applied sciences

Article

Separable Reversible Data Hiding in Encrypted AMBTC Images Using Hamming Code

Cheonshik Kim

Department of Computer Engineering, Sejong University, Seoul 05006, Korea; mipsan@sejong.ac.kr

Abstract: Data hiding is a field widely used in copyright, annotation, and secret communication for digital content, and has been continuously studied for more than 20 years. In general, data hiding uses the original image as a cover image to hide data, but recently, the research area has expanded to research on improving the security and privacy protection of image contents by encrypting the image. This research is called separable reversible data hiding in an encrypted image (SRDH-EI). In this paper, we proposed a more efficient SRDH-EI method based on AMBTC. AMBTC can guarantee very good network transmission efficiency for applications that do not require a particularly high image quality because the compression time is short and calculation is simple compared to other existing compression methods. The SRDH-EI method presented here divides AMBTC into non-overlapping 4×4 blocks and then performs image encryption on them. Thereafter, data can be hidden using a Hamming code for each block. The proposed method has an advantage in that the quality of the cover image and the hiding capacity can be adjusted by appropriately using the value T of the difference between the two quantization levels. The experimental results proved the efficiency and superiority of our proposed model.

Keywords: reversible data hiding (RDH); absolute moment block truncation coding (AMBTC); hamming code; separable RDH in an encrypted image (SRDH-EI)

check for updates

Citation: Kim, C. Separable Reversible Data Hiding in Encrypted AMBTC Images Using Hamming Code. *Appl. Sci.* **2022**, *12*, 8225. https://doi.org/10.3390/app12168225

Academic Editor: Krzysztof Koszela

Received: 23 July 2022
Accepted: 12 August 2022
Published: 17 August 2022

1. Introduction

Images require metadata that describe their existence and summarize the underlying image data. That is, details on the author, creation date, modification date, file size, etc. The purpose of this paper was to support image security by encrypting the compressed image with AMBTC, one of the compression methods of gray images, and creating a model that supports Data Hiding (DH) [1–3]. DH can be used for covert communication, authentication and copyright based on a variety of digital media (e.g., digital content such as images, video, audio, etc.). The receiving end is used to extract data from the encrypted image and support the recovery of the original cover image. The data hidden in the image should not be detectable by an attacker.

Reversible DH(RDH) [4,5] is a kind of DH. It is possible to recover the original image after extracting data from the marked image, and various RDHs have been proposed so far. RDH can be classified into three specialities. That is, pixel extension (DE) [5–7] histogram shift (HS) [8], prediction error extension (PE) [9]. DE is a method of first doubling the difference between two adjacent pixels, and then hiding one bit in the LSB (least significant bit). In HS, the zero and peak points in the histogram of the cover image are used to hide secret metadata. PE is a method of hiding data by using the difference in the prediction error between the original pixel and the predicted pixel.

On the other hand, image encryption [10,11] is a method used to protect digital information by encrypting the image so that only the user who knows the encryption key can restore the image, thereby protecting the privacy of the image. In particular, in order to share secret images, owners, assistants, and administrators must encrypt the image and then distribute it using a trusted service provider (SP). For example, hospital administrators can embed personal information, authentication, and diagnostic opinion data into encrypted X-ray, CT, or MRI images. In this case, encrypted image-based RDH combines the advantages of RDH and image encryption. It can be useful as it is a compromise between advantages.

If the original image is encrypted and delivered to the SP, the SP cannot estimate the image before it is encrypted. Because the SP does not have the encryption key, it cannot recover this image. The role of the SP is to hide additional data in the encrypted image and transmit the encrypted image to the receiver. After decoding the image on the receiver side, the recovered cover image is lossless, and metadata can also be extracted on the receiver side. Since Zhang [12] introduced an RDH encrypted image (RDH-EI)-based RDH using the fluctuation principle, the extended RDH-EI has been actively studied. RDH-EI has a similar process to RDH, except that the cover image is encrypted before hiding the metadata. In fact, RDH-EI can be utilized for various applications such as authentication, copyright and personal security protection [13–18].

Zhang [19] demonstrated that data extraction and image recovery are separable and that hidden metadata can be extracted without errors. The process is called Separable RDHEI (SRDH-EI) [19–21]. That is, the receiving side can use EK (encryption key) and DK (data secret key) to decrypt and extract the encrypted image regardless of the application order. In [20], Qian et al. proposed n-ary SRDH-EI using HS and solved the original image recovery problem, which is the problem of Zhang's method. Yin et al. [21] proposed SRDH-EI with a general payload and error-free data extraction by introducing multi-granularity permutation.

Yin et al. (2018) [22] first proposed an AMBTC-based RDH-EI method. The first step is to encrypt the upper and lower averages of triples in the AMBTC-compressed video using a stream cipher. The prediction error histogram correction technique can then be used to insert additional data into the redundant space.

Wang et al. (2019) [23] proposed an embedding method based on an AMBTC (absolute moment block truncation coding) compressed image. They used an adaptive variable N-bit bit-plane truncated image embedding method to embed secret data in each block. Here, the secret data are extracted from the receiver side, and the stored peaks and zeros are retrieved to restore the original AMBTC image. Since the histogram of the image is used for data hiding, the amount of hidden data can vary greatly depending on the characteristics of the image. Su et al. (2022) [24] utilized the redundant space derived from the quantization level to insert a secret message. Data hiding was then performed using a labeling strategy. It achieved a higher embedding capacity than most existing methods.

Meanwhile, Block Truncation Coding (BTC) [25] is one of the available 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 that many researchers are interested in DH based on Absolute Moment BTC (AMBTC) [26], which was developed from BTC recently.

Chuang and Chang [27] proposed a DH method in which each block of the bitmap is divided into a smooth block and a complex block, and then the bitmap of the smooth block is replaced with a secret bit. A block is divided into a smooth block and a complex block with the difference value (threshold value: T) between the two quantization levels representing the block, thereby controlling the quality of the image. That is, if the threshold value T is lowered, the image quality is improved, but the data hiding capacity is reduced. If the size of the threshold value increases, the image quality can be lowered and the hidden data capacity can be increased. Ou and Sun [28] proposed a method by which to adjust image distortion by adjusting two quantization levels, but the original image is required for recomputation. Chen et al. [29] proposed a lossless DH method using two quantization level orders.

In this paper, we proposed the SRDH-EI method for AMBTC. AMBTC has the advantage of not having a high computational complexity, which can be useful for real-time video and non-critical image/video processing. In addition, low-resolution compressed images are suitable for efficient use in low-power communication based on WSN (Wireless

Sensor Networks). The proposed SRDH-EI method consists of the following three steps: image encryption, data hiding, data extraction, and image recovery. AMBTC consists of a bitmap and two quantization levels and is encrypted with a bitmap and a stream cipher with random bits for the two quantization levels. Here, the bit operation, such as XOR, is often used as the encryption/decryption operation. The goal of the proposed model was to achieve the scenario of SRDH-EI. Data hiding uses a Hamming code to hide the message in an encrypted bitmap. Data hiding using the Hamming code has the advantage of good data hiding efficiency and can minimize errors occurring in the cover image. As a result, the image quality of the decrypted image is very good. By conducting simulations, we demonstrated that this method achieves high embedding capacity, good image quality, and error-free hidden bit restoration.

The remainder of this paper is structured as follows. Section 2 describes related studies. Section 3 describes and discusses SRDH-EI. Section 4 presents the experimental results. Finally, Section 5 provides the conclusion.

