DBS method in respect of bitmap BM equally.

Bai & Chang's method must increase the T value in order to conceal 130000 bits of data, and as a result, the errors are accumulated rapidly. Since Bai & Chang's method[20] uses up to 4LSB for data concealment, the size of the error is inevitably increased. Since our proposed method uses up to 3LSB and the frequent count of 3LSB is also not very high, the negative effect on image quality is less than that of Bai & Chang's method[20]. In the end, we proved that the proposed method has a better optimization performance than Bai & Chang's method[20].

Table 2. Performance comparison of the proposed scheme and previous schemes (using 120000 bits)

Images	Ou and Sun[19]		Bai and Chang[20]		W Hong[23]		The proposed	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Boats	31.3506	0.6433	30.3823	0.6675	31.3846	0.6828	31.4158	0.7298
Goldhill	31.7499	0.6942	31.1779	0.7345	32.2203	0.7279	31.4158	0.7642
Airplane	31.7282	0.6614	31.1754	0.6526	31.9034	0.7042	31.3737	0.7305
Lena	33.2362	0.6614	32.4231	0.6870	33.3540	0.7094	33.4090	0.7566
Peppers	33.3636	0.6556	32.7389	0.6966	33.3905	0.7081	33.5822	0.7316
Zelda	35.4041	0.6936	34.5442	0.7177	35.7839	0.7335	35.9442	0.7778
Average	32.8054	0.6683	32.0736	0.6943	33.0061	0.7110	32.8568	0.7484

Table 2 shows an experiment to compare PSNR and SSIM after concealing the same amount of data (120000 bits) in the cover image for a more objective performance check and reliable comparison. The SSIM of the proposed method shows the highest value. On the other hand, in the case of PSNR, W Hong's method [23] shows a high average. In fact, PSNR only quantifies the quality of reconstructed or damaged images in relation to facts. it is not compared to HVS. A better evaluation criterion would be SSIM, which evaluates the structure of the image. The SSIM of the reconstructed image for the ground image is always 1, and if the value is close to 1, you can see that the image quality is excellent. Therefore, we can see that our proposed method is superior to the existing methods in terms of SSIM. Our proposed method is able to obtain better performance by creating a look up table in order to obtain more optimal values than W Hong's method [23] .

5. Conclusion

In this paper, we introduced a DH method that applies DBS and optimized HC (7,4) to AMBTC compressed grayscale images. The basic unit of AMBTC is *trio*, which consists of two quantization levels and one bitmap, and is represented by $trio(a, b, bitmap)$. Therefore, AMBTC is a trio set, and the proposed DH method is applied to each block of an image. The proposed method may have different final performance results depending on the characteristics of each block. Therefore, we divide every block into two groups (smooth blocks and complex blocks) and apply the proposed method. The distinction whether a block is a smooth block or a complex block is determined by the difference between the two quantization levels of the block. That is, if a difference ($|a - b|$) between two quantization levels is smaller than or equal to the threshold T, it is categorized as a smooth block. When hiding data in a complex block with a difference higher than threshold T, a relatively higher MSE errors increase compared to a smooth block. Therefore, it is important in terms of DH to distinguish the block. In other words, the smoother the blocks are, the more they help to maintain the image

quality while concealed more data. In this paper, our proposed method achieved optimization level by HC (7,4) based on look up table. As a result, it was shown through experiments that our proposed scheme surpasses the performance of Hong's method. Experimental results show that the proposed scheme provides high EC while suppress the loss of quality of the cover image.

Author Contributions: Conceptualization, C.N. Yang and C. Kim; methodology, D. Shin and C. Kim; validation, C.N. Yang and C. Kim; formal analysis, C.N. Yang; writing - original draft preparation, C. Kim and L. Leng; funding acquisition, D.Shin, C.N. Yang, C. Kim, and L. Leng. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Natural Science Foundation of China (61866028), Key Program Project of Research and Development (Jiangxi Provincial Department of Science and Technology)(20171ACE50024), Foundation of China Scholarship Council (CSC201908360075), Open Foundation of Key Laboratory of Jiangxi Province for Image Processing and Pattern Recognition (ET201680245, TX201604002). This research was supported in part by the Ministry of Science and Technology (MOST), under Grant MOST 108-2221-E-259-009-MY2 and 109-2221-E-259-010. This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by (2015R1D1A1A01059253),(2018R1D1A1B07047395) and was supported under the framework of international cooperation program managed by NRF (2016K2A9A2A05005255).

