electronics

MDPI

Article

Dual Reversible Data Hiding in Encrypted Halftone Images Using Matrix Encoding

Cheonshik Kim [1,*], Nhu-Ngoc Dao [1], Ki-Hyun Jung [2] and Lu Leng [3,*]

[1] Department of Computer Engineering, Sejong University, Seoul 05006, Republic of Korea; nndao@sejong.ac.kr
[2] Department of Software Convergence, Andong National University, Andong 36729, Republic of Korea; kingjung@anu.ac.kr
[3] School of Software, Nanchang Hangkong University, Nanchang 330063, China
* Correspondence: mipsan@sejong.ac.kr (C.K); leng@nchu.edu.cn (L.L.)

Abstract: Data hiding and reversible data hiding research has primarily focused on grayscale and color images, because binary and halftone images are prone to visual distortion caused by a small number of errors in pixel representation. As a result, reversible data hiding is more useful than halftone-based data hiding. This study proposes an investigation of encrypted halftone images based on dual reversible data hiding, which improves the reversibility and security of the image by utilizing a dual cover image. Since halftone images are adequately compressed, they are beneficial in low-channel-bandwidth environments. Hamming code (HC) (7,4) is applied to each block of the halftone image to hide the secret data, and two halftone images are recorded and sent to different receivers at the end of the embedding process. Recipients can use the proposed method and the two marked images to extract the message and recover the cover halftone image. The proposed data hiding method can enhance the quality of the decrypted image by appropriately increasing the block size, and conversely, sufficiently large amounts of data can be hidden by reducing the block size. The experimental results provide evidence of the effectiveness of the proposed method in terms of both image quality and the embedding rate.

Keywords: data hiding (DH); reversible DH (RDH); encryption; dual RDH; dual encrypted RDH; halftone image; Hamming code (HC)

Citation: Kim, C.; Dao, N.-N.; Jung, K.-H.; Leng, L. Dual Reversible Data Hiding in Encrypted Halftone Images Using Matrix Encoding. *Electronics* **2023**, *12*, 3134. https://doi.org/10.3390/electronics12143134

Academic Editors: Aleksandra Kawala-Sterniuk, Aleksandra Świetlicka and Piotr Schneider

Received: 23 June 2023
Revised: 14 July 2023
Accepted: 18 July 2023
Published: 19 July 2023

1. Introduction

The world we live in today is called the "digital world", and every day, a huge amount of digital content is created, stored, and shared. In today's environment where the Internet is widely used, it is often desirable to embed authentication data in images to ensure the copyright status and authentication of digital content (e.g., images, video, audio, etc.). In addition, this method can be an important option when confidential communication is required. This technique of secretly hiding data in an image is referred to as the data hiding (DH) technique [1–3]. An image containing data is called a marked image, and the marked image must be visually identical to the original image so that the information hidden in the image cannot be detected by an attacker.

However, a problem arises as the marked image generated by data hiding cannot be restored to its original form. For this reason, DH is suitable for applications where image recovery is not required. Reversible data hiding (RDH) [4–9], which can recover the original image after extracting data from the marked image, is required in fields (e.g., military and medical) where decision-making based on accurate image quality is essential.

Most existing RDH algorithms are mainly based on lossless compression (Fridrich et al. (2002) [5], difference expansion (DE) (Tian (2003) [6], Alattar (2004) [7]), histogram shift (HS) (Ni et al. (2006) [8], Qin et al. (2013) [9]), and prediction error extension (PEE) (Dragoi et al. (2014) [10]). These RDH algorithms usually exploit spatial correlations between pixels within pixel pairs or prediction errors to contain secret data.

Encryption [11–13] is an effective way of protecting the privacy of various types of digital content from attackers, especially in secret image sharing [14,15], where the owner must encrypt the image before a trusted service provider (SP) shares the images. Encrypted-image-based RDH is absolutely necessary for privacy protection.

RDH methods can be broadly divided into single-image-based RH methods, double-image-based RDH methods, and encrypted-image-based RDH methods. Encrypted-image-based RDH (RDH-EI) [13,16–25] differs from previous RDH methods in that it aims to protect both the original images and the secret data simultaneously. In RDHEI, there are three users: the content owner, data hider, and receiver. The content owner encrypts the original image with the encryption key (EK) and uploads it to the cloud server. The data hiding key (DK) can be used to insert additional secret data into the encrypted image, and without the EK, the encrypted image cannot be recovered. If the recipient has both the EK and DK, the original image can be recovered and the secret data extracted.

In Zhang (2011) [16], a data hiding method was proposed in which the least significant bit (LSB) in the half pixels of each block is inverted to hide data. Before that, the original video is encrypted with an encryption key, converted into a scrambled image, and divided into several blocks. The receiver uses a fluctuation function to extract the secret data and recover the image. With this algorithm, data extraction and image recovery errors can occur with complex blocks. Undoubtedly, the RDH-EI can be used for a variety of existing applications, such as authentication, copyright, and personal security protection.

To separate data extraction and image recovery, Zhang (2012) [17] first introduced the separable RDHEI (SRDHEI) method. The original image is encrypted with a stream cipher. The LSB of the encrypted image is compressed and provided with white space for data insertion. Image decryption and data extraction can be performed independently. On the receiver side, the encryption key (EK) and the data hiding key (DK) can be used to decrypt or extract data, regardless of the order in which they are used. If you have both keys, you can recover the original image and extract data.

Wu and Sun (2014) [18] used a most significant bit (MSB) substitution technique to embed a secret bit in an encrypted image. On the receiver side, prediction techniques are used to extract the data and recover the original image. Shiu et al. (2015) [19] used the DE technique to convert adjacent original pixel pairs into odd or even pixel pairs and then encrypted the pixel pairs using Paillier encryption. A secret bit was added to each encrypted pixel pair. During extraction, bit "0" can be extracted if the pixel pair is even or odd. Otherwise, you obtain bit "1". The payload is about 0.5 bpp. However, the use of Paillier encryption leads to an increase in the size of the data and causes high storage costs.

Yi et al. (2019) [20] proposed a binary tree differential coding algorithm. The algorithm can have an embedding capacity exceeding 2 bpp. Wang et al. (2019) [21] proposed an AMBTC-based data hiding method. Here, an adaptive variable N-bit level truncation method is used to hide data in each block. Since the histogram of an image is used for data hiding, the amount of data is determined by the characteristics of the image.