2. Related Works

2.1. Absolute Moment Block Truncation Coding

AMBTC (Absolute Moment Block Truncation Coding) [26] is a type of BTC [25] that provides a faster compression processing speed and better image quality. A block consists of two quantization levels and a bitmap, and one block is compressed by preserving the moment. The two quantization values represent a high mean and a low mean for the pixel values of each block. Before the gray image is compressed with AMBTC, the image is first divided into non-overlapping sized blocks. That is, assuming that the variable corresponding to the block size set is $k \times k$, the block size may be (2×2), (4×4), (6×6), and (8×8). The mean of each block is as follows.

$$\bar{g} = \frac{1}{k \times k} \sum_{i=1}^{k^2} g_i \tag{1}$$

In Equation (1), g_i is the i-th pixel in the block. Pixels constituting each block are compressed into a bitmap. That is, the pixels in the block are replaced by a '1' if the pixel is greater than the mean value $\bar{g}$, and a '0' otherwise. Bitmap blocks are divided into the following two groups: '1' and '0'. Bitmap $b_i = \{b_{i1}, b_{i2}, \ldots, b_{ik\times k}\}$ is generated as a result of Equation (2).

$$b_i = \begin{cases} 1, & if\ (g_i \geq \bar{g}), \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

In order to recover a bit-plane compressed into a bitmap into a gray image, two quantization levels representing each block are needed. The method of obtaining two quantization levels in each block is provided in Equations (3) and (4), where t and $k^2 - t$ represent the number of '0' and the number of '1', respectively.

$$Q_1 = \left\lfloor \frac{1}{t} \sum_{g_i < \bar{g}}^{k^2} g_i \right\rfloor \tag{3}$$

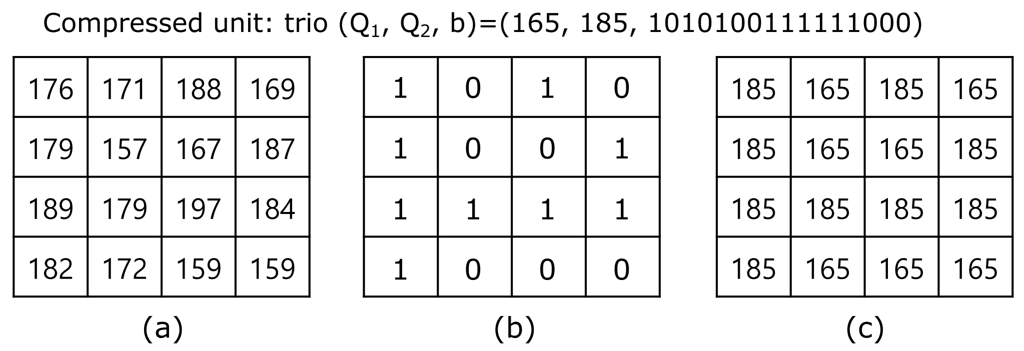
$$Q_2 = \left\lfloor \frac{1}{k^2 - t} \sum_{g_i \geq \bar{g}}^{k^2} g_i \right\rfloor \tag{4}$$

Here, Q_1 and Q_2 may be used for decoding AMBTC. An image block can be compressed into two quantization levels Q_1, Q_2 and a bitmap M and cab be expressed as a *trio* (Q_1, Q_2, b). According to the value of the bitmap, Q_1 and Q_2 values are decoded by applying Equation (5). Finally, the compressed image can be decoded into a gray image.

$$g_i = \begin{cases} Q_1, & \text{if } (b_i = 0), \\ Q_2, & \text{otherwise} \end{cases} \tag{5}$$

The size of the block is inversely proportional to the image quality and compression ratio. If the block size increases, the compression rate increases but the image quality deteriorates. If the block size decreases, the compression rate decreases, and the image quality improves. We used a block size of 4×4 in this study. The reason is that it is considered as a size that complies with the appropriate level of compression ratio and PSNR. If the block size is increased to (8×8), the PSNR will decrease. If we reduce the block size to (2×2), PSNR will increase, but the number of pixels required to apply our proposed RDH will be insufficient.

Example 1. *As shown in Figure 1a is a block of the original image and the process is explained using this. If a block of gray images g = {176, 171, 188, 169, 179, 157, 167, 187, 189, 179, 197, 184, 182, 172, 159, 159}, the mean of this block is 106. Applying Equations (1)–(4), the trio of this block becomes trio*(Q_1, Q_2, M) = *(165, 185, 1010100111111000). Here, the mean* ($\bar{g}$) *is 106, the low mean* (Q_1) *is 165, and the high mean* (Q_2) *is 185. A bitmap (b) is constructed by assigning '1' to those values that are greater than the average and '0' to those that are not. Decoding using Equation (5) results in* $g' = \{185, 165, 185, 165, 185, 165, 165, 185, 185, 185, 185, 185, 185, 165, 165, 165\}$. *(c) is generated by the decoding process.*


Compressed unit: trio (Q_1, Q_2, b)=(165, 185, 1010100111111000)

(a)

176	171	188	169
179	157	167	187
189	179	197	184
182	172	159	159

(b)

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

(c)

185	165	185	165
185	165	165	185
185	185	185	185
185	165	165	165

Figure 1. An example of AMBTC: (**a**) an original block; (**b**) a bitmap block; (**c**) a reconstruction block.

2.2. Hamming Code

Hamming codes [30,31] have been widely used for error control in digital communications and data storage. It has interesting properties that make encoding and decoding tasks easier. When r is a non-negative integer that is a parity space dimension, the length of the codeword becomes $\mathcal{N} = 2^r - 1$ and the data bits become $\mathcal{K} = \mathcal{N} - r$. A single error-correcting linear block code with a minimum distance of 3 for all codewords was selected. Let x be a $\mathcal{K}$-bit information word. An $\mathcal{N}$-bit codeword y is generated using $y = xG$, where G is a $\mathcal{K}$-by-$\mathcal{N}$ generator matrix.

It is in fact the space of vectors $y \in \mathbb{F}_2^n$ such that $Hy = 0$ for the parity check matrix H . The parity check matrix for the HC(7,4) Hamming code is as follows

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{6}$$

Let us assume that the codeword y' has an error such as $e = (y \oplus y')$. In this case, we could correct one error $(e = y - y')$ from the codeword y' by using the syndrome $\eta = y' \cdot H^T$, where the syndrome denotes the position of the error in the codeword. As show in Equation (7), the error e can be obtained.

$$\begin{cases} \tilde{y} \cdot H^T = (e \oplus y) \cdot H^T = e \cdot H^T + y \cdot H^T \\ (y \cdot H^T) = (x \cdot G) \cdot H^T = x \cdot (G \cdot H^T) = 0) \\ = e \cdot H^T + 0 = e \cdot H^T \end{cases} \tag{7}$$

The following are check matrices for HC(15, 11) binary Hamming codes (Equation (8)).

$$H = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{8}$$

Example 2. *Suppose Alice wants to send codeword* $r = [1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0]$ *to Bob. Here, the codeword* r *contains an error in the second bit. That is, Bob receives codeword* $\tilde{r} = [1\,1\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0]$ *with an error. Bob computes the syndrome* η *and finds the error* e*. As a result, he obtains the codeword* $r = [1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0]$ *which has repaired the error.*

2.3. Yin et al.'s Scheme

Yin et al. (2018) [22] proposed a prediction error-based RDH technique in encrypted AMBTC compressed images. In this technique, for each *trio*, the two quantization levels Q_1 and Q_2 are encrypted with a bit-based XOR operation using the same random binary code. Bitmap b is encrypted in a similar way. Then, we can obtain the prediction error values by calculating the difference between the encrypted Q_1 and Q_2. The RDH procedure is as follows: First, the 16-bit secret message is replaced directly with the bitmap in the i-th block when Q_2 equals Q_1, and f^i is set to '1' at the same time. Otherwise f^i is set to '0'. Assume here that the two peak bins are P^1 and P^2, the two zero bins are Z^1 and Z^2, and $Z^1 < P^1 < P^2 < Z^2$. The secret message s can be embedded into Q_1 by changing Q_1 to Q_1' according to Equation (9):

$$Q_1' = \begin{cases} Q_1 + 1, \text{ if } (Z^1 < PE < P^1), \\ Q_1 + s, \text{ if } (PE = P^1 \text{ and } f \neq Q_1), \\ Q_1 - s, \text{ if } (PE = P^2 \text{ and } f \neq Q_1), \\ Q_1 - 1, \text{ if } (P^2 < PE < Z^2), \end{cases} \tag{9}$$

where PE is the prediction error value between two quantization levels in the i-th block. Here, some auxiliary data including two zero-bins, two peak-bins, and a location map are embedded together with the messages.

