entropy

MDPI

Article

High-Payload Data-Hiding Method for AMBTC Decompressed Images

Jung-Yao Yeh [1], Chih-Cheng Chen [2], Po-Liang Liu [1,*] and Ying-Hsuan Huang [3]

1. Graduate Institute of Precision Engineering, National Chung Hsing University, No. 250, Kuo-Kuang Road, Taichung City 402, Taiwan; honor0425@gmail.com
2. Department of Computer Science and Engineering, National Chung Hsing University, No. 250, Kuo-Kuang Road, Taichung City 402, Taiwan; salu.chen@gmail.com
3. Aeronautical Systems Research Division, National Chung-Shan Institute of Science and Technology, Taichung 40722, Taiwan; ying.hsuan0909@gmail.com

* Correspondence: pliu@dragon.nchu.edu.tw

Received: 19 December 2019; Accepted: 23 January 2020; Published: 25 January 2020

Abstract: Data hiding is the art of embedding data into a cover image without any perceptual distortion of the cover image. Moreover, data hiding is a very crucial research topic in information security because it can be used for various applications. In this study, we proposed a high-capacity data-hiding scheme for absolute moment block truncation coding (AMBTC) decompressed images. We statistically analyzed the composition of the secret data string and developed a unique encoding and decoding dictionary search for adjusting pixel values. The dictionary was used in the embedding and extraction stages. The dictionary provides high data-hiding capacity because the secret data was compressed using dictionary-based coding. The experimental results of this study reveal that the proposed scheme is better than the existing schemes, with respect to the data-hiding capacity and visual quality.

Keywords: data hiding; AMBTC; steganography; stego image; dictionary-based coding; pixel value adjusting

1. Introduction

The concealment of information within media files is commonly used in various applications. This process originates from the hieroglyphs used in the Egyptian civilization. Other cultures, such as the Chinese culture, adopted a more physical approach to hide messages by writing them on silk or paper, rolling the material into a ball, and covering the material with wax to communicate political or military secrets. Data hiding is nearly indispensable for every aspect in our daily lives whether for good or evil intentions.

Due to its rapid growth, the Internet has recently become far more popular than traditional media. Data is accessible by everyone due to the popularity of the Internet. Therefore, possessing the capabilities of detecting copyright violations, forgery, and fraud is crucial. Many techniques, such as steganography and cryptography, have been designed to secure digital data. The difference between steganography and cryptography is as follows: In cryptography (e.g., chaos-based encrypted systems, secure pseudo-random number generator, etc. [1]) users are aware that there is an encrypted image, but they cannot efficiently decode the encrypted image unless they know the proper key. In steganography, users can easily decode the encrypted message, but most people do not notice that there is an encrypted message. In this study, we focused on the techniques used for hiding data in images.

The schemes present for hiding data in an image can be broadly classified into two categories, irreversible data-hiding schemes [2–4] and reversible data-hiding schemes [5–7]. In the irreversible data-hiding schemes, a recipient can extract the secret information. However, the original image cannot be recovered after extracting the secret information. In the reversible data-hiding schemes, the hidden data can be extracted from the image, and the original image can be retrieved from a stego image without any distortion. Two factors affect a data-hiding scheme, i.e., visual quality and embedding payload. A high-quality data-hiding scheme should not raise any suspicions of adversaries. Therefore, this type of scheme should provide low image distortion and high payload.

To decrease the size of a digital image file or accelerate the transmission, a data-hiding scheme that employs a compressed image should be developed. Many compressed file formats have been proposed, such as JPEG and JPEG2000. Wang et al. [8] proposed a lossless data-hiding method for JPEG images by using adaptive embedding. Lee et al. [9] proposed a scheme in which a secret image was compressed using JPEG2000 and then, embedded in the cover image by using tri-way pixel value differencing. Nevertheless, both JPEG and JPEG2000 need complicated computation for image compression and decompression.

Another popular technique used for image compression is block truncation coding (BTC) [10]. Compared with the methods using JPEG and JPEG2000, BTC is a simple and efficient encoding technique that is used for image compression. Therefore, the computation cost is relatively low when a data-hiding scheme is based on BTC.

Lema and Mitchell [11] proposed the absolute moment BTC (AMBTC) technique to improve the compression performance of BTC. When AMBTC is used, the first absolute moment is maintained with the mean. To exploit the advantages of AMBTC compression, we proposed an AMBTC decompressed image-based data-hiding scheme by using a pixel adjusting strategy.

The basic idea of the proposed study is to preliminarily calculate the probability of secret data and then select the best codebook for embedding the secret data. The secret data are embedded into the AMBTC compression image by modifying the pixel value according to the codebook. Experimental results reveal that the proposed scheme is almost better than the current state of the art method in terms of the hiding capacity.

The remainder of the paper is organized as follows: Section 2 describes the relevant approaches such as the BTC and AMBTC techniques for data hiding; Section 3 describes the implementation flow of the scheme proposed for data hiding; Section 4 discusses several experimental results are presented, and some issues; and finally, Section 5 specifies the conclusions and future work.

2. Related Works

Before describing the high data-hiding capacity of the proposed scheme, we review the AMBTC technique and some recently developed AMBTC-based data-hiding methods.

2.1. Absolute Moment Block Truncation Coding (AMBTC)

BTC, a simple and efficient block-based lossy image compression method, is used for grayscale images. Although the BTC method provides a low compression ratio, it is a popular image compression method because of its low complexity with respect to both computation and implementation. In the BTC algorithm, an image X, with $M \times N$ pixels, is divided into nonoverlapping blocks. Each block has $n \times n$ pixels, and the pixel values can be different. The mean and standard deviation of each pixel value are calculated before conducting BTC. In general, two statistical characteristics change from one block to another.