Mohammadi et al. (2020) [23] proposed an intelligent RDHEI method with a high payload. The original image is divided into blocks, and the prediction error between the reference pixel and other pixels in each block is calculated. The payload of each pixel can be determined based on the magnitude of the prediction error. With the help of the block labels, data extraction and image recovery can be conducted perfectly.

The two-image-based RDH algorithm creates two marked images by making two copies of the mark image and then inserting a secret message into each copy. For security reasons, an attacker cannot decrypt the secret message without accessing both copies of the cover image simultaneously. Lu et al. (2015) [26] presented a two-image algorithm based on the concept of the central convolution of secret messages. Yao et al. (2017) [27] presented a dual-image RDH algorithm with minimal pixel coordinate distortion.

Lee and Huang (2013) [28] presented a dual-image-based RDH algorithm using directional combinations of pixel coordinates. Lu et al. (2017) [29] proposed a frequency coding algorithm to overcome the drawbacks of Lu et al. [26]. Jana et al. (2018) [30] proposed a

(7,4) Hamming-code-based embedding ([31]) technique in which secret message bits are embedded by error generation and the original image is recovered using a Hamming error correction code. Sun et al. (2020) [32] proposed a fully reversible dual-image-based RDH method for encrypted halftone images. Their method facilitates the recovery of the original image along with sufficient data hiding. After decrypting the image, the original image can be recovered properly, and the data hidden by SP can be extracted accurately.

The cover image can be roughly divided into two features. They are continuous tone images and halftone [33] images. Continuous tone images include BMP or JPEG, while halftone images consist of pixels that are either 0 or 1. Most RDH and RDH-EI studies are based on continuous-tone grayscale images. This ensures a lower degree of image distortion, even when the data are sufficiently occluded. Unlike grayscale images, halftone images use 1 bit per pixel, so there are many limitations when implementing the RDH algorithm. Since the information redundancy in binary images is usually low, the traditional RDH algorithm cannot be applied to halftone images.

For example, the difference expansion (DE) method modifies the difference between adjacent pixels and can hide 1 bit of data, but cannot hide data when this method is applied directly to a halftone image. The histogram modification method requires a large number of redundant pixels to hide a sufficient amount of data. Therefore, it is not suitable for halftone images. Also, other existing RDH methods, such as pixel prediction, cannot be applied to halftone images due to their nature. Therefore, halftone images have a lower embedding capacity than continuous tone images.

As mentioned earlier, relatively few studies have been conducted on halftone DH [34–39] due to its difficulties. A DH algorithm was proposed using a method of forced switching at random positions. Fu and Oscar [39] introduced Data Hiding Smart Pair Toggling (DHSPT), which hides data by forcibly toggling complementary colors at pseudo-random positions within a halftone image. The complementary pixels are selected to minimize the possibility of forming visually undesirable clusters. In addition, for situations where the halftone method is an error diffusion, a modified data concealment error diffusion (MDHED) process that integrates the data concealment task into the error diffusion process was proposed.

Pan et al. (2007) [34] proposed a DH method based on a lookup table, which has a time complexity consideration. Tsai (2009) [35] proposed an RDH method for vector quantized compressed images using histogram modification. Xuan et al. (2008) [36] introduced RDH using the run-length method, which has the advantage of requiring little book-keeping to invert original halftone images. However, finding a suitable hiding position is very time consuming. Kim et al. (2013) [37] proposed RDH for halftone images using histogram modification. This method is effective for binary images as a data hiding method that uses the pattern with the highest frequency and the pattern with the lowest frequency among the 4-bit patterns. Yin et al. [38] proposed a RDH method for halftone images based on dynamic embedding state group (DESG), which can embed at least 1 bit of secret messages per embeddable pixel or pattern.

In this paper, we present a novel approach for reversible data hiding (RDH) based on the encryption of dual halftone images. Our method uses the efficient Hamming code [40,41], referred to in the work of Rurik and Moon, which minimizes the number of erroneous pixels. This strategy allows a considerable number of secret bits to be hidden while minimizing the distortion of the image.

The concept of RDH in encrypted images (RDH-EI) was originally proposed by Zhang [16], who applied it to grayscale images. However, our research breaks new ground by applying RDH-EI to halftone images, which is a significant departure from previous studies. The contribution of our study lies in the innovative application of RDH-EI to dual halftone images, a concept that has not been explored before. The complexity of halftone images, which are characterized by pixels composed of individual bits, presents a significant challenge for data hiding. However, this complexity also increases the security of our method. Recovering the original image and the secret data is particularly difficult without displaying both images, which makes our method more secure than methods

based on a single image. Our approach partially incorporates the concept of secret sharing, which further enhances the security of our method. We believe that our research represents a significant advance in the field of data hiding and image encryption.

The rest of this paper is organized as follows: Section 2 introduces the necessary background for the proposed method, including the Hamming codes and error diffusion halftoning. The proposed scheme is elaborated in Section 3. The experimental results are presented in Section 4. Finally, Section 5 concludes this paper.

2. Preliminaries

2.1. Error Diffusion Method for Halftone Images

Halftoning [33] is the conversion of a multicolor image into a two-color image, creating a visual representation that gives the impression of the original multicolor image when viewed from a distance. Halftone images are commonly used for printing books, magazines, newspapers, and computer printers that are common in everyday life [42]. For copyright protection and authentication reasons, it is desirable to hide metadata, such as the company ID, owner information, and date and time of creation, in the halftone image. The most commonly used method for converting a multitone image to a halftone image is the error diffusion method (ED). Although the ED approach is somewhat complicated, the halftone images it produces have good visual quality. For ED, the popular kernels are "Jarvis" and "Steinberg" [43].

Floyd–Steinberg [43] dithering is an image dithering algorithm that was first published in 1976 that uses error diffusion to achieve dithering. Floyd–Steinberg dithering is one of the error diffusion methods that preserves as much detail as possible in the original image while reducing the color in the image. In this method, the color of each pixel is replaced by the nearest color, and any errors are distributed to neighboring pixels. This algorithm is widely used due to its computational efficiency, accuracy, and high-quality results. The advantage of ED with Steinberg is that it gives a fine texture and good contrast.