3. Proposed Scheme

In this section, we provide detailed step-by-step methods for the AMBTC-based SRDH-EI model. This model is broadly divided into the following three phases: image encryption, data hiding, data extraction, and image recovery. The diagram of the proposed model is briefly sketched in Figure 2. The image owner uses AMBTC and the image encryption key to transform AMBTC into an encrypted image. SP uses the encrypted AMBTC and data secret key to hide additional (meta)data in the encrypted image. The recipient who receives the image can extract data or recover the image from the encrypted image by using the data secret key and the image encryption key. In addition, two keys can be used to manage data extraction and image recovery (Figure 2).

3.1. AMBTC Image Encryption

In this section, we provide a detailed step-by-step explanation of the encryption process for the compressed image AMBTC. The method of compressing the original image with AMBTC is based on Section 2.1. It is assumed that the size of the image is $N \times N$ and the size of one block is 4×4.

Input: An AMBTC cover image I and key EK.
Output: An encrypted cover image I' with size $N \times N$.

Step 1: Read a $trio(Q_1, Q_2, M)$ from the cover image I, where Q_1 and Q_2 are the quantization levels and M is a bitmap $M = (m_1, m_2, \cdots, m_{15}, m_{16})$.

Step 2: The quantization levels are $Q_1 = (a_8, a_7 \ldots a_1)$ and $Q_2 = (b_8, b_7 \ldots b_1)$, where a_8 is the Most Significant Bit (MSB) of Q_1 and a_1 is the LSB of Q_1. Similarly, b_8 is MSB of Q_2 and b_1 is the LSB of Q_2. $r = (r_8, r_7 \ldots r_1)$ is pseudo-random binary generated by the encryption key EK. Encryption for two quantization levels Q_1 and Q_2 is obtained by using Equations (10) and (11).

$$\begin{cases} \tilde{a}_{i,j} = a_{i,j} \oplus r_{i,j}, \quad j = 0 \ldots 7 \\ \tilde{b}_{i,j} = b_{i,j} \oplus r_{i,j}, \quad j = 0 \ldots 7 \end{cases} \tag{10}$$

$$\begin{cases} Q_{1i} = \sum_{j=0}^{7} \tilde{a}_{i,j} \times 2^j \\ Q_{2i} = \sum_{j=0}^{7} \tilde{b}_{i,j} \times 2^j \end{cases} \tag{11}$$

Here, i is the index of the block and j is the subscript for the two quantization levels Q_1 and Q_2 converted into binary numbers.

Step 3: For a bitmap $M = (m_1, m_2, \ldots, m_{16})$ of $trio_i$, a pseudo-random binary number r is generated by the image encryption key EK. An encrypted bitmap block M' is obtained by applying the XOR operation between the bitmap and the pseudo random bits r (Equation (12)).

$$M' = m_j \oplus r_j, j = 1 \ldots 16 \tag{12}$$

Step 4: If i is not the last block, it moves to *Step 1* and repeats the given process. When the given procedure is completed, the encrypted AMBTC is finally generated.

When this process is completed, an encrypted cover image is finally created. Data hiding is then applied to the bitmap of each block of this encrypted cover image.

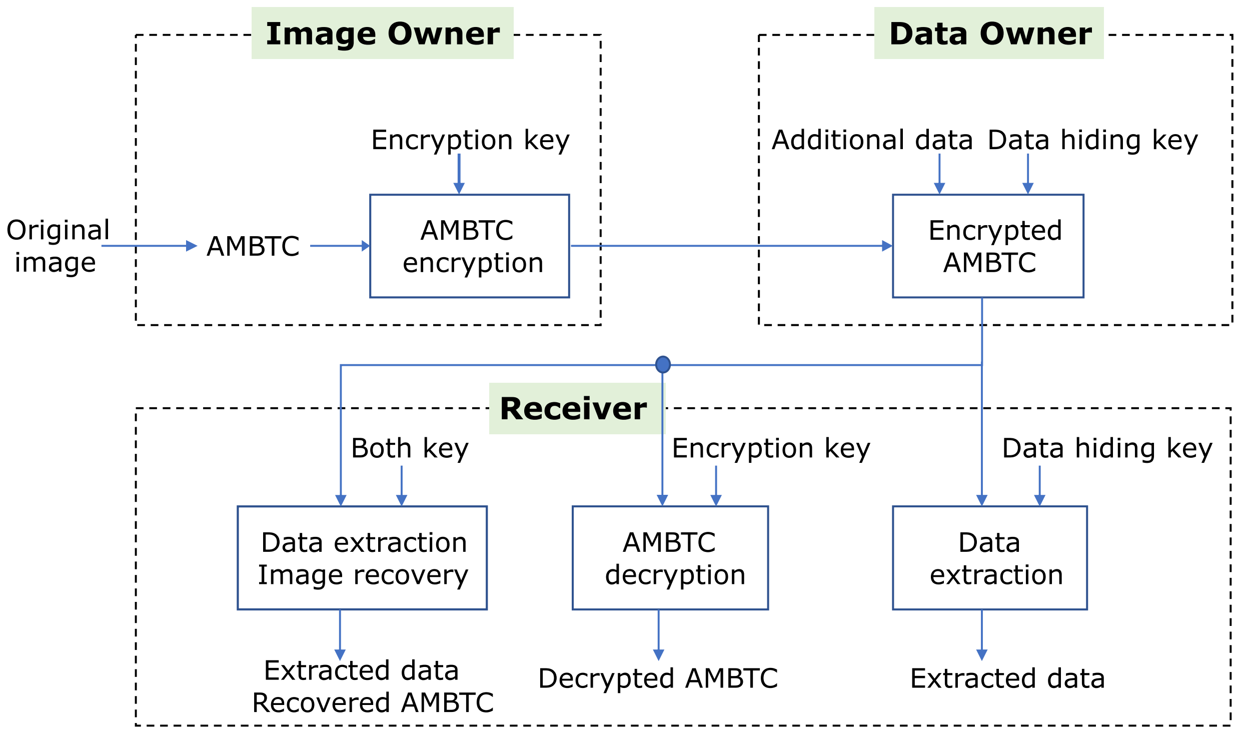


Figure 2. Block diagram of the proposed scheme.

3.2. Data Embedding

In this section, we provide a detailed step-by-step description of the process of hiding data in the encrypted AMBTC image.

Input: An encrypted AMBTC cover image I' and secret data $S = (s_1, s_2, \ldots, s_n)$.
Output: An encrypted marked image I''.

Step 1: Read a $trio(Q_1, Q_2, M')$ from the encrypted AMBTC image I', where Q_1 and Q_2 are the quantization levels and M' is a bitmap $M' = (m_1, m_2, \ldots, m_{15}, m_{16})$.