The hardware implementation of BTC is challenging because the square and square root functions are involved. To resolve this problem, AMBTC [11] was proposed as a type of BTC. The AMBTC uses the first absolute moment and mean values instead of using the standard deviation value. The main difference between AMBTC and BTC is that the mean and standard deviation values of a block are preserved in BTC. However, in AMBTC, the high mean and low mean values of a block are preserved.

As in BTC, an image X is divided into nonoverlapping blocks with $n \times n$ pixels also in the AMBTC encoding phase. For each block, the mean $\overline{x}$ and the absolute moment α of the pixel values are calculated using

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

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

Note that $m = n \times n$.

The pixel value x_i is compared with the mean $\overline{x}$ for composing a bit plane for each pixel in the block. If the pixel value x_i is greater than the mean $\overline{x}$, then x_i is denoted as 1. Otherwise, the pixel value is denoted as 0. The equation of bit representation is

$$P_i = \begin{cases} 1, \text{ if } x_i > \overline{x}, \\ 0, \text{ otherwise.} \end{cases} \tag{3}$$

In the AMBTC-compressed block reconstruction phase, the block reconstruction is conducted using two values L_m and H_m. The values of L_m and H_m are computed using

$$L_m = \overline{x} - \frac{m a}{2(m-q)}, \tag{4}$$

$$H_m = \overline{x} + \frac{m a}{2q}. \tag{5}$$

In Equations (4) and (5), q represents the number of pixels with pixel values greater than $\overline{x}$. Thus, a compressed block has two values L_m and H_m, where L_m is the low mean value and H_m is the high mean value. To reconstruct a block, the pixels that are assigned the value of 0 in the bit plane are replaced with the L_m value, and the pixels assigned the value of 1 in the bit plane are replaced with the H_m value by

$$x'_i = \begin{cases} H_m, \text{ if } p_i = 1, \\ L_m, \text{ if } p_i = 0. \end{cases} \tag{6}$$

2.2. Related Work of BTC and AMBTC Based Data Hiding Schemes

BTC has significantly low complexity and requires less memory. Therefore, BTC is a good scheme for data hiding. Chuang and Chang proposed a data-hiding scheme for BTC-compressed images for embedding data in the bitmaps of smooth blocks to obtain an improved image quality. There are two steps in in the embedding process of the scheme proposed by Chuang and Chang. Initially, a cover image is compressed into blocks by using BTC for calculating two quantized data and the bit plane corresponding to each block. Finally, the secret data is embedded into the bitmaps of the predefined smooth blocks that satisfy the following equation: $H_m - L_m < \mathit{Threshold}$. The smooth blocks were selected because bit replacement in these bit planes causes a slight distortion in the BTC image. In the extraction process of the scheme proposed by Chuang and Chang [12], the difference $H_m - L_m$ has to be first calculated. If $H_m - L_m < \mathit{Threshold}$, then the secret bit in the bit plane p'_i is extracted. However, in this scheme, the stego image quality degrades significantly as the threshold values increases.

Hong et al. [13] proposed a reversible data-hiding scheme based on bit plane flipping according to the corresponding secret bit. In the embedding process, each image block was compressed using AMBTC-compressed codes to determine whether the block is embeddable or not. If $L_m < H_m$, then the block is considered embeddable. Otherwise, the block is considered non-embeddable. For each embeddable block, if the secret bit is 1, then the bit plane p_i is flipped to $\overline{p}_i$, where $\overline{p}_i$ is not an operator. If the secret bit is 0, then no operation is required. In the extraction process, if $L_m > H_m$ in p'_i, then the secret in p'_i is 1. Otherwise, the secret bit in p'_i is 0. The scheme presented by Hong et al. does not hide data in blocks with $L_m = H_m$. Therefore, Chen et al. [14] proposed a reversible data-hiding method to improve the scheme by Hong et al. The AMBTC-compressed block that has $L_m = H_m$ is a smooth area, which is considered unnecessary bit plane information. Thus, the secret bit can be embedded all bits in the bit plane block to improve the scheme by Hong et al.

Li et al. [15] introduced a data-hiding scheme by using the histogram shifting technique on BTC-compressed mean tables for further improving the hiding capacity, while maintaining the quality of the BTC-compressed image. The hiding scheme comprises two main steps. The first step is based on the bit plane flipping method that hides secret bits by swapping the high mean and low mean values. In the second step, histogram shifting is conducted on the resulting mean tables after swapping. This scheme requires no additional data in the stego code stream. Therefore, very low distortion is observed in this scheme after data embedding, and the security of the embedded data is enhanced. However, this technique cannot provide a sufficient data-hiding capacity and requires overhead information to record a histogram.

Lin et al. [16] proposed a technique to explore the redundancy in a block of AMBTC-compressed images to determine whether the block is embeddable. If the secret bits and bit plane combined in the block has more than three different cases, the block is marked as an embeddable block. Four disjoint sets were created using this technique of embeddable blocks for embedding data using different combinations of the mean value and its standard deviation.

Ou and Sun [17] proposed a data-hiding scheme with minimum distortion based on AMBTC. In this scheme, a predefined threshold is used to determine if a block of the AMBTC-compressed codes is a smooth or complex block in which data are embedded. If an AMBTC-compressed block $H_m - L_m < Threshold$, then the block is considered a smooth block. All bit planes in smooth blocks are used to embed data by replacing the bits of the block with secret data bits. The two quantization levels in the smooth block are then recalculated to reduce distortion in the image. In the complex blocks, a proportion of secret bits were concealed by exchanging the order of two quantization levels and toggling the bit plane. By performing this method, the payload can be increased without any distortion. Both smooth and complex blocks can be used to embed data in an AMBTC-compressed block. Therefore, the payload of this scheme was obviously enhanced.