In other words, the Floyd–Steinberg dithering algorithm adds (or pushes) the remaining quantization errors to nearby pixels for later processing. There are several error diffusion algorithms for improving the halftone quality. Most existing algorithms are based on the Floyd–Steinberg error diffusion algorithm. In the following text, we consider the Floyd–Steinberg error diffusion algorithm, which is a typical error diffusion algorithm. Figure 1a briefly illustrates the flow of this algorithm through a schematic diagram. Figure 1b depicts the kernel used by Floyd and Steinberg.

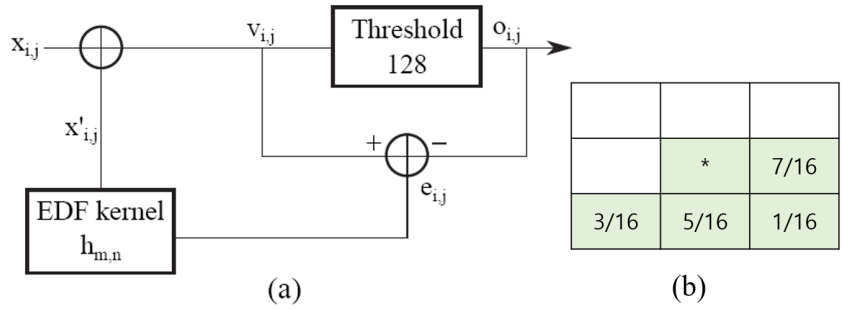


Figure 1. (**a**) ED diagram, (**b**) Floyd and Steinberg's kernel.

In Figure 1, the variables $x_{i,j}$ and $v_{i,j}$ represent the sum of diffusion errors added to the value of the current input pixel and the neighboring processing pixel where the kernel is located, respectively. The variables $o_{i,j}$ and $h_{m,n}$ are also binary and error kernels, respectively.

$$o_{i,j} = \begin{cases} 1, & if\ (x'_{i,j} \geq 128), \\ 0, & otherwise. \end{cases} \tag{1}$$

$$v_{i,j} = x_{i,j} - \sum_{m=0}^{2} \sum_{n=-2}^{2} e_{i+m,j+n} \times h_{m,n} \tag{2}$$

In the context of error diffusion in halftone imaging, the 'kernel' plays a crucial role. It manages the distribution of the difference $e_{i,j}$, i.e., the discrepancy between $o_{i,j}$ (the original pixel value) and $x_{i,j}$ (the modified pixel value). The variable $v_{i,j}$ represents the adjusted pixel value after error diffusion. It can be calculated using Equations (1)–(3). Finally, the signal $e_{i,j}$ serves as a representative of the accumulated error value transferred to the neighboring pixels during the error diffusion process.

$$e_{i,j} = v_{i,j} + o_{i,j} \tag{3}$$

In the image processing algorithm, pixels marked with an asterisk (*) (Figure 1b) are those that are being scanned, while empty pixels represent those that have already been processed. This algorithm systematically scans the image from left to right and from top to bottom, quantizing pixel values one-by-one. At each step, the calculated quantization errors are applied to adjacent pixels that are yet to be processed, not to pixels that have already been quantized. This process implies that a large number of pixels that are rounded up or down during quantization could affect the rounding direction of the next pixel, resulting in an equilibrium where the quantization error is reduced to near zero on average. However, for optimal dithering, it is important that the estimated quantization errors are as accurate as possible so that rounding errors do not affect the final result.

2.2. Hamming Codes

We explain the basic concepts of the Hamming code (HC) [40,41] for cover coding. A linear perfect single error correcting code with a minimum distance of 3 is called HC(n, k). The codeword length for the integer $m \geq 2$ is $n = 2^m - 1$, and the message length is $k = 2^m - m - 1$. However, if multiple errors occur in a codeword, error correction cannot be guaranteed. Equation (4) provides the parity check matrix $\mathcal{H}$ for the HC. Suppose $\mathcal{H}$ is a $k \times n$-dimensional matrix with $G \cdot \mathcal{H}^T = [0]_{(n-k)\times k}$, where $G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$.

$$\mathcal{H} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{4}$$

Let $y \in \mathbb{F}_2^n (n = 2^k - 1)$ be a codeword obtained from an information word $x \in \mathbb{F}_2^{n-k}$ via a $(n - k) \times n$ generator matrix G, where $y = x \cdot G$. We can figure out a one-bit error pattern to correctly decode the codeword with a bit error. Assume that the received codeword is y', and the error pattern is $e = (y \oplus y')$. The position of e can be obtained from Equation (5).

$$\begin{cases} y' \cdot \mathcal{H}^T = (e \oplus y) \cdot \mathcal{H}^T = e \cdot \mathcal{H}^T + y \cdot \mathcal{H}^T \\ (supp. : y = (x \cdot G) \cdot \mathcal{H}^T = 0) \\ = e \cdot \mathcal{H}^T + 0 = e \cdot \mathcal{H}^T \end{cases} \tag{5}$$

Suppose that an error bit occurs at $\hat{y}$ (the 6th bit from the left). That is, $e = (e_1, e_2, \ldots, e_7) = (0\ 0\ 0\ 0\ 1\ 0)$. In this case, the syndrome $\eta = (110)$. That is, errors in codewords can be accurately identified with the syndrome. We divide the cover image into N/n subsets of n pixels each to hide the message bits in each set, where n is the length of the code. For a codeword of code length n, an error may or may not occur in n locations. The decoding procedure for the HC is described as follows:

Step 1: Calculate the syndrome $\eta(y') = b(y') \cdot \mathcal{H}^T$, where $b(y'_i) = (y'_i \ mod\ 2)$ and y' is the received codeword.

Step 2: Find the error pattern e from the syndrome.

Step 3: Modify the cover object so that $y = (y' \oplus e)$, and then decode $x = y \oplus G^{-1}$.

3. Proposed Method

In this paper, we introduce a new reversible data hiding in encrypted images (RDH-EI) method applied to two encrypted halftone images. Figure 2 presents a simplified representation of this proposed method. The halftone image used in the experiment is generated by applying Floyd–Steinberg dithering to the original gray image. In this process, the owner encrypts the original halftone image using an encryption key (EK) to generate the cover image, denoted as $\mathcal{I}$. After creating a duplicate of the cover image, referred to as $\mathcal{I}_2$, the owner sends both encrypted cover images to the service provider. The service providers (SPs), using data-hiding keys (DKs), embed additional information within these encrypted images, resulting in two marked images.