Step 2: The exclusive-or operation between the secret message bits s_i and the pseudorandom bits r_i are calculated using Equation (13), where r is determined via DK using a standard stream cipher.

$$s'_i = s_i \oplus r_i \tag{13}$$

Step 3: A bitmap $M' = \{m_1, m_2, \ldots, m_{15}\}$ is assigned to codeword c. For codeword c, the syndrome η is obtained by employing Equation (14).

$$\eta = (c \cdot H^T) \bmod 2 \tag{14}$$

Step 4: Exclusive-or operation is performed on the syndrome and the encrypted bits. That is, $\hat{\eta} = \eta \oplus s'_{i+4}$. If syndrome $\hat{\eta} = 0$, this means no operation. Otherwise, flip the $\hat{\eta}$-th coordinate $c_{\hat{\eta}}$ of c. For image restoration, the rows and columns of the changed location $c_{\hat{\eta}}$ are obtained using Equation (15), where rc is the value of the row and column, and 4 bit information of rc is added to the last bit of M'. The rc needs to be converted to binary before it is added last to M'. In the formula, cnt is first initialized to 0 and then the formula is applied.

$$rc = \sum_{i=1}^{4}\sum_{j=1}^{4}\begin{cases}(i\,||\,j), \; if(\hat{\eta} = cnt), \\ \qquad ++cnt\end{cases} \tag{15}$$

Step 5: After the operation, the codeword c is assigned to M'^{15}_1, i.e., $M'^{15}_1 = c$. Add payload coordinates rc after bitmap M'. That is, $M' = M' \,||\, rc$.

Step 6: Go to *Step 1* to continue the embedding processes until all messages have been embedded in the encrypted cover image.

If this process is repeatedly applied as often as the size of the image, marked images I'' are obtained. Using Example 3 below, we briefly reviewed the data hiding process.

Example 3. *As given in Figure 3, when the bitmap block M' = (1010010110100101) and the secret bit s = (1010), codeword c = (1010010110100101) is first constructed for data hiding. Next, the syndrome η is calculated. That is, $\eta = H \cdot c^T$ = (0000). Then, an exclusive-or operation is performed on the syndrome and the secret bit. The result is the decimal number 10. 4-bit data hiding is completed by inverting the number at position 10 of the bitmap. To restore the block, it is necessary to store the payload for the 10th position. The data hiding process is completed by adding (0110) at the end of the bitmap M'.*

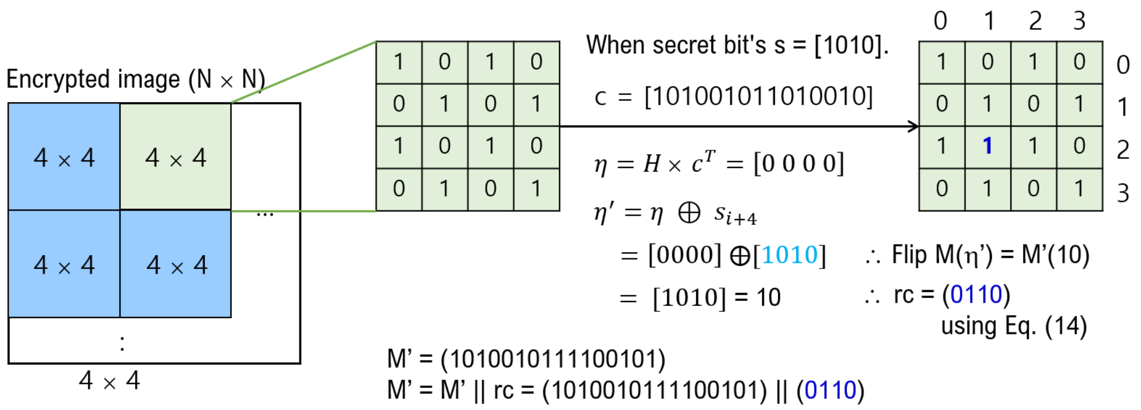


Figure 3. Block diagram of the data embedding example.

3.3. Data Extraction and Recovering Procedure

In this section, we introduce the process of data extraction and image restoration from the proposed model. If the receiving side has only the encryption key EK or data secret key DK, image restoration or extraction is possible. If both keys are present, image restoration and data extraction are possible. If we do not have the encryption key and data secret key, image restoration and data extraction cannot be performed. These features can not only protect images from attackers, but also hide data. We describe the process of extracting hidden information from the marked images I'' created in Section 3.2 and restoring the original image. The detailed process is described in stages.

Input: An encrypted AMBTC cover image I'' sized $N \times N$, and both keys EK and DK.
Output: A secret message and the cover image I sized $N \times N$ and $S' = (s_1, s_2, \ldots, s_n)$.

Step 1: Read the i-th $trio(Q_1, Q_2, M')$ from the encrypted cover image I'', where Q_1 and Q_2 are the quantization levels and M' is a bitmap $M' = (m'_1, m'_2, \ldots, m'_{15}, m'_{16})$.

Step 2: Assign the bitmap M'^{15}_1 to codeword c, i.e., $c = M'^{15}_1$.

Step 3: Calculate the syndrome $\hat{\eta} = H \cdot c^T$ for codeword c, and then assign the syndrome value $\hat{\eta}^T$ to the secret bits s'_{i+4}, i.e., $s'_{i+4} = \hat{\eta}^T$, $i = i + 4$. For the extracted 4-bit data, decoding is performed using r created by DK using a standard stream cipher. That is, $s_{i+4} = s'_{i+4} \oplus r_i$.

Step 4: Repeat *Steps 1* and *3* for all blocks. Then, the secret data are recovered completely.

Step 5: The original bitmap containing hidden bits can be recovered from the encrypted image by the EK. For decryption, the exclusive-or results of the encrypted bits and pseudo-random bits are calculated via Equation (16). This restores the bitmap $M = (m_1, m_2, \ldots, m_{16})$.

$$m_j = \sum_{j=1}^{16} m'_j \oplus r_j \tag{16}$$

Step 6: In the data hiding process, the location $\hat{\eta}$ of the codeword was changed, and to restore it during the image restoration process, the payload was attached to the back of M' and recorded. In this was, the bitmap M is restored. That is, after obtaining the changed matrix position(row, column) from the bitmap, the restored bitmap M is obtained using Equation (17), where $rc(1)$ is the row and $rc(2)$ is the column position of the location $\hat{\eta}$.

$$M = \sum_{u=1}^{4} \sum_{v=1}^{4} \begin{cases} \text{flip}(M'_{u,v}), \ if(u = rc(1) \text{ and } v = rc(2)), \\ M'_{u,v}, \ \text{otherwise}. \end{cases} \tag{17}$$

Step 7: Repeat *Steps 1* and *6* for all blocks. Then, all bitmaps are recovered completely.

The image I restored by this process is the cover image and becomes the original AMBTC image.

4. Experimental Results and Discussion

In Section 3, we proposed AMBTC-based SRDH-EI, and in this section, we describe in detail the comparison and analysis of the simulation results to verify the performance of the proposed method. The computing platform used in the experiment has a Core i5-8250U processor, 1.60 GHz speed, 8 GB RAM, and the software for the simulation is MATLAB R2019b. The standard USC-SIPI image database [32] was used for experiments on the proposed model. Among them, some of the original 512×512 grayscale images were used for the experiment. Figure 4 shows a series of test images (e.g., Lena, Pepper, Airplane, Boat, Goldhill, Couple, Baboon, Zelda) used in the experiment.