Malik et al. [18] modified the AMBTC compression technique for embedding secret data. In their method, one-bit plane is converted to two-bit planes that can attain better image quality and high capacity. Although this scheme has high visual quality and high payload, it causes permanent distortion to the original AMBTC code and requires overhead information. Malik et al. [19] proposed an AMBTC compression-based data-hiding scheme by using the pixel value adjusting strategy. In this technique, the stream of secret bits was converted to digits with a base of three. Then, the pixel values of the AMBTC-compressed block are modified, at the most by one, to hide secret data. This scheme could maintain a balance between the hiding capacity and quality of a stego image.

As discussed above, data hiding by using the AMBTC technique is an issue worthy of more research. In this study, we extended the work of Malik et al. [19] to embed a larger amount of secret data. In the next section, the proposed scheme is discussed.

3. Proposed Scheme

Figure 1 shows the flowchart of our application. First, one monitoring image on the unmanned aerial vehicle was compressed because the transmitting volume of wireless network is limited. When the command post or chief's car receives the compression codes, they are decoded as the decompressed image. In addition, they embed secret data into the reconstructed image, thereby cheating hackers and avoiding attacks. Finally, the headquarters can extract secret data and recover the decompressed image.

The main aim of the study is to present a data-hiding scheme with high data-hiding capacity and high image quality. In the scheme, secret data is hidden in an AMBTC decompressed image. The AMBTC decompressed image is losslessly reconstructed and the secret data, then, is losslessly revealed from the reconstructed image. The AMBTC encoding procedures are described in Section 2.

Before embedding the secret data, the cover image must be compressed using the AMBTC algorithm. In other words, the proposed scheme uses the AMBTC decompressed image to embed the secret data.

The proposed scheme involves three stages: In the first stage, an appropriate encoding and decoding dictionary is found. The dictionary is used in the second stage to embed the data. In the third stage, the secret data is extracted. The details of the proposed scheme are presented in Figure 1.

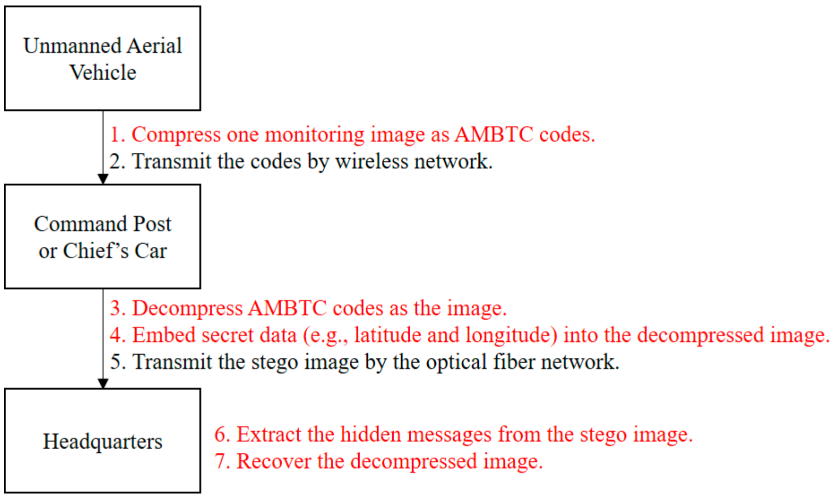

Unmanned Aerial Vehicle
1. Compress one monitoring image as AMBTC codes.
2. Transmit the codes by wireless network.
Command Post or Chief's Car
3. Decompress AMBTC codes as the image.
4. Embed secret data (e.g., latitude and longitude) into the decompressed image.
5. Transmit the stego image by the optical fiber network.
Headquarters
6. Extract the hidden messages from the stego image.
7. Recover the decompressed image.

Figure 1. Flowchart of our applications.

3.1. Finding a Unique Decodable Dictionary

A binary secret sequence S comprises 0 and 1 values and is denoted as $S = \{s_1,\ s_2, \ldots, s_N\}$, where $s_i \in \{0, 1\}$ for $i = 1 \sim N$. Consider the dictionaries D_1, D_2, D_3, and D_4 formed using K subsets of S, that is, $S_{p1}, S_{p2}, \ldots,\ S_{pK}$. Different image quality is obtained due to the different dictionaries. Thus, we can calculate each probability of symbol S_p in S. The amount of information in each symbol I_a can be represented by

$$I_a = -log_2\left(\mathrm{pr}\left(S_{pk}\right)\right). \tag{7}$$

Then, the average information per symbol interval is $H\left(S_{pk}\right)$ and can be represented by

$$H\left(S_{pk}\right) = -\sum_{k=1}^{n} pr\left(S_{pk}\right) log_2\left(pr\left(S_{pk}\right)\right). \tag{8}$$

The average information $H\left(S_{pk}\right)$ is referred to as the entropy. The dictionary with the smallest entropy H should be selected because it can achieve the best encoding benefit. The following explains why the dictionary of the smallest entropy is used: Assume that there is only one symbol's type in the whole secret sequence. In other words, the other types never occur. In this case, the entropy is equal to 0, i.e., $H\left(S_{pk}\right) = 0$. Afterwards, the specific symbols are replaced by the absolute minimum value "0", thereby controlling the distortion level in the data embedding phase. Consequently, the proposed method selects the dictionary of the smallest dictionary.