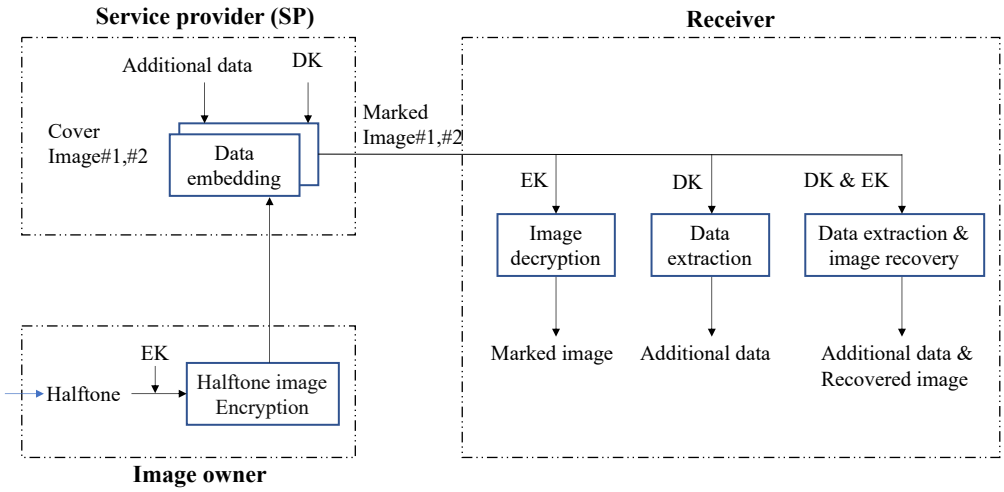


Figure 2. Schematic diagram for the embedment procedure.

On the receiver's side, there are three possible scenarios: (i) The secret message can be extracted if the recipient only has the DK. (ii) If they only have the EK, he can almost certainly recover the original halftone image with some noise. (iii) If the recipient has both the DK and EK keys, they can recover the original halftone image and extract the secret data securely (Figure 2).

3.1. Halftone Image Encryption

In this section, the halftone image encoding method is explained in detail in a step-by-step manner. The halftone image compression method is based on Section 2.1. Suppose the size of the image is $N \times N$. The original halftone image $\mathcal{I}$ has a size of N by N, and the number of pixels is equal to the image size. This is because each pixel is represented by one bit. Assuming that the position of a given pixel in the image is (i, j), the range of i and j is $\{1 \leq (i \,\&\, j) \leq N\}$. To encode the halftone image, the exclusive OR operation between the original pixel value and the pseudorandom number bits $ek_{i,j}$ is calculated according to Equation (6).

$$\tilde{\mathcal{I}} = \mathcal{I}_{i,j} \oplus ek_{i,j} \tag{6}$$

where $\mathcal{I}_{i,j}$ is the pixel value of the cover image $\mathcal{I}$, and $ek_{i,j}$ is generated by the stream cipher using the encryption key (EK). For example, you can use a secure stream cipher such as RC4.

3.2. Embedding Procedure

This section explains the procedure for hiding data based on encrypted duplicate images. The owner encrypts the original image and transmits it to the SP. A diagram of the data embedding procedure is shown in Figure 2. Given a cover image $\mathcal{I}$, the service provider copies $\mathcal{I}$ to create two identical cover images: $\mathcal{I}_1$ and $\mathcal{I}_2$.

Input : Two cover images, $\tilde{\mathcal{I}}_1$ and $\tilde{\mathcal{I}}_2$, and secret bits $m = (m_1, m_2, \ldots, m_n)$
Output: Two marked images, $\tilde{\mathcal{I}}'_1$ and $\tilde{\mathcal{I}}'_2$.

Step 1: Two cover images, $\tilde{\mathcal{I}}_1$ and $\tilde{\mathcal{I}}_2$, are divided into nonoverlapping blocks of size $\mathcal{N} = M \times M$. The number of blocks in each image is $n = N^2/M^2$, and the variable i stands for $\{1 \leq i \leq n\}$.

Step 2: Blocks B_{I_1} and B_{I_2} of size $\mathcal{N}$ are read from the two cover images, $\tilde{\mathcal{I}}_1$ and $\tilde{\mathcal{I}}_2$, and transformed into one-dimensional vectors (codewords) y_{I_1} and y_{I_2}. The three secret bits (m_i) are encrypted as:

$$\xi_j = m_j \oplus k_j \tag{7}$$

In this case, ξ_j is an encrypted secret bit, and $k_i = \sum_{j=1}^{\mathcal{N}} (B_j \times j) \; mod \; 3 + 1$.

Step 3: The syndrome for the codeword y is calculated according to the formula:

$$\eta = \mathcal{H} \cdot y^T \tag{8}$$

Step 4: The new syndrome is computed by embedding three bits: $\eta'_1 = \eta_1 \oplus \xi$. Then, η'_1 is converted to decimals, $D(\eta'_1)$, as follows:

$$y_j = \begin{cases} y_j, & if \; j \neq D(\eta') \\ 1 - y_j, & if \; j = D(\eta') \end{cases} \tag{9}$$

where $\{1 \leq j \leq \mathcal{N}\}$ and $j = 1, 2, \ldots, 7$.

Step 5: After calculating the syndrome for the codeword y_{I_2}, the syndrome η'_1 is hidden in y_{I_2}. That is, $\eta'_2 = (\mathcal{H} \times y) \oplus \eta'_1$. Then, η'_2 is converted to decimals, $D(\eta'_2)$, and put it into Equation (9).

Step 6: The values at the corresponding position of the images $\tilde{\mathcal{I}}_1$ and $\tilde{\mathcal{I}}_2$ are replaced by the pixels y_{I_1} and y_{I_2}.

Step 7: Steps 2, 3, ..., 6 are performed while the variable i is less than n.

3.3. Extraction and Recovery Procedure

This section describes the process by which the receiver decodes the image, extracts the extra bits, or does both, depending on the three situations on the receiver side.

3.3.1. Image Decryption

If the receiver has only the encryption key (EK), it can decrypt the marked images $\tilde{\mathcal{I}}_1$ and $\tilde{\mathcal{I}}_2$ without extracting the watermark. The image decoded directly from $\tilde{\mathcal{I}}'$ is called a marked image $\mathcal{I}'$ and is generated by the following formula:

$$\mathcal{I}'_{i,j} = \tilde{\mathcal{I}}'_{i,j} \oplus ek_{i,j} \tag{10}$$

In this case, $ek_{i,j}$ generates the same stream cipher as in Section 3.2 with the same key EK.