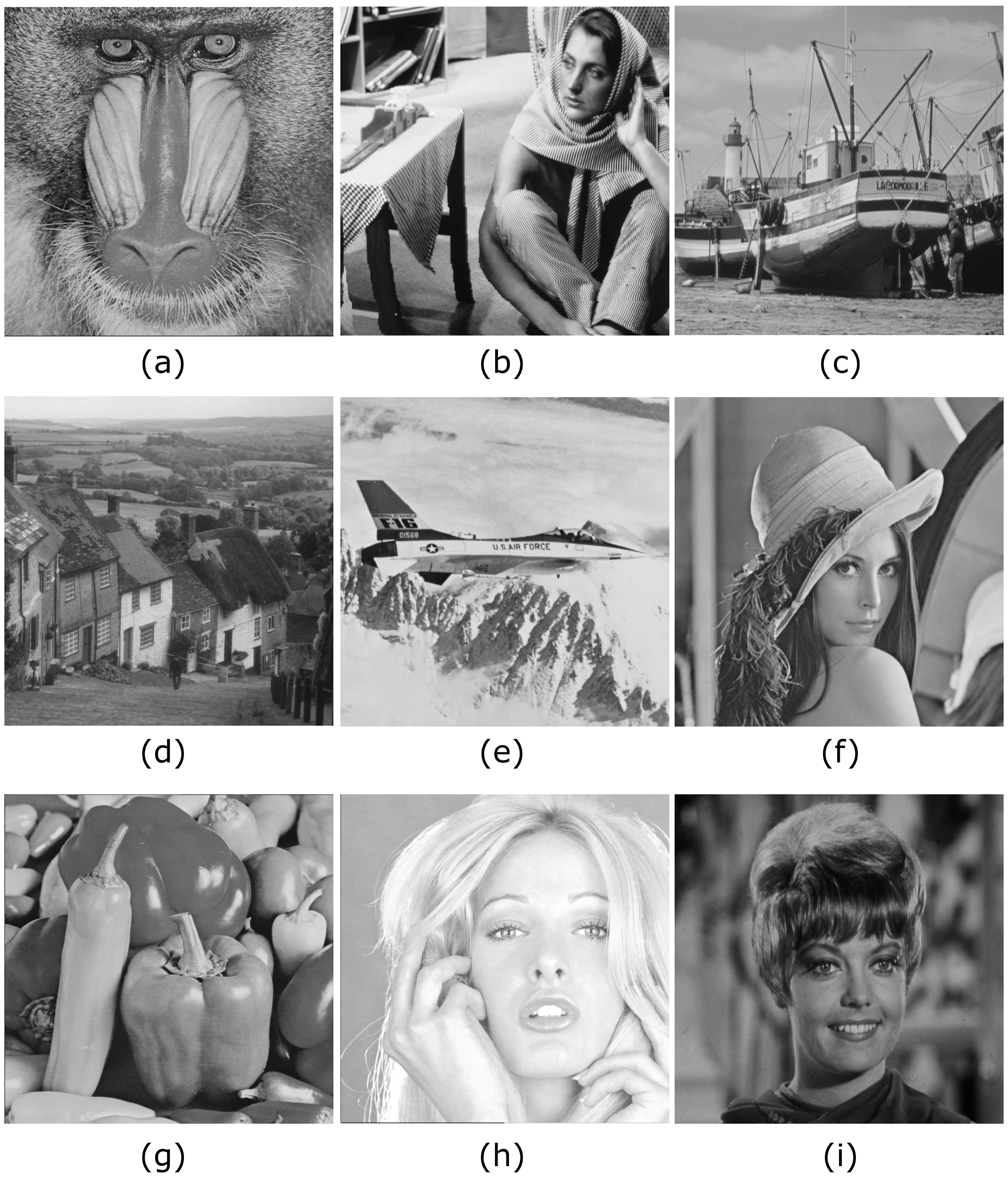


Figure 4. Nine test images: (**a**) Baboon, (**b**) Barbara, (**c**) Boat, (**d**) Goldhill, (**e**) Airplane, (**f**) Lena, (**g**) Peppers, (**h**) Tiffany, (**i**) Zelda (512 × 512).

SSIM (Structural Similarity Index Metric) and PSNR (Peak Signal-to-Noise Ratio) were used for evaluation. The quality of the reconstructed image with data was measured as PSNR and defined as

$$PSNR = 10log_{10}\frac{255^2}{MSE}. \tag{18}$$

PSNR is calculated as $10log_{10}$ (signal power/noise power), and signal power and noise power are calculated using peak power. Here, 255^2 is the allowable pixel intensity. The MSE used for PSNR is the difference in mean intensity between the marked image and the reference image, and the lower the MSE value, the better the image quality can be evaluated. That is, MSE is the mean of squared errors $(g_i - g_i')^2$, where g_i and g_i' are the reference image and the distorted image, respectively. MSE is calculated as follows:

$$MSE = \frac{1}{N \times N} \sum_{i=1}^{N^2} (g_i - g_i')^2 \tag{19}$$

In addition, another measurement introduced for performance evaluation is SSIM, which is a formula (Equation (20)) that measures the similarity between the original image and the marked image.

$$SSIM(g, g') = \frac{(2\mu_g \mu_{g'} + c_1)(2\sigma_{gg'} + c_2)}{(\mu_g^2 + \mu_{g'}^2 + c_1)(\sigma_g^2 + \sigma_{g'}^2 + c_2)} \tag{20}$$

where μ_g and $\mu_{g'}$ are the mean values of g and g', respectively, and c_1 is the stabilization constant and $\sigma_g^2, \sigma_g'^2$, and $\sigma_{gg'}$ are the variances and covariances of the cover image and the stego image. c_1 and c_2 are constant values to avoid division by zero problems.

Table 1 compares the PSNR and SSIM measurements for the cover image, the encrypted image, and the encrypted image when the difference between the two quantization levels is $T = 5$. First, it shows the PSNR and SSIM measurements between AMBTC-compressed images for the original image. The PSNRs of Baboon and Barbara are 26.9765 and 27.0756, which are measured below 30 dB. Baboon is a high-frequency image, and when compressed with AMBTC, the SSIM is z 0.8872 image, which is not classified as bad for the human visual system. It can be seen that the Zelda image has the highest PSNR of 36.6537. The PSNR and SSIM of the encrypted AMBTC are 10 or less and 0.0159 or less, respectively. For this reason, it can be seen that the encrypted image is encrypted to such a level that it is difficult to infer the original image from the encrypted image. For the encrypted image, data were hidden by the proposed method with two quantization levels $T = 5$. When looking at the PSNR and SSIM measurements of the decoded AMBTC including the data, it can be confirmed that it is almost similar to the cover AMBTC. That is, PSNR and SSIM are acceptable even if enough data is concealed.

Table 1. The PSNR and SSIM of the encrypted images and decrypted images when $T = 5$.

Images	Cover AMBTC		Encrypted AMBTC		Decrypted AMBTC	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Barboon	26.9765	0.8872	9.5038	0.0147	26.9757	0.887
Barbara	27.0756	0.9248	7.8847	0.0140	27.0702	0.923
Boat	31.5700	0.9388	8.9932	0.0146	31.5444	0.9355
Goldhill	32.8360	0.9208	9.0695	0.0159	32.8207	0.9196
Airplane	32.0372	0.9504	8.7070	0.0150	32.0017	0.9462
Lena	33.6556	0.9468	9.3058	0.0167	33.599	0.9423
Pepper	34.0968	0.9395	9.1550	0.0147	34.0397	0.9356
Tiffany	35.6576	0.9473	6.8659	0.0140	35.5651	0.9426
Zelda	36.6537	0.9476	8.8716	0.0140	36.5378	0.9432
Average	32.3736	0.9337	8.6421	0.0153	32.2394	0.9306

Figure 5 shows the encrypted image mentioned in Table 1, and data hiding is applied to the encrypted image and transmitted to the receiver. The receiver cannot know the existence of the data and the form of the image from this encrypted image. This is the advantage of encryption RDH. Encryption for AMBTC was performed for bitmap and two quantization levels, respectively. If only one of the bitmaps or the two quantization levels is encrypted, a part of the original image may remain in the encrypted image.