An example is used to explain the above procedure. Assume the secret sequence $S = \{0011101111001101100100000010011010\}$. In dictionary D_1 listed in Table 1, the secret sequence is represented as $S = \{001, 11, 01, 11, 10, 01, 10, 11, 001, 000, 001, 001, 10, 10\}$ for easy readability. According to D_1, the total number of information is 12.4670 and the average information $H\left(S_{pk}\right)$ per symbol at S is 2.1570. In dictionary D_2, which is listed in Table 2, the secret sequence can be represented as $S = \{00, 11, 10, 11, 11, 00, 11, 011, 00, 10, 00, 00, 10, 011, 010\}$. According to D_2, the total number of information is 12.1451 and the average information $H\left(S_{pk}\right)$ per symbol at S is 2.2264. The third and fourth dictionaries are constructed in the same manner, and their entropies values are listed in Tables 3 and 4, respectively. Obviously, the entropy of D_1 is the smallest among all the dictionaries. Therefore, we used D_1 to encode the secret sequence.

Table 1. Total number of data was 12.4670 with an entropy H of 2.1570 in the first dictionary D_1.

Symbol (S_p)	Freq. (S_p Count)	Amount of Information
000	1	3.4594
001	4	1.4594
01	2	3.0444
10	4	2.0444
11	3	2.4594

Table 2. Total number of data was 12.1451 with an entropy H of 2.2264 in the second dictionary D_2.

Symbol (S_p)	Freq. (S_p Count)	Amount of Information
00	5	1.7225
010	1	3.4594
011	2	2.4594
10	3	2.4594
11	4	2.0444

Table 3. Total number of data was 12.0751 with an entropy H of 2.2405 in the third dictionary D_3.

Symbol (S_p)	Freq. (S_p Count)	Amount of Information
00	3	2.4150
01	3	2.4150
100	3	1.8301
101	1	3.4150
11	4	2

Table 4. Total number of data was 12.5602 with an entropy H of 2.1726 in the fourth dictionary D_4.

Symbol (S_p)	Freq. (S_p Count)	Amount of Information
00	5	1.7225
01	3	2.4594
10	1	4.0444
110	3	1.8745
111	2	2.4594

Subsequently, the symbols in the selected dictionary are encoded further to obtaining the embedded digits. According to the rule of thumb of data encoding, S_p with the maximum occurrence frequency was encoded as the absolute minimum value. By contrast, S_p with the lowest occurrence frequency was encoded as the absolute maximum value. Consequently, S_p was sorted based on the occurrence frequency, and then, its sorted index was encoded to obtain the adjusting pixel values P_v, i.e.,

$$p_v = \begin{cases} -\left\lfloor \frac{\textit{Sort index}}{2} \right\rfloor, \text{ if Sort index is an odd number,} \\ \left\lfloor \frac{\textit{Sort index}}{2} \right\rfloor, \text{ otherwise.} \end{cases} \tag{9}$$

The following example is used to explain how to encode most symbols as smaller digits, as listed in Table 5. The occurrence frequencies of two symbols, "001" and "10", are 4, which are higher than those of other symbols. According to Equation (9), the symbol "001" is encoded as the absolute minimum value "0". Moreover, the symbol "10" is encoded as the second smallest value "1". The remaining symbols are encoded in the same manner.

Table 5. Dictionary example presenting the pixel value adjusting method.

Symbol (S_p)	Freq. (S_p Count)	Sorted Index	Adjusting Pixel Values (P_v)
000	1	5	−2
001	4	1	0
01	2	4	2
10	4	2	1
11	3	3	−1

3.2. Embedding Stage

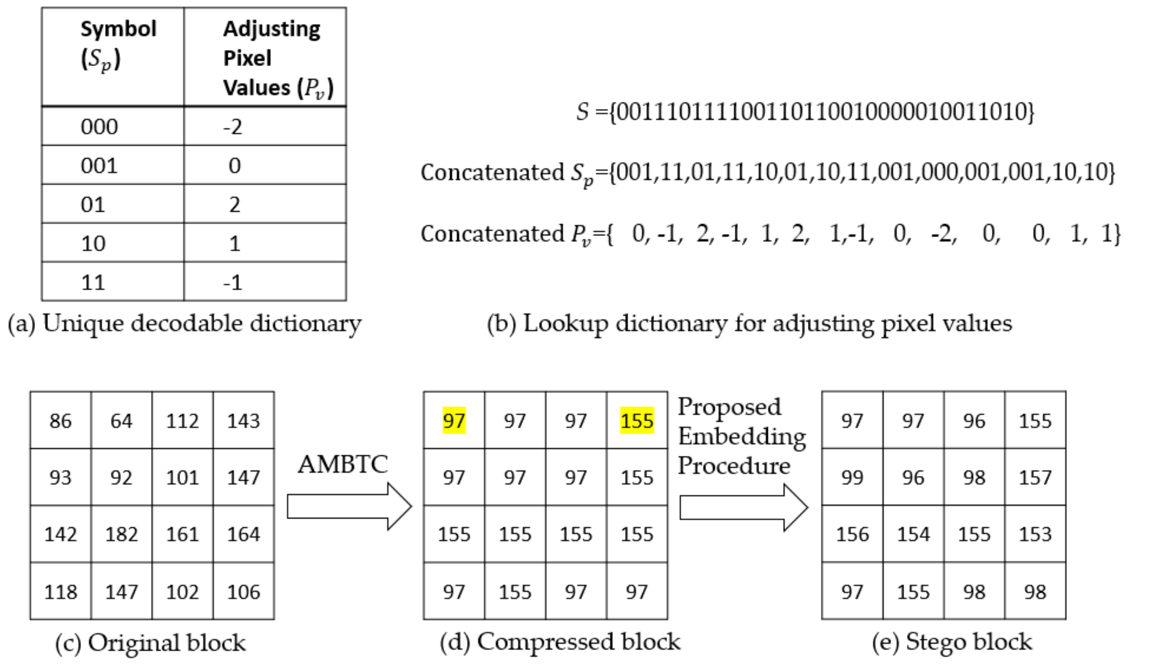
The AMBTC decompressed blocks b_i in the original AMBTC decompressed image T are sequentially scanned. If the difference between H_m and L_m is smaller than 4, then the block is considered a non-embeddable block. Otherwise, the block is an embeddable block. In the first embeddable block, the binary representation of the ID number of the selected dictionary is embedded into the least significant bits (LSBs) of the second Hm and the second L_m. Note that the number of dictionaries is four, thus the two LSBS can effectively represent the ID number. The other blocks are then used to embed the secret data by using the pixel value adjusting strategy.