3.3.2. Extraction Procedure

If only the data hiding key (DK) is assigned to the receiver, secret data can be extracted from the cover image. The details are explained in a step-by-step manner.

Input: Two marked images, $\tilde{\mathcal{I}}_1'$ and $\tilde{\mathcal{I}}_2'$, and the data key (DK).
Output: The secret bits $m = (m_1, m_2, \ldots, m_n)$.

Step 1: Two marked images, $\tilde{\mathcal{I}}_1'$, are divided into nonoverlapping $\mathcal{N} = M \times M$ blocks. The number of blocks in each image in this case is $n = (N^2/M^2)$.

Step 2: For the marked images, $\tilde{\mathcal{I}}_1'$ and $\tilde{\mathcal{I}}_1'$, blocks B_{I_1} and B_{I_2} of size $\mathcal{N}$ are read and transformed into one-dimensional vectors, y_{I_1} and y_{I_2}. We obtain the syndrome as $\eta_1 = \mathcal{H} \times y_{I_1}$ and $\eta_2 = \mathcal{H} \times y_{I_2}$ using Equation (8). The obtained syndrome η_1 is the encoded hidden three bits.

Step 3: The syndrome η_2 is an error made in the codeword y_{I_1}. Therefore, the error can be recovered by applying η_2 to Equation (9). Then, after obtaining the data encryption key $k_i = \sum_{j=1}^{M \times M}(B_j \times j)\ mod\ 3 + 1$, decoding is conducted with 3 bits of data ξ, i.e., $m_i = \xi_i \oplus k_i$.

Step 4: Steps 1 to 3 are repeated when the variable i is less than n.

3.3.3. Data Extraction and Image Decryption

Using DK and EK, the receiver can extract the hidden bits from the marked image $\mathcal{I}_1$ and restore the original halftone image $\mathcal{I}$. In this section, the receiver first extracts the encrypted secret message from the marked image I_1 and then uses image I_2 to recover image I_1, restoring image I_2.

Input: Two marked images, $\tilde{\mathcal{I}}_1'$ and $\tilde{\mathcal{I}}_2'$, the data hiding key (DK), and the encryption key (EK).
Output: A secret bit stream m and the original halftone image $\mathcal{I}$.

Step 1: Two marked images $\tilde{\mathcal{I}}_1'$ are each partitioned into nonoverlapping blocks of size $\mathcal{N} = M \times M$. The number of blocks in each image is $n = (N^2/M^2)$.

Step 2: For the marked images, $\tilde{\mathcal{I}}_1'$ and $\tilde{\mathcal{I}}_2'$, blocks B_{I_1} and B_{I_2} of size $\mathcal{N}$ are read and transformed into one-dimensional vectors, y_{I_1} and y_{I_2}. We obtain the syndrome as $\eta_1 = \mathcal{H} \times y_{I_1}$ and $\eta_2 = \mathcal{H} \times y_{I_2}$ using Equation (8). The obtained syndrome η_1 is the encoded hidden three bits.

Step 3: The syndrome η_2 is an error made in the codeword y_{I_1}. Therefore, the error can be recovered by applying η_2 to Equation (9). Then, after obtaining the data encryption key $k_i = \sum_{j=1}^{M^2}(B_j \times j)\ mod\ 3$, decoding is conducted with 3 bits of data ξ, i.e., $m_i = \xi_i \oplus k_i$. The 3 bits are added to the vector φ.

$$\varphi = \varphi || m_i \tag{11}$$

Step 4: The recovered codeword is copied (y_{I_1}) to y_{I_2}. Two images can be recovered by replacing y_{I_1} and y_{I_2} with pixels at the corresponding positions of the marked images $\mathcal{I}_1$ and $\mathcal{I}_2$.

Step 5: The procedures described in steps 2 to 4 are applied repeatedly as long as i is less than n. If i is greater than n, proceed to step 6.

Step 6: Here, the cover image is recovered by the XOR operation of the two-dimensional vector ek generated with the encryption key (EK) and the images $\tilde{\mathcal{I}}_1$. It is generated by the following formula.

$$\mathcal{I}_{i,j} = \tilde{\mathcal{I}}_{i,j} \oplus ek_{i,j} \tag{12}$$

3.4. Examples

3.4.1. Data Embedding

Example 1. *Make a copy of the original image $\mathcal{I}$ as $\mathcal{I}_2$ and use $\mathcal{I}$ as $\mathcal{I}_1$. Suppose a block read from the original image $\mathcal{I}_1$ is $B_i = [1\ 0\ 0\ 1\ 1\ 1\ 0]$, the codeword is $h = [1\ 0\ 0\ 1\ 1\ 1\ 0]$. And, the secret bit is $m = [1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1]$. $ek_j = [1\ 0\ 1\ 0\ 1\ 0\ 1]$ is generated with the EK for image encryption.*

(1) *The encrypted codeword becomes* $\tilde{y}_1 = y_1 \oplus ek_j = [1\ 0\ 0\ 1\ 1\ 1\ 0] \oplus [1\ 0\ 1\ 0\ 1\ 0\ 1] = [0\ 0\ 1\ 1\ 0\ 1\ 1]$ *and* $\tilde{y}_2 = y_2 \oplus ek_j = [1\ 0\ 0\ 1\ 1\ 1\ 0] \oplus [1\ 0\ 1\ 0\ 1\ 0\ 1] = [0\ 0\ 1\ 1\ 0\ 1\ 1]$.

(2) *The syndrome for the codeword* $\tilde{y}_1$ *is computed. That is,* $\eta_1 = \mathcal{H} \cdot \tilde{y}_1^T = \mathcal{H} \cdot [0\ 0\ 1\ 1\ 0\ 1\ 1]^T = [1\ 1\ 0]^T$.

(3) *For the secret bit S, encryption is performed using the key* k_j. *That is,* $\xi = m_i \oplus k_i = [1\ 0\ 1] \oplus [0\ 1\ 0] = [1\ 1\ 1]$. *Then, a new syndrome is computed. That is,* $\eta' = [1\ 1\ 0] \oplus [1\ 1\ 1] = [0\ 0\ 1]$.

(4) *The pixel corresponding to the position of syndrome* η_1' *of the codeword* $\tilde{y}_1$ *is flipped. That is,* $\tilde{y}_1' = [1\ 0\ 1\ 1\ 0\ 1\ 1]$.