References

1. Chang, C.C., Li, C.T., Shi, Y.Q. Privacy-aware reversible watermarking in cloud computing environments, *IEEE Access* **2018**, *6*, 70720–70733.
2. Byun, S.W., Son, H.S., Lee, S.P. Fast and robust watermarking method based on DCT specific location, *IEEE Access* **2019**, *7*, 100706 –100718.
3. Kim, C., Yang, C.N. Watermark with DSA signature using predictive coding, *Multimedia Tools and Applications* **2015**, *74(14)*, 5189–5203.
4. Kim, H.J., Kim, C., Choi, Y., Wang, S., Zhang, X. Improved modification direction methods, *Computers & Mathematics with Applications* **2010**, *60(2)*, 319–325.
5. Kim, C. Data hiding by an improved exploiting modification direction, *Multimedia Tools and Applications* **2010**, *69(3)*, 569–584.
6. Petitcolas, F. A. P., Anderson, R. J., Kuhn, M. G. Information hiding-a survey, *Proc. IEEE* **1999**, *87*, 1062–1078.
7. Kim, C., Shin, D., Yang, C.N., Chen, Y.C., Wu, S.Y. Data hiding using sequential hamming + k with m overlapped pixels, *KSII Trans. Internet Info.* **2019**, *13(12)*, 6159–6174.
8. Mielikainen, J. LSB matching revisited, *IEEE Signal Proc. Let.* **2006**, *13*,285–287.
9. Chan, C.K., Cheng, L.M. Hiding data in images by simple LSB substitution, *Pattern Recognit.* **2004**, *37*,469–474.
10. C. Kim, D. Shin, B.G. Kim, et, al. Secure medical images based on data hiding using a hybrid scheme with the Hamming code, *J. Real-Time Image Process.* **2018**, *14*,115–126.
11. Leng, L., Li, M., Kim, C., Bi, X. Dual-source discrimination power analysis for multi-instance contactless palmprint recognition, *Multimedia Tools and Applications* **2017**, *76(1)*,333–354.
12. Leng, L., Zhang, J. S., Khan, M. K., Chen, X., Alghathbar, K. Dynamic weighted discrimination power analysis: a novel approach for face and palmprint recognition in DCT domain, *International Journal of the Physical Sciences* **2010**, *5(17)*, 2543–2554.
13. Deeba, F., Kun, S., Dharejo, F.A., Zhou, Y. Wavelet-Based Enhanced Medical Image Super Resolution, *IEEE Access* **2020**, *8*, 37035–37044.
14. Delp, E., Mitchell, O. Image compression using block truncation coding, *IEEE Trans. Commun.* **1979**, *27(9)*, 1335–1342.
15. Lema, M.D., Mitchell, O.R. Absolute moment block truncation coding and its application to color images, *IEEE Trans. Commun.* **1984**, *COM-32*, 1148–1157.
16. Kumar, R., Singh, S., Jung, K.H. Human visual system based enhanced AMBTC for color image compression using interpolation, *2019 6th International Conference on Signal Processing and Integrated Networks (SPIN) Mar* **2019**, *385*, 903–907.
17. Hong, W., Chen, T.S., Shiu, C.W. Lossless steganography for AMBTC compressed images, *2008 Congress on Image and Signal Processing* **2008**, 13–17.
18. Chuang, J.C.; Chang, C.C. Using a simple and fast image compression algorithm to hide secret information, *Int. J. Comput. Appl.* **2006**, *28*, 329–333.

19. Ou, D., Sun, W. High payload image steganography with minimum distortion based on absolute moment block truncation coding, *Multimed. Tools Appl.* **2015**, *74*, 9117–9139.
20. Bai, J., Chang, C.C. High payload steganographic scheme for compressed images with Hamming code, *Int. J. Network Secur.* **2016**, *18*, 1122–1129.
21. R. Kumar, D. S. Kim and K. H. Enhanced AMBTC based data hiding method using hamming distance and pixel value differencing, *J. Inf. Secur. Appl.* **2019**, *47*, 94–10.
22. Chen, J., Hong,W., Chen, T.S., Shiu, C.W. Steganography for BTC compressed images using no distortion technique, *Imaging Sci. J.* **2013**, *58*, 177–185.
23. Hong, W. Efficient data hiding based on block truncation coding using pixel pair matching technique, *Symmetry* **2018**, *10(2)*, 1–8.
24. Hong, W., Chen, T.S. A novel data embedding method using adaptive pixel pair matching, *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 176–184.
25. Huang, Y.H., Chang, C.C., Chen, Y.H. Hybrid secret hiding schemes based on absolute moment block truncation coding, *Multimed. Tools Appl.* **2017**, *76*, 6159–6174.
26. Chen, Y.Y., Chi, K.Y. Cloud image watermarking: High quality data hiding, and blind decoding scheme based on block truncation coding, *Multimed. Syst.* **2019**, *25*, 551–563.
27. Malik, A., Sikka, G., Verma, H.K. An AMBTC compression-based data hiding scheme using pixel value adjusting strategy, *Multidimens. Syst. Signal Process.* **2018**, *29*, 1801–1818.
28. Lin, J., Weng, S., Zhang, T., Ou, B., Chang, C.C. Two-Layer Reversible Data Hiding Based on AMBTC Image With (7, 4) Hamming Code, *IEEE Access* **2019**, *8*, 21534–21548.
29. Sampat, M. P., Wang, Z., Gupta, S., Bovik, A.C., Markey, M.K. Complex wavelet structural similarity: a new image similarity index, *IEEE Transactions on Image Processing* **2009**, *18(11)*, 2385–2401.