In each embeddable block, the first H_m and the first L_m are defined as non-embeddable pixels, which are used as the reference information of data extraction and image recovery. For the embeddable block b_i, each pixel x'_i except the first H_m and the first L_m is increased by the adjusting pixel values P_v, that is, $x''_i = x'_i + P_v$. The difference between maximum P_v and minimum P_v in the difference D is equal to 4. It implies that the distortion of pixels is low. The embedding pseudocode is shown in Algorithm 1 as follows:

Algorithm 1: Embedding pseudocode

```
foreach $AMBTC - compressed\ block\ b_i in\ T$ do
    if $H_m - L_m \leq 4$ then      /* non-embeddable block */
        Do nothing;
    else if $b_i$ is first embeddable block then
        embedding dictionary $D$ number;
    else
        foreach pixel $x'_i$ in $b_i$ do
            find adjusting pixel values $P_v$ in $D$;
            $x''_i = x'_i + P_v$;
        end
    end
end
```

Figure 2 displays the embedding example in which = {0011101111001101100 10000010011010}. Figure 2a presents the appropriate dictionary D found in Section 3.1. This dictionary was used to encode the secret sequence. After looking up the dictionary D, S is divided into many subsets S_p, as shown in Figure 2b. These subsets are mapped using the adjusting pixel values P_v, which are just the embedded value.



Symbol (S_p)	Adjusting Pixel Values (P_v)
000	-2
001	0
01	2
10	1
11	-1

(a) Unique decodable dictionary

S ={001110111100110110010000010011010}

Concatenated S_p={001,11,01,11,10,01,10,11,001,000,001,001,10,10}

Concatenated P_v={ 0, -1, 2, -1, 1, 2, 1,-1, 0, -2, 0, 0, 1, 1}

(b) Lookup dictionary for adjusting pixel values

86	64	112	143
93	92	101	147
142	182	161	164
118	147	102	106

(c) Original block

97	97	97	155
97	97	97	155
155	155	155	155
97	155	97	97

(d) Compressed block

97	97	96	155
99	96	98	157
156	154	155	153
97	155	98	98

(e) Stego block

Figure 2. Example illustrating the proposed embedding stage.

To embed these values, the original block must be compressed and decompressed using the AMBTC algorithm, as shown in Figure 2c. After using the AMBTC algorithm, the AMBTC decompressed block can be reconstructed using a low mean value L_m of 97 and a high mean value H_m of 155, as shown in Figure 2d. Both the first L_m and first H_m are non-embeddable pixels and are marked with yellow color for easy readability. They are used as reference information of data extraction and image recovery. For the AMBTC decompressed block, the pixel, except the first H_m and the first L_m, is increased by the adjusting pixel values P_v to obtain the stego pixel. Figure 2e shows the stego block.

If the overflow or underflow problem occurs in any altered pixel of the block, then all of the pixels in the corresponding block remain unchanged. In other words, the block cannot be used to embed any secret bit. In addition, the proposed method records the ID number of the non-embeddable block to discriminate between the embeddable block and the non-embeddable block.

3.3. Extraction Stage

In the extraction stage, the secret data is extracted from the stego image T'. Moreover, T' can be used to recover the original AMBTC decompressed image T. The details of the procedures are listed as follows:

1. Scan the stego AMBTC decompressed block b'_i in T' sequentially. If the difference between H_m and L_m is smaller than 4, then this block is considered a non-embeddable block. Otherwise, it is an embeddable block.

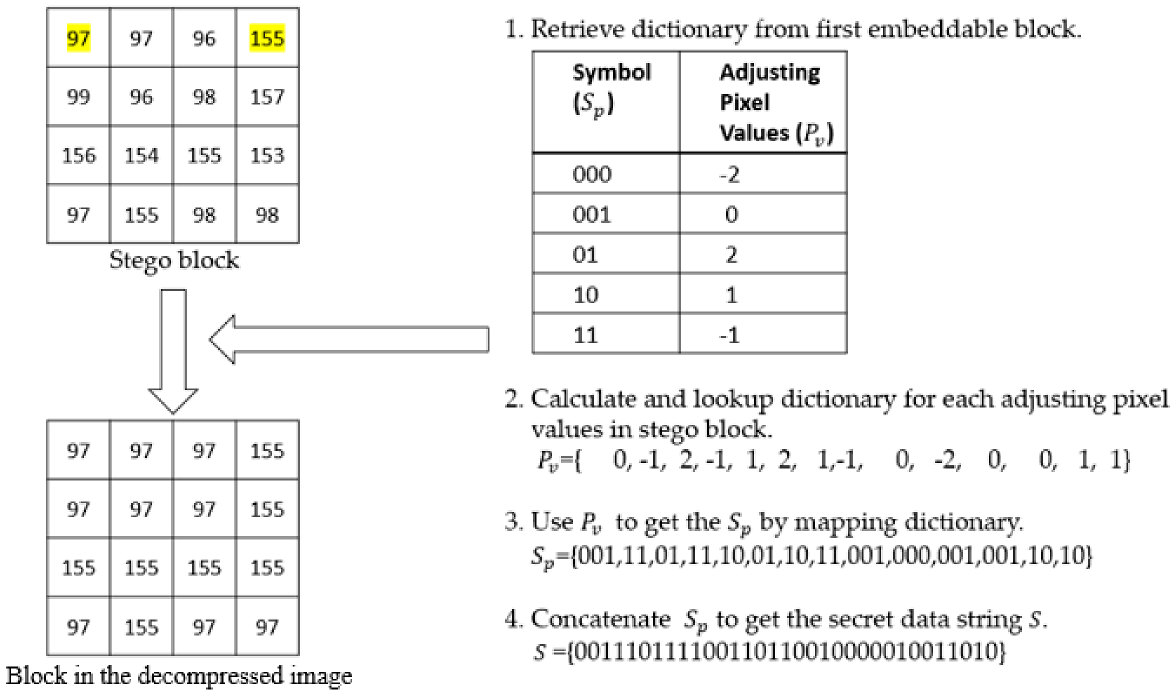
2. Retrieve the ID number of the selected dictionary D from the first embedded block. In the first embedded block, both the LSBs of the second Hm and the second Lm are extracted, i.e., binary representation of the ID number of the selected dictionary. Therefore, the proposed method can reconstruct the selected dictionary. In addition, both the LSBs are replaced by the first Hm and the first Lm, thereby recovering the original decompressed pixel.