(5) *The syndrome* η_1' *is hidden in the codeword* $\tilde{y}_2$. *That is,* $\eta_2' = (\mathcal{H} \cdot \tilde{y}_2) \oplus \eta_1' = [1\ 1\ 0] \oplus [0\ 0\ 1] = [1\ 1\ 1]$. *The pixel corresponding to the position of the syndrome* η_2' *of the codeword* $\tilde{y}_2$ *is flipped. That is,* $\tilde{y}_2' = [0\ 0\ 1\ 1\ 0\ 1\ 0]$.

3.4.2. Data Extraction and Image Restoration

Example 2. *The receiving side receives encrypted marked images* $\tilde{\mathcal{I}}_1'$ *and* $\tilde{\mathcal{I}}_2'$. *The key generated with the EK for image encryption is* $ek_j = [1\ 0\ 1\ 0\ 1\ 0\ 1]$. *Assuming that a block read from the image* $\tilde{\mathcal{I}}_1'$ *is* $B_{I_1} = [1\ 0\ 1\ 1\ 0\ 1\ 1]$, *and the codeword is* $\tilde{y}_{I_1} = [1\ 0\ 1\ 1\ 0\ 1\ 1]$. *The codeword of a block read from* $\tilde{\mathcal{I}}_2'$ *becomes* $\tilde{y}_{I_2} = [0\ 0\ 1\ 1\ 0\ 1\ 0]$.

(1) *Compute the syndrome for the codeword* $\tilde{h}_1$. *That is,* $\eta_1 = \mathcal{H} \cdot \tilde{y}_1^T = \mathcal{H} \cdot [1\ 0\ 1\ 1\ 0\ 1\ 1]^T = [1\ 1\ 1]^T$.

(2) *If the syndrome is calculated for the codeword* $\tilde{y}_2$, *i.e.,* $\eta_2 = \mathcal{H} \cdot \tilde{y}_2^T = \mathcal{H} \cdot [0\ 0\ 1\ 1\ 0\ 1\ 0] = [0\ 0\ 1]$. *If the syndrome* η_2 *is applied to* y_1' *using Equation (9), the error pixel can be restored. That is,* $y_1 = [0\ 0\ 1\ 1\ 0\ 1\ 1]$. *The encrypted bits are recovered using the data key* $k_i = [0\ 1\ 0]$. *That is,* $m_i = \xi \oplus k_i = [1\ 1\ 1] \oplus [0\ 1\ 0] = [1\ 0\ 1]$

4. Experimental Results

In this section, we evaluate the performance of our proposed method in terms of the data hiding capacity, image quality, and ability to recover the original images. Our proposed experimental model selected nine images from the standard USC-SIPI image [44] database and used them for the experiment. The selected grayscale images were converted to halftone images using Floyd–Steinberg dithering. An experiment using nine 512×512 halftone images of Baboon, Barbara, Boat, Goldhill, Airplane, Lena, Peppers, Tiffany, and Zelda was conducted, as shown in Figure 3.

To perform a comprehensive evaluation of the halftone images, we used a special type of filter that simulates the perception of images by the human visual system (HVS). Since it is not possible to evaluate halftone images with the conventional PSNR, it is necessary to convert them into grayscale images with special filters and then to try an evaluation with PSNR. This filter, called a Gaussian low-pass filter (LPF), is characterized by a 7×7 square matrix and a standard deviation of 2.0. The use of this filter allowed us to quantitatively measure the visual quality of the images. The images examined were 512×512 halftone images generated by a special process called Steinberg kernel error diffusion dithering applied to an 8-bit grayscale image. For more details, see Figure 3.

In the experiments, the modified peak signal-to-noise ratio (MPSNR) (Equation (15)) was used as an objective quality measure, which corresponds to the PSNR (Equations (13) and (14) between the multitone cover image and the low-pass filtered marked halftone images.

PSNR is the most popular criterion for measuring the distortion between the cover image and the marked images. It is defined as follows:

$$PSNR(\mathcal{I}, \tilde{\mathcal{I}}) = 20 \cdot log_{10}\left(\frac{MAX}{\sqrt{MSE}}\right), \tag{13}$$

where MSE is the mean square error between the original grayscale image and the shadow image:

$$MSE = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}[\mathcal{I}_{i,j} - \tilde{\mathcal{I}}_{i,j}]^2, \tag{14}$$

The symbols $\mathcal{I}_{i,j}$ and $\tilde{\mathcal{I}}_{i,j}$ stand for the pixel values of the original grayscale image and the marked image at the respective positions, respectively, and $N \times N$ is the width and height of the original image.

Figure 3. Test images: (**a**) Baboon, (**b**) Barbara, (**c**) Boat, (**d**) Goldhill, (**e**) Airplane, (**f**) Lena, (**g**) Peppers, (**h**) Tiffany, and (**i**) Zelda (512×512).

The MPSNR is a quality measure that attempts to model the human visual system. First, a simple inverse halftone $\tilde{\mathcal{I}}_{low}$ is generated with a low-pass filter. For our purposes, a 7×7 Gaussian low-pass filter has been proven to be useful. The matrix $\tilde{\mathcal{I}}_{low}$ was then input to the PSNR function to generate the MPSNR (see Equation (15)). This function allows the automatic testing of algorithms. Note that the MPSNR, like the PSNR, measures the relative visual quality, which means that MPSNR measurements can only be used to compare variants of the same image.

$$MPSNR(\mathcal{I}, \tilde{\mathcal{I}}) = PSNR(\mathcal{I}, \tilde{\mathcal{I}}_{low}) \tag{15}$$

The dB value of each image in Figure 3 shows the quality of the cover image by measuring the MPSNR between the original gray image and the halftone cover image restored to a grayscale image.

Another performance measure is SSIM, a formula (Equation (16)) that measures the similarity between the original image and the marked image.

$$SSIM(\mathcal{I}, \tilde{\mathcal{I}}) = \frac{(2\mu_{\mathcal{I}}\mu'_{\mathcal{I}} + c_1)(2\sigma_{\mathcal{I}\tilde{\mathcal{I}}} + c_2)}{(\mu_{\mathcal{I}}^2 + \mu_{\mathcal{I}}'^2 + c_1)(\sigma_{\mathcal{I}}^2 + \sigma_{\mathcal{I}}'^2 + c_2)} \tag{16}$$

where $\mu_{\mathcal{I}}$ and $\mu'_{\mathcal{I}}$ are the mean values of $\mathcal{I}$ and $\tilde{\mathcal{I}}$, respectively, c_1 is the stabilization constant, and $\mu_{\mathcal{I}}^2$, $\mu_{\tilde{\mathcal{I}}}^2$, and $\sigma_{\mathcal{I}\tilde{\mathcal{I}}}$ are the variances and covariances of the cover image and the marked image. c_1 and c_2 are constant values that are used to avoid problems with division by zero.