Table 2 compares EC and PSNR according to the difference T between the two quantization levels. As the value of T decreases, EC decreases while PSNR increases. While, as the value of T increases, EC increases while PSNR deteriorates. In Table 2, it can be seen that EC and PSNR were measured while increasing from $T = 5$ to $T = 20$. When $T = 5$, the average EC is 21,295 and the PSNR is 32.2393, which is high with an average of 32 dB. When $T = 10$, the average EC is 36,657 and the PSNR remains above 32 dB. When $T = 20$, the average EC was measured to be 48,336 and the PSNR was 29 dB, which can be seen to drop to less than 30 dB, but it shows a similar performance to the human visual system. In conclusion, if a block with a small difference between the two quantization levels is sufficient, not only can a large amount of data be hidden, but also the PSNR is excellent. This is affected by the characteristics of the image. Table 2 shows that it is difficult to hide a sufficient amount of data in the case of high-frequency images such as Baboon. In the case of a low-frequency image such as Pepper, since the difference between the two quantization levels in many blocks is relatively small, it is possible to hide a large amount of data while maintaining a high PSNR.

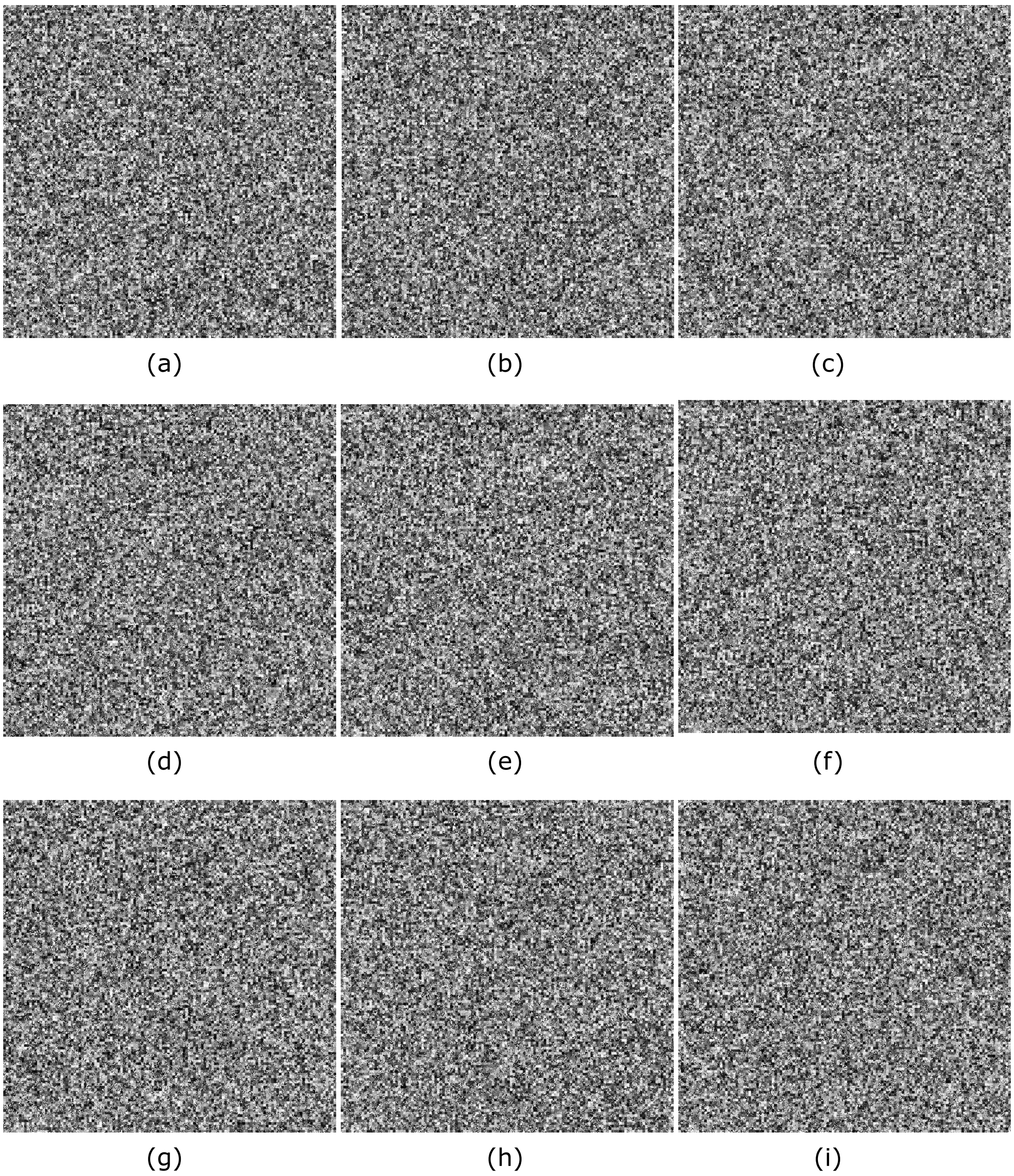


Figure 5. Encryption results of the cover images: (**a**) Baboon, (**b**) Barbara, (**c**) Boat, (**d**) Goldhill, (**e**) Airplane, (**f**) Lena, (**g**) Peppers, (**h**) Tiffany, (**i**) Zelda.

Table 2. Comparison of EC and PSNR performance according to the difference between the two quantization levels.

Images	T = 5		T = 10		T = 15		T = 20	
	EC (Bits)	PSNR	EC (Bits)	PSNR	EC (Bits)	PSNR	EC (Bits)	PSNR
Baboon	1113	26.9757	10121	26.9571	18,649	26.9161	24,557	26.8607
Barbara	14,889	27.0702	26,493	27.0485	32,125	27.0208	35,153	26.9922
Boat	28,217	31.5444	36,133	31.5034	42,313	31.4161	47,357	31.2857
Goldhill	9385	32.8207	27,769	32.6851	41,445	32.4429	48,965	32.1992
Airplane	34,549	32.0017	44,637	31.9485	49,469	31.8705	52,657	31.7764
Lena	29,677	33.599	44,217	33.4861	50,637	33.3503	54,329	33.2079
Pepper	21,149	34.0397	45,997	33.8319	52,921	33.6764	56,149	33.5425
Tiffany	29,381	35.5651	45,457	35.3734	52,609	35.1416	56,497	34.9002
Zelda	23,301	36.5378	49,093	36.1607	55,857	35.9139	59,365	35.6629
Average	21,295	32.2393	36,657	32.1105	44,002	31.9720	48,336	29.0427

Table 3 compares Yin et al.'s method of an AMBTC-based model with existing methods and our proposed method. Cover AMBTC is a criterion for evaluating performance, and a comparison of the PSNR can be conducted based on this criterion. PSNR^1 and PSNR^2 were obtained from the decrypted images and recovered images, respectively. Here, we measure PSNR^1 and PSNR^2 when $T = 5$. When hiding message bits in a bitmap, flip the bitmap from '0' to '1' or '1' to '0'. That is, when the difference between the two quantization levels is 1, the error occurring in this block is 1. If $T = 2$, the error is 2. In the end, if the value of T has increased, the amount of data to be hidden can be sufficiently increased, but the error will increase, the PSNR will deteriorate, and image damage will be large. Table 2 shows the EC and PSNR for the image while increasing the T value. Table 3 shows the simulation results for $T = 5$, where PSNR and EC were judged to be at appropriate levels. $T = 5$ was selected under the judgment that EC was at an appropriate level while minimizing PSNR. Comparing the payload of Yin et al.'s with the payload of our proposed method, we demonstrated that the average payload of our proposed method is 20 times better. In addition, if the image is restored, it shows the advantage of being able to restore the original cover image.

Table 3. Comparison of the performance between Yin et al.'s method and the proposed method.