3. Calculate the adjusting pixel values by using $P_v = x''_i - H_m$ or $P_v = x''_i - L_m$ for each embeddable block b'_i. After obtaining P_v, we can look up the dictionary D to obtain the symbol S_p. After concatenating all S_p, we obtain the secret sequence S and recover the original AMBTC decompressed image T. The extraction and recovery pseudocode are shown in Algorithm 2.

Algorithm 2: Extraction and recovery pseudocode

```
foreach block $b'_i$ in $T'$ do
    if $H_m - L_m \le 4$ then        /* non-embeddable block */
        Do nothing;
    else if $b_i$ is first embeddable block then
        get the dictionary $D$ number;
    else
        foreach pixel $x''_i$ in $b'_i$ do
            get the first $H_m$ and $L_m$;
            if $x''_i = H_m$ then
                $P_v = x''_i - H_m$;
            else
                $P_v = x''_i - L_m$;
            end
            find the symbol $S_p$ in $D$;
            $S = S + S_p$;
        end
    end
end
```

Figure 3 illustrates the extraction and recovery example. First, the dictionary is retrieved from the first embeddable block. Second, the adjusting pixel values are calculated as $P_v = x''_i - H_m$ or $P_v = x''_i - L_m$. Third, P_v is mapped with the dictionary values to obtain S_p. Finally, S_p is concatenated for obtaining the secret sequence S and the AMBTC decompressed block.



97	97	96	155
99	96	98	157
156	154	155	153
97	155	98	98

Stego block

1. Retrieve dictionary from first embeddable block.

Symbol (S_p)	Adjusting Pixel Values (P_v)
000	-2
001	0
01	2
10	1
11	-1

2. Calculate and lookup dictionary for each adjusting pixel values in stego block.
   P_v={ 0, -1, 2, -1, 1, 2, 1,-1, 0, -2, 0, 0, 1, 1}

3. Use P_v to get the S_p by mapping dictionary.
   S_p={001,11,01,11,10,01,10,11,001,000,001,001,10,10}

4. Concatenate S_p to get the secret data string S.
   S ={0011101111100110110010000010011010}

97	97	97	155
97	97	97	155
155	155	155	155
97	155	97	97

Block in the decompressed image

Figure 3. Example illustrating the proposed extraction stage.

4. Experimental Results and Discussion

Some experimental cover images were tested to demonstrate the efficiency of the proposed scheme. In the experiments, the proposed scheme was verified using the following six test cover images: airplane, boat, lena, mandrill, peppers, and sailboat. As shown in Figure 4, all the images had the same size of 512 × 512 pixels with 256 grayscales, and the features of the images were diverse. The block size of the image presented in the AMBTC format was 4×4 pixels. A random binary sequence generated using a MATLAB (R2018a) function was used in the experiments as the secret sequence, where our secret data are the same as the secret data of the related works [15–17,19]. Note that each bit in the sequence has equal probability of being 0 or 1.

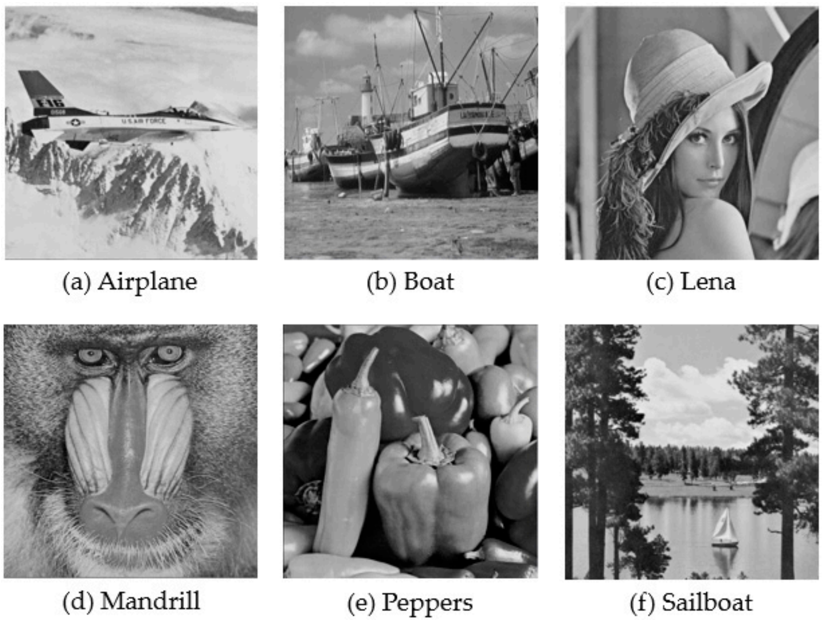


Figure 4. Test cover images.