Table 1 compares the PSNR of the cover halftone image and the two marked halftones with the original image. Each block is divided by 16×16, and Hamming code is applied to hide 3 bits of data.

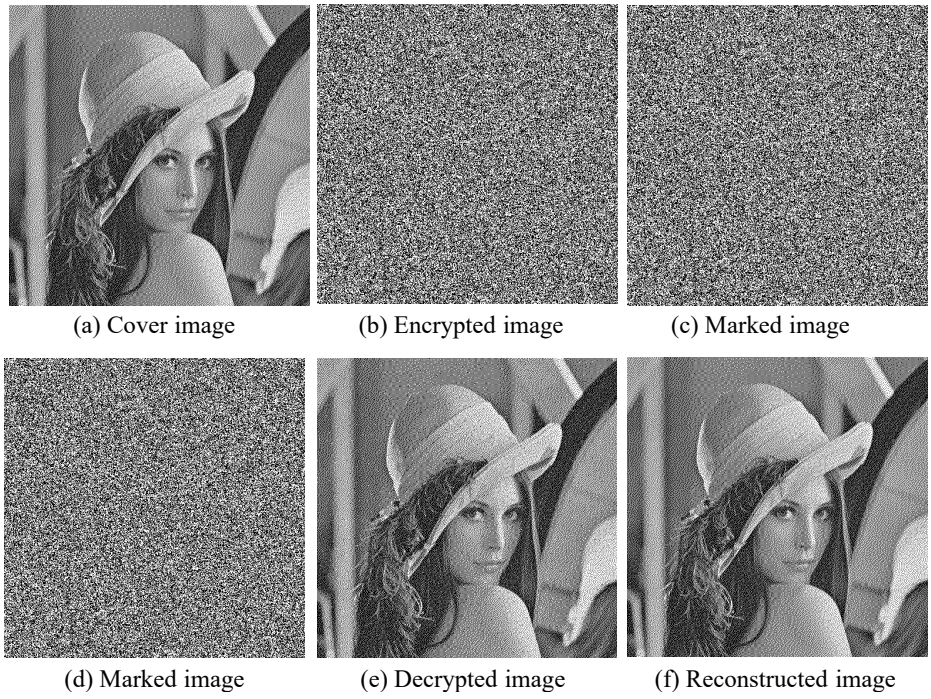
Table 1. Comparisons of the PSNR and SSIM for dual decrypted halftone images.

Images	$\text{PSNR}_{(O-I)}$	$\text{SSIM}_{(O-I)}$	EC	$\text{PSNR}_{(O-I')}$ #1	$\text{PSNR}_{(O-I')}$ #2	$\text{PSNR}_{(I-I')}$ #1	$\text{PSNR}_{(I-I')}$ #2
Baboon	22.555	0.5918	3072	22.0782	22.0682	35.6489	35.635
Barbara	21.8521	0.6584	3072	21.4894	21.4849	35.2901	35.2795
Boat	28.0218	0.7645	3072	27.2214	27.2226	36.1873	36.2187
Goldhill	28.5819	0.7372	3072	27.9131	27.905	36.5539	36.523
Airplane	28.6479	0.8102	3072	27.6455	27.6503	36.2903	36.3067
Lena	30.1849	0.8075	3072	29.3125	29.3316	36.7046	36.7221
Peppers	30.4007	0.7968	3072	29.511	29.5013	36.6943	36.6599
Tiffany	28.7022	0.7179	3072	28.1571	28.1554	36.1126	36.0712
Zelda	31.9277	0.8045	3072	31.3182	31.3328	37.0002	36.9989
Average	27.8749	0.7432	3072	27.1829	27.1836	36.2758	36.2683

To hide data, a procedure for generating an error for 1 bit of data is required for each block. Of course, once the data have been extracted, the errors can be recovered by applying the RDH algorithm. Table 1 shows the actual hidden bit size under this condition.

$\text{PSNR}_{(O-\mathcal{I})}$ and $\text{SSIM}_{(O-\mathcal{I})}$ measure the PSNR and SSIM, respectively, between the cover image and the original image. Since the halftone image is an image compressed by one-eighth of the gray image, the original image cannot be reproduced perfectly, but it is reproduced at a relatively high level. $PSNR(\mathcal{I} - \mathcal{I}')$#1 was used to measure the PSNR between the cover image and the marked image#1. As a result of the measurement, a high PSNR of more than 35 dB was measured. $PSNR(\mathcal{I} - \mathcal{I}')$#2 is the same as $PSNR(\mathcal{I} - \mathcal{I}')$#1, but the PSNR evaluation for the second image. $PSNR(\mathcal{O} - \mathcal{I}')$ is a measure of the PSNR between an original (grayscale) image and a marked (grayscale) image. In the case of the Lena image, the quality of the recovered marked image was 29.5013 dB, slightly less than 30 dB, but showing a result close to the original image. In the case of images with high frequencies, such as Baboon, it is relatively difficult to convert them to grayscale images, so the specific image quality is not high. The quality of the marked image can be improved by proper control of the hidden data, and conversely, a sufficiently large amount of data can be hidden by reducing the block size.

Figure 4a is the original halftone, and Figure 4b is the encoded image for the original image. Figure 4c,d is an image with additional data hidden in an encoded image. Figure 4e is the decoded image with two marked images. Figure 4f is the recovered image after data extraction with two marked images using the reconstruction method. Figure 4e illustrates the visual consequences of embedding 3 bits of data in a 16×16 block, resulting in a 1-pixel error. When testing with an 8-bit pixel image, it is common to hide the data by flipping the least significant bit (LSB). This method usually results in minimal distortion of the image. However, with an image consisting of 1 bit of pixels, some degree of damage is inevitable, as the figure shows. Despite these challenges, the proposed approach uses reversible data hiding (RDH) to recover the original cover image, as shown in Figure 4f. This recovery feature somewhat compensates for the drawback of image distortion. To restore a grayscale image, we can resort to Gaussian filtering [45]. Note, however, that this technique may not completely eliminate the strong salt-and-pepper noise. Effective recovery of a grayscale image is crucial to ensure accurate measurement of the peak signal-to-noise ratio (PSNR).