Images	Cover Images	Yin et al.'s [22]			Proposed Method		
	PSNR	Payload	PSNR^1	PSNR^2	Payload	PSNR^1	PSNR^2
Baboon	26.9765	141	23.66	$+\infty$	1113	26.9757	$+\infty$
Boat	31.57	554	30.32	$+\infty$	28,217	31.5444	$+\infty$
Airplane	32.0372	1458	30.40	$+\infty$	34,549	32.0017	$+\infty$
Lena	33.6556	1213	33.00	$+\infty$	29,677	33.5990	$+\infty$
Tiffany	35.6576	1687	29.81	$+\infty$	29,381	35.5651	$+\infty$
Average	31.9793	1010	27.93	$+\infty$	24,587	31.9371	$+\infty$

When implementing Yin et al.'s method [22], it is necessary to record some auxiliary information and embed it into the bitmap of the first several triples by sacrificing some embedding space. Thus, their scheme does not fully excavate and make full use of the characteristic of the AMBTC compression codes. In Wang et al's method [23], each block records a pair of peak and zero points. Since the histogram modification method proposed here was first devised for gray images, it does not provide a sufficient amount of redundant bits for data hiding the compressed images, so the data hiding capacity is relatively small. Su et al.'s method [24] considers the natural correlation of quantization levels within a block and had an improved embedding capacity compared to the Yin et al.'s scheme. The method we proposed is a method that can modify 1 pixel for each block and hide 4 bits, and is an optimized method to ensure data hiding efficiency and a high image quality. As shown in Table 4, it can be seen that the EC of our proposed method has the best performance.

Table 4. Comparison of maximum EC between our scheme and previous schemes.

Images	**Yin et al.'s (2018) [22]**	**Wang et al. (2019) [23]**	**Su et al.'s (2020) [24]**	**Proposed**
Baboon	273	-	14,976	64,881
Boat	932	22,050	29,382	65,369
Airplane	1365	23,610	34,564	65,013
Lena	1270	22,287	36,418	65,329
Barbara	524	-	13,644	62,945
Peppers	926	22,142	41,080	64,929
Average	881	22,522	28,344	64,744

Figure 6 shows the results of decryption after hiding data in the encrypted image when the difference between the two quantization levels is $T = 5$. The quality of the decoded image including the data can be visually compared to the original. When looking at Figure 6a,b, which are the decoded images, the PSNR is clearly not high below 30 dB, which shows the image quality. Since the original image of (a) and (b) was also less than 30 dB, the PSNR of the decoded image was measured to be less than 30 dB. Except for this, the performance is over 30 dB. When compressing the original image with AMBTC, since compression representing two quantization levels per block was used, the same level of performance as the original image was not observable, but as shown in Table 1, the SSIM was 0.9 or higher except for the case of Barboon. It can be seen that the characteristics of the original image are sufficiently reflected.

In the case of restoring the original image by removing the error occurring for each block in the decoded image, it can be restored with the cover AMBTC shown in Table 1. In this way, it is expected that it is suitable for various applications because not only is the quality of the image sufficiently similar to that of the original image, but the compression can also be performed at a high level.

Table 5 summarizes and compares the performance of the proposed method and the existing method. Zhang [12] proposed a method capable of extracting data without errors from the encrypted image, but there is a problem in the complete restoration of the original cover image. Ma et al.'s method [13] and Zhang et al.'s method [19] achieved data extraction and complete restoration of the original cover image. However, both approaches require reserve space before hiding data. The method of Yin et al. [22] maintained the block correlation of the image by using the histogram of the prediction error in the data embedding step and used the method of utilizing the extra space. This method also recovers the original image without errors using both keys. In Wang et al.'s method [23], the RDH is based on the histogram modified embedding method, which is utilized for obtaining a high payload and low storage memory. Su et al.'s method [24] means that secret messages can be embedded by exploiting the redundant room derived from the quantization levels. Data hiding is then performed with the use of a labelling strategy. This method is an RDH method, which has already been verified. Therefore, there is no problem with restoring the original image. The method we proposed is a method that uses a Hamming code, has excellent data hiding performance and restored image quality, and it is also possible to restore the original cover image using this method.

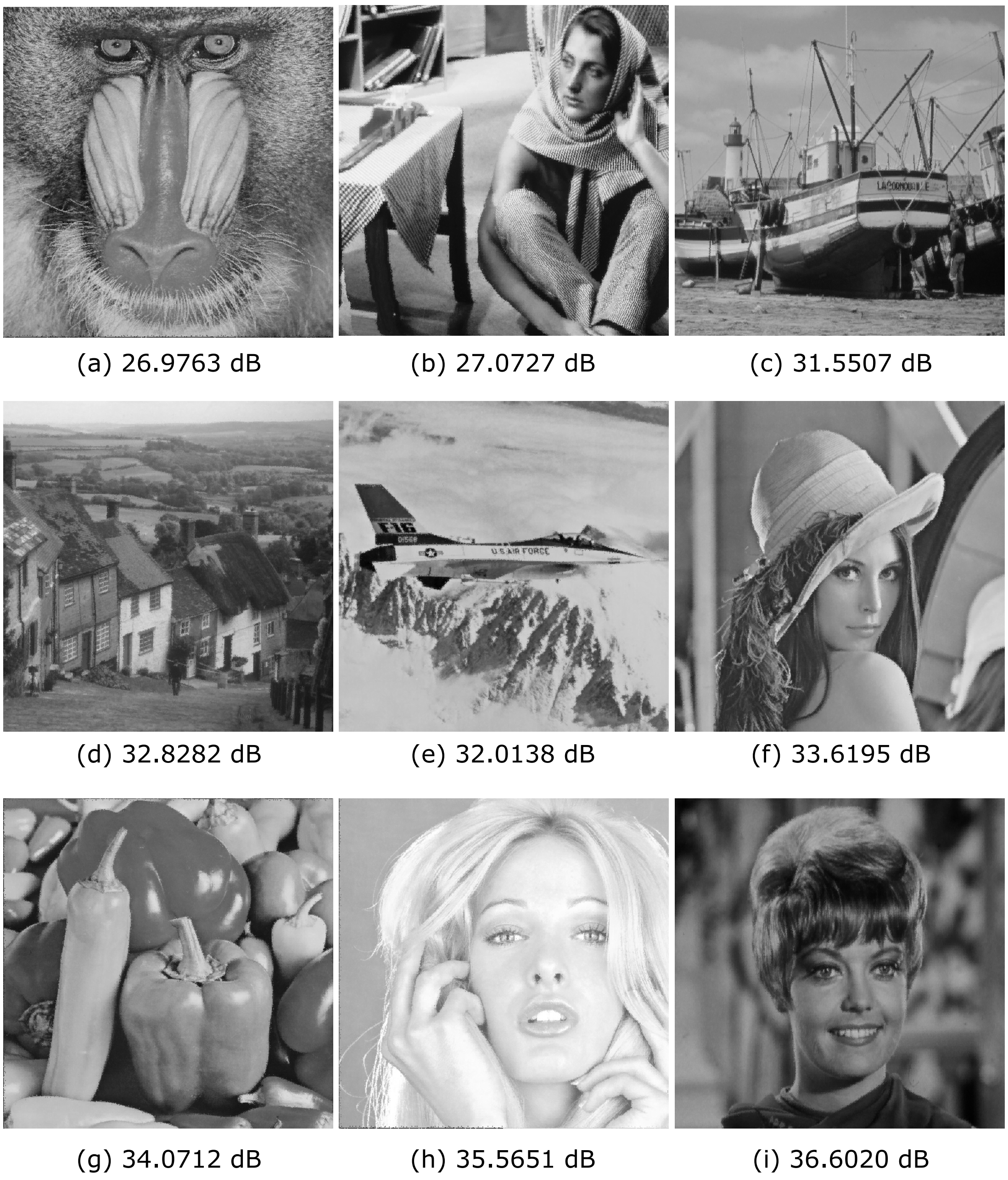


Figure 6. Comparison of quality of images recovered from encrypted images: (**a**) Baboon, (**b**) Barbara, (**c**) Boat, (**d**) Goldhill, (**e**) Airplane, (**f**) Lena, (**g**) Peppers, (**h**) Tiffany, (**i**) Zelda.