The proposed scheme was evaluated and compared with the aforementioned schemes in terms of two performance measures, i.e., hiding capacity and peak signal-to-noise ratio (PSNR). The hiding capacity can be defined as the number of secret data bits that can be hidden into a cover image. The PSNR is an objective measure used for determining the visual quality of an image. The higher the PSNR of a stego image, the better its visual quality is. The rule of thumb is that when the PSNR is higher than 30 dB, the human eyes cannot easily perceive the difference between the cover image and the stego image. PSNR is defined by

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

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(x_{ij} - x'_{ij}\right)^2, \tag{11}$$

where x_{ij} and x'_{ij} are the original and stego grayscale pixel values located at (i, j), respectively.

To present the superiority of the proposed scheme, we compared our scheme with the schemes presented by Li et al. [15], Lin et al. [16], Ou and Sun [17], and Malik et al. [19], as shown in Table 6. The proposed scheme achieved the highest data-hiding capacity for all five images except for the airplane image. The data-hiding capacity of the pixel value adjusting strategy was determined using the number of smooth blocks. If there are many smooth zones in a cover image, non-embeddable blocks are observed in abundance in the image. Moreover, the pixel value adjusting strategy used in the proposed scheme is modified at the most by 2, whereas the strategy used in the scheme proposed by Malik et al. is modified at the most by 1. Therefore, compared with the scheme presented by Malik et al., our scheme has a higher number of non-embeddable blocks in the airplane image. Non-embeddable blocks can be observed in black in Figure 5a,b. This is the main factor that causes the hiding capacity of the scheme proposed by Malik et al. to be better than that of the proposed scheme for the airplane image. For the other five images, the hiding capacity of our scheme is better than that of the scheme presented by Malik et al. by an enhancement value in the range of 10.13% to 29.89%. The hiding capacity of our scheme is better than the schemes proposed by Li et al., Lin et al., and Ou and Sun. Thus, we conclude that our scheme is better than the existing AMBTC- and BTC-based data-hiding schemes, in terms of the hiding capacity.

Table 6. Comparison between hiding capacity and PSNR for different images for the proposed scheme and other AMBTC- and BTC-based schemes.

Method	Performance	Airplane	Boat	Lena	Mandrill	Peppers	Sailboat
Proposed	Hiding capacity (bits)	338,836	495,582	437,577	515,836	478,591	479,407
	PSNR (dB)	32.0908	31.0397	33.0562	26.9348	33.0087	29.7857
Malik et al. (2018)	Hiding capacity (bits)	397,147	397,380	397,348	397,105	397,057	397,466
	PSNR (dB)	31.9018	31.0926	33.102	26.9474	33.304	29.8081
Ou and Sun (2015)	Hiding capacity (bits)	223,039	217,264	234,004	141,919	238,969	219,169
	PSNR (dB)	30.71	29.54	30.87	26.02	31.59	28.61
Lin et al. (2013)	Hiding capacity (bits)	261,984	262,096	262,112	262,144	261,984	262,064
	PSNR (dB)	31.64	30.32	33.05	26.0047	32.2021	28.8049
Li et al. (2011)	Hiding capacity (bits)	17,659	16,580	16,789	16,880	17,264	16,990
	PSNR (dB)	30.18	31.83	31.05	27	32.35	28.43

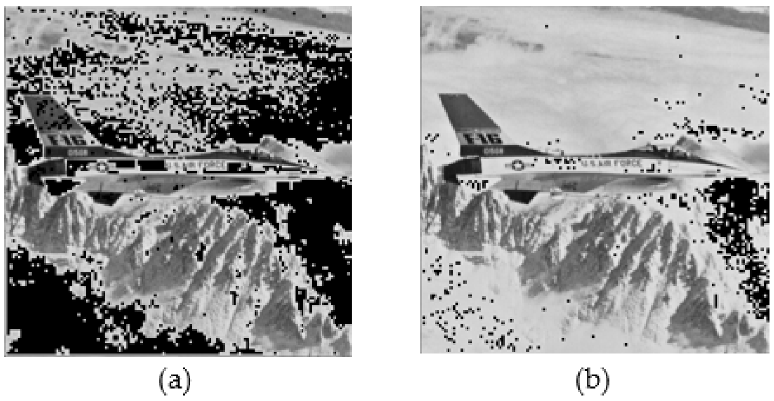


Figure 5. Non-embeddable blocks in "airplane.": (**a**) Proposed scheme and (**b**) scheme proposed by Malik et al.

Table 7 lists the comparison between the method by Malik et al. and the proposed method in terms of structural similarity index (SSIM). As mentioned above, the SSIM value of the method by Malik et al. is greater than that of the proposed method because the proposed method embeds more secret data. In other words, the maximum hiding capacity of the proposed method is higher than that of the method by Malik et al.

Table 7. Comparison between hiding capacity and SSIM for different images for the proposed scheme and the other method by Malik et al.

Image	Malik et al.'s Method		Proposed Method	
	Hiding Capacity (bits)	SSIM	Hiding Capacity (bits)	SSIM
Airplane	397,147	0.947	338,836	0.9447
Boat	397,380	0.918	495,582	0.915
Lena	397,348	0.937	437,577	0.933
Mandrill	397,105	0.886	515,836	0.885
Peppers	397,057	0.931	478,591	0.927
Sailboat	397,466	0.915	479,407	0.912

The PSNR is the other factor for evaluating performance of a hiding scheme. Table 6 presents that the PSNR of our scheme is better than the schemes proposed by Li et al., Lin et al., and Ou and Sun. For the airplane stego image, the proposed scheme has a better PSNR but a weaker hiding capacity than the scheme proposed by Malik et al., because our scheme has a higher number of non-embeddable blocks than the scheme by Malik et al. Note that a non-embeddable block maintains the image quality but decreases the hiding capacity. For the other five stego images, the PSNR obtained using the proposed scheme is weaker than that obtained using the scheme proposed by Malik et al. However, because the PSNR difference is less than 0.29 dB for the five stego images, they would not be distinguishable by human vision due to such negligible differences. By contrast, the hiding capacities are significantly increased by a value in the range of 10.13% to 29.89% for the other five stego images. This implies that a tradeoff exists between the PSNR and hiding capacity when the pixel value adjusting strategy is used. The visual quality of the proposed scheme is observed to be above the average value of that of the baseline schemes.