Table 5. Performance comparison among previous schemes and proposed scheme.

Method	**Criterion**			
	Separable	**Error in Extracted Data**	**Error in Reconstructed Cover Image**	**Embedding Efficiency**
Ma et al.'s [13]	O	X	X	Mid
Zhang's [12]	X	X	O	Low
Zhang et al.'s [19]	O	X	X	Low
Yin et al.'s [22]	O	X	O	High
Wang et al.'s [23]	O	X	O	High
Su et al.'s [24]	O	X	O	High
Proposed	O	X	O	High

5. Conclusions

In this paper, we proposed an AMBTC-based SRDH-EI model. This model consists of the encryption of the AMBTC cover image, data hiding of the encrypted AMBTC cover image, recovery of the encrypted AMBTC and data extraction. As a result of the simulation, the proposed method demonstrated twice the EC performance of the method proposed by Su et al. [24], which displayed the best performance among the existing methods. In addition, as shown in Table 3, the high image quality was maintained with an average of 31.93 dB when $T = 5$. In the future, we plan to conduct further research on effective ways to hide larger amounts of data while maintaining high image quality.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The author declares no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AMBTC	Absolute Moment Block Truncation Coding
BTC	Block Truncation Coding
DH	Data Hiding
RDH	Reversible Data Hiding
SRDH-EI	Separable RDH in an Encrypted Image
HC	Hamming Code
I	Cover Image
I'	Encrypted Cover Image
I''	Marked Image
g	pixel of grascale image
Q	quantization levels
M	bitmap
c	codeword
H	parity check matrix
m	hidden bits
η	syndrome
EC	Embedding capacity
PSNR	Peak Signal-to-Noise Ratio
MSE	Mean-Squared Error
DE	Difference Expansion
HS	Histogram Shift

References

1. Bender, W.; Gruhl, D.; Morimote, N.; Lu, A. Techniques for data hiding. *IBM Syst. J.* **1996**, *35*, 313–336. [CrossRef]
2. Kim, C.; Shin, D.-K.; Yang, C.-N.; Leng, L. Hybrid Data Hiding Based on AMBTC Using Enhanced Hamming Code. *Appl. Sci.* **2020**, *10*, 5336. [CrossRef]
3. Yang, C.N.; Wu, S.Y.; Chou, Y.S.; Kim, C. Enhanced stego-image quality and embedding capacity for the partial reversible data hiding scheme. *Multimed. Tools Appl.* **2019**, *78*, 18595–18616. [CrossRef]
4. Shi, Y.Q.; Li, X.; Zhang, X.; Wu, H.-T.; Ma, B. Reversible data hiding: Advances in the past two decades. *IEEE Access* **2016**, *4*, 3210–3237. [CrossRef]
5. Tian, J. Reversible data embedding using a difference expansion. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 890–896. [CrossRef]
6. Wu, D.C.; Tsai, W.H. A steganographic method for images by pixel-value differencing. *Pattern Recognit. Lett.* **2013**, *24*, 1613–1626. [CrossRef]
7. Alattar, A.M. Reversible watermark using the difference expansion of a generalized integer transform. *IEEE Trans. Image Process.* **2004**, *13*, 1147–1156. [CrossRef]

8. Ni, Z.; Shi, Y.Q.; Ansari, N.; Su, W. Reversible data hiding. *IEEE Trans. Circuits Syst. Video Technol.* **2006**, *16*, 354–362.
9. Tsai, P.Y.; Hu Y.C.; Yeh, H.L. Reversible image hiding scheme using predictive coding and histogram shifting. *Signal Process.* **2009**, *89*, 1129–1143. [CrossRef]
10. Abanda,Y.; Tiedeu, A. Image encryption by chaos mixing. *IET Image Process.* **2016**, *10*, 742–750. [CrossRef]
11. Li, C.; Lin, D.; Lü, J. Cryptanalyzing an image-scrambling encryption algorithm of pixel bits. *IEEE Trans. Multimed.* **2017**, *24*, 64–71. [CrossRef]
12. Zhang, X. Reversible data hiding in encrypted image. *IEEE Signal Process. Lett.* **2011**, *18*, 255–258. [CrossRef]
13. Ma, K.; Zhang, W.; Zhao, X.; Yu, N.; Li, F. Reversible data hiding in encrypted images by reserving room before encryption. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 553–562. [CrossRef]
14. Zhang, W.; Ma, K.; Yu, N. Reversibility improved data hiding in encrypted images. *Signal Process.* **2014**, *94*, 118–127. [CrossRef]
15. Hong, W.; Chen, T.S.; Wu, H.Y. An improved reversible data hiding in encrypted images using side match. *IEEE Signal Process. Lett.* **2012**, *19*, 199–202. [CrossRef]
16. Liao, X.; Shu, C. Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels. *J. Vis. Commun. Image Represent.* **2015**, *28*, 21–27. [CrossRef]
17. Wu, X.; Sun, W. High-capacity reversible data hiding in encrypted images by prediction error. *Signal Process.* **2014**, *104*, 387–400. [CrossRef]
18. Yin, Z.; Abel, A.; Zhang, X.; Luo, B. Reversible data hiding encrypted image based on block histogram shifting. In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016; pp. 2129–2133.
19. Zhang, X. Separable reversible data hiding in encrypted image. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 826–832. [CrossRef]
20. Qian, Z.; Han, X.; Zhang, X. Separable reversible data hiding in encrypted images by n-nary histogram modification. In Proceedings of the 3rd International Conference on Multimedia Technology (ICMT-13), Guangzhou, China, 29 November–1 December 2013; pp. 869–876.
21. Yin, Z.; Luo, B.; Hong, W. Separable and error-free reversible data hiding in encrypted image with high payload. *Sci. World J.* **2014**, *2014*, 604876. [CrossRef]
22. Yin, Z.; Niu, X.; Zhang, X.; Tang, J.; Luo, B. Reversible data hiding in encrypted AMBTC images. *Multimed. Tools Appl.* **2018**, *77*, 18067–18083. [CrossRef]
23. Wang, H.Y.; Lin, H.J.; Gao, X.Y.; Cheng, W.H.; Chen, Y.Y. Reversible AMBTC-based data hiding with security improvement by chaotic encryption. *IEEE Access* **2019**, *7*, 38337–38347. [CrossRef]
24. Su, G.; Chang, C.; Lin, C. A high capacity reversible data hiding in encrypted AMBTC-compressed images. *IEEE Access* **2020**, *8*, 26984–27000. [CrossRef]
25. Delp, E.; Mitchell, O. Image compression using block truncation coding. *IEEE Trans. Commun.* **1979**, *27*, 1335–1342. [CrossRef]
26. Lema, M.D.; Mitchell, O.R. Absolute moment block truncation coding and its application to color images. *IEEE Trans. Commun.* **1984**, *32*, 1148–1157. [CrossRef]
27. 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.
28. 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]
29. 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. [CrossRef]
30. Rurik, W.; Mazumdar, A. Hamming codes as error-reducing codes. In Proceedings of the 2016 IEEE Information Theory Workshop (ITW), Cambridge, UK, 11–14 September 2016; pp. 404–408.
31. Moon, T.K. *Error Correction Coding–Mathematical Methods and Algorithms*; John Wiley and Sons: Hoboken, NJ, USA, 2005; pp. 2001–2006.
32. Image Databases. Available online: https://www.imageprocessingplace.com/root_files_V3/image_databases.htm (accessed on 10 August 2022).