5. Conclusions

A high-capacity data-hiding scheme was proposed in this study for an AMBTC-compressed image. The proposed scheme has many more properties than only high capacity. In this scheme, the dictionary-based coding scheme and the pixel value adjusting strategy were combined to increase the hiding capacity and attain a satisfactory visual quality. Experimental results reveal that the proposed scheme is better than the existing AMBTC-based data-hiding schemes in terms of the hiding capacity. Moreover, the visual quality of the proposed scheme is better than that of baseline schemes. In the future, we should combine the method by Liao et al. [20] with the proposed method to discriminate the image smoothness, thereby enhancing the hiding capacity. In addition, we should try to add the concept of partition strategy [21] into the proposed method to embed more secret data into the color images.

Author Contributions: Conceptualization, C-C.C.; data curation, J-Y.Y.; funding acquisition, P-L.L.; investigation, J-Y.Y.; methodology, J-Y.Y. and C-C.C.; supervision, P-L.L.; validation, J-Y.Y. and Y-H.H.; writing—original draft, J-Y.Y.; writing—review & editing, P-L.L.

Funding: This research was funded by the Ministry of Science and Technology (Taiwan), No. MOST 108-2221-E-005-001.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, C.; Zhang, Y.; Xie, E.Y. When an Attacker Meets a Cipher-image in 2018, A Year in Review. *J. Inf. Secur. Appl.* **2019**, *48*, 1–9. [CrossRef]
2. Mielikainen, J. LSB Matching Revisited. *IEEE Signal Process. Lett.* **2006**, *13*, 285–287. [CrossRef]
3. Zhang, X.; Wang, S. Efficient Steganographic Embedding by Exploiting Modification Direction. *IEEE Commun. Lett.* **2006**, *10*, 781–783. [CrossRef]
4. Hong, W.; Chen, T. A Novel Data Embedding Method Using Adaptive Pixel Pair Matching. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 176–184. [CrossRef]
5. Tian, J. Reversible Data Embedding Using a Difference Expansion. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 890–896. [CrossRef]
6. Peng, F.; Li, X.; Yang, B. Adaptive Reversible Data Hiding Scheme Based on Integer Transform. *Signal Process.* **2012**, *92*, 54–62. [CrossRef]
7. Tai, W.L.; Yeh, C.M.; Chang, C.C. Reversible Data Hiding Based on Histogram Modification of Pixel Differences. *IEEE Trans. Circuits Syst. Video Technol.* **2009**, *19*, 906–910.
8. Wang, K.; Lu, Z.M.; Hu, Y.J. A High Capacity Lossless Data Hiding Scheme for JPEG Images. *J. Syst. Softw.* **2013**, *86*, 1965–1975. [CrossRef]
9. Lee, Y.P.; Lee, J.C.; Chen, W.K.; Chang, K.C.; Su, I.J.; Chang, C.P. High-Payload Image Hiding with Quality Recovery Using Tri-Way Pixel-Value Differencing. *Inf. Sci. (Ny).* **2012**, *191*, 214–225. [CrossRef]

10. Delp, E.; Mitchell, O. Image Compression Using Block Truncation Coding. *IEEE Trans. Commun.* **1979**, *27*, 1335–1342. [CrossRef]
11. Lema, M.; Mitchell, O. Absolute Moment Block Truncation Coding and Its Application to Color Images. *IEEE Trans. Commun.* **1984**, *32*, 1148–1157. [CrossRef]
12. 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.
13. Hong, W.; Chen, T.S.; Shiu, C.W. Lossless Steganography for AMBTC-Compressed Images. In Proceedings of the 2008 Congress on Image and Signal Processing, Sanya, Hainan, China, 27–30 May 2008; pp. 13–17.
14. Chen, J.; Hong, W.; Chen, T.S.; Shiu, C.W. Steganography for BTC Compressed Images Using no Distortion Technique. *Imaging Sci. J.* **2010**, *58*, 177–185. [CrossRef]
15. Li, C.H.; Lu, Z.M.; Su, Y.X. Reversible Data Hiding for BTC-Compressed Images Based on Bitplane Flipping and Histogram Shifting of Mean Tables. *Inf. Technol. J.* **2011**, *10*, 1421–1426. [CrossRef]
16. Lin, C.C.; Liu, X.L.; Tai, W.L.; Yuan, S.M. A Novel Reversible Data Hiding Scheme Based on AMBTC Compression Technique. *Multimed. Tools Appl.* **2015**, *74*, 3823–3842. [CrossRef]
17. 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. [CrossRef]
18. Malik, A.; Sikka, G.; Verma, H.K. A High Payload Data Hiding Scheme Based on Modified AMBTC Technique. *Multimed. Tools Appl.* **2017**, *76*, 14151–14167. [CrossRef]
19. 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. [CrossRef]
20. Liao, X.; Qin, Z.; Ding, L. Data Embedding in Digital Images Using Critical Functions. *Signal Process. Image Commun.* **2017**, *58*, 146–156. [CrossRef]
21. Liao, X.; Yu, Y.; Li, B.; Li, Z.; Qin, Z. A New Payload Partition Strategy in Color Image Steganography. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, in press. [CrossRef]