(a) Cover image (b) Encrypted image (c) Marked image

(d) Marked image (e) Decrypted image (f) Reconstructed image

Figure 4. Lena image when the size of the blocks was 8×8.

Table 2 shows the results of measuring the data hiding capacity (embedding capacity or EC) and peak signal-to-noise ratio (PSNR) when the block size $M \times M$ was set to 6×6, 8×8, 16×16, and 32×32. As the block size increased, the EC decreased while the dB value of the PSNR increased. As a result, the quality of the marked image approached the quality of the cover image. For example, with a block size of 6×6, the EC was 21,845 bits, which correlates with a relatively low PSNR value. When the block size increased to 8×8, the changes in the EC and PSNR were not high. In contrast, with a block size of 32×32, we were able to hide 768 bits (EC) with a corresponding PSNR value of 44 dB, indicating that the image quality was close to that of the cover image. The choice of block size may be guided by specific use cases—larger blocks may be preferred when a larger amount of data needs to be hidden or when the quality of the displayed image is of paramount importance.

Table 2. Comparison of the PSNR and SSIM of dual images according to the block sizes.

Images	Block Size	EC	$SSIM_1$	$SSIM_2$	$PSNR_1$	$PSNR_2$
Baboon	6×6	21845	0.8048	0.8029	28.9734	28.9522
	8×8	12,288	0.8591	0.8728	32.3177	31.2014
	16×16	3072	0.9623	0.9482	38.4097	35.6341
	32×32	768	0.9907	0.9679	44.4792	38.1529
Barbara	6×6	21845	0.7978	0.7981	28.5111	28.4622
	8×8	12,288	0.8835	0.8644	31.7912	30.8172
	16×16	3072	0.9688	0.9417	37.9815	35.2427
	32×32	768	0.9921	0.964	44.0283	37.7172
Boat	6×6	21,845	0.7707	0.7704	29.0851	29.0656
	8×8	12,288	0.8641	0.8494	32.1207	31.4613
	16×16	3072	0.964	0.9425	38.3214	36.2107
	32×32	768	0.9909	0.9671	44.3455	39.1038
Goldhill	6×6	21,845	0.7772	0.7771	29.2395	29.1847
	8×8	12,288	0.8696	0.8546	32.1959	31.599
	16×16	3072	0.9652	0.944	38.3211	36.537
	32×32	768	0.9915	0.9682	44.3402	39.5786

Table 2. *Cont.*

Images	Block Size	EC	$SSIM_1$	$SSIM_2$	$PSNR_1$	$PSNR_2$
Airplane	6 × 6	21,845	0.7677	0.7679	29.0986	29.1266
	8 × 8	12,288	0.8626	0.8463	32.1791	31.5276
	16 × 16	3072	0.9633	0.9405	38.4126	36.3031
	32 × 32	768	0.9908	0.9662	44.3968	39.073
Lena	6 × 6	21,845	0.7614	0.7614	29.3758	29.3754
	8 × 8	12,288	0.8593	0.8428	32.3345	31.7695
	16 × 16	3072	0.9628	0.9403	38.4455	36.7003
	32 × 32	768	0.9906	0.9669	44.4566	39.7725
Peppers	6 × 6	21,845	0.7592	0.7597	29.3592	29.3409
	8 × 8	12,288	0.8579	0.8433	32.2904	31.758
	16 × 16	3072	0.9615	0.9413	38.4548	36.6804
	32 × 32	768	0.9905	0.9674	44.4372	39.8912
Tiffany	6 × 6	21,845	0.7612	0.7615	28.519	28.5294
	8 × 8	12,288	0.8567	0.8377	31.9722	31.0842
	16 × 16	3072	0.9608	0.936	38.239	36.0513
	32 × 32	768	0.9901	0.9631	44.2637	39.0831
Zelda	6 × 6	21,845	0.7486	0.7504	29.329	29.371
	8 × 8	12,288	0.8529	0.8353	32.3256	31.8625
	16 × 16	3072	0.9608	0.9391	38.5188	37.0048
	32 × 32	768	0.9899	0.9668	44.5039	40.3822

Figure 5 shows a visual comparison of the quality of the original image, the cover image, and the marked image used to hide data. The marked image was obtained by first encoding the cover image, dividing it into specific block sizes (in this case, 16 × 16 blocks), and then applying our proposed data hiding algorithm. Figure 5d is a grayscale image recovered from the cover image that is visually very similar to the original image. Figure 5e is an 8-bit grayscale image obtained by applying the image decoding algorithm to the marked image. Figure 5f shows the image obtained after applying the data extraction and reversible data hiding (RDH) methods to the marked image and then applying the image decoding algorithm. The peak signal-to-noise ratio (PSNR) was used to measure the quality of an image. For this purpose, a halftone image must be converted to an 8-bit grayscale image. Even though our proposed method is RDH, we recommend leaving the block size at 16 or more. This ensures a certain level of image quality, even if only the decryption of the encrypted marked image is performed.

Figure 6 shows that the embedding capacity of the proposed scheme is larger than those of the previous schemes, i.e., Pan et al. (2007) [34], Tsai (2009) [35], Xuan et al. (2008) [36], Kim et al. (2013) [37], and Yin et al. (2021) [38].

The methods shown in Figure 6 are generally implemented as reversible data hiding (RDH) algorithms based on halftone images. Apart from our proposed method, the existing techniques use a histogram-based 4 × 4 or run-length pattern to hide data. More specifically, they use either the most frequent pattern or the least frequent pattern next to an empty pattern to hide the data. Hiding data using histograms often results in a smaller amount of clean data, because conventional payload data are required for image recovery. Histogram correction is the best known and most commonly used method for RDH in grayscale images. Previous studies have demonstrated the feasibility of restoring halftone images by applying the histogram shift method, an optimal technique for restoring grayscale images to halftone images. Unlike these methods, our proposed technique can hide data in all given blocks, since it does not rely on a particular pattern. In contrast, hiding data with a histogram is only able to hide data in a given pattern, which limits the amount of data that can be hidden. Figure 6 shows an example that illustrates these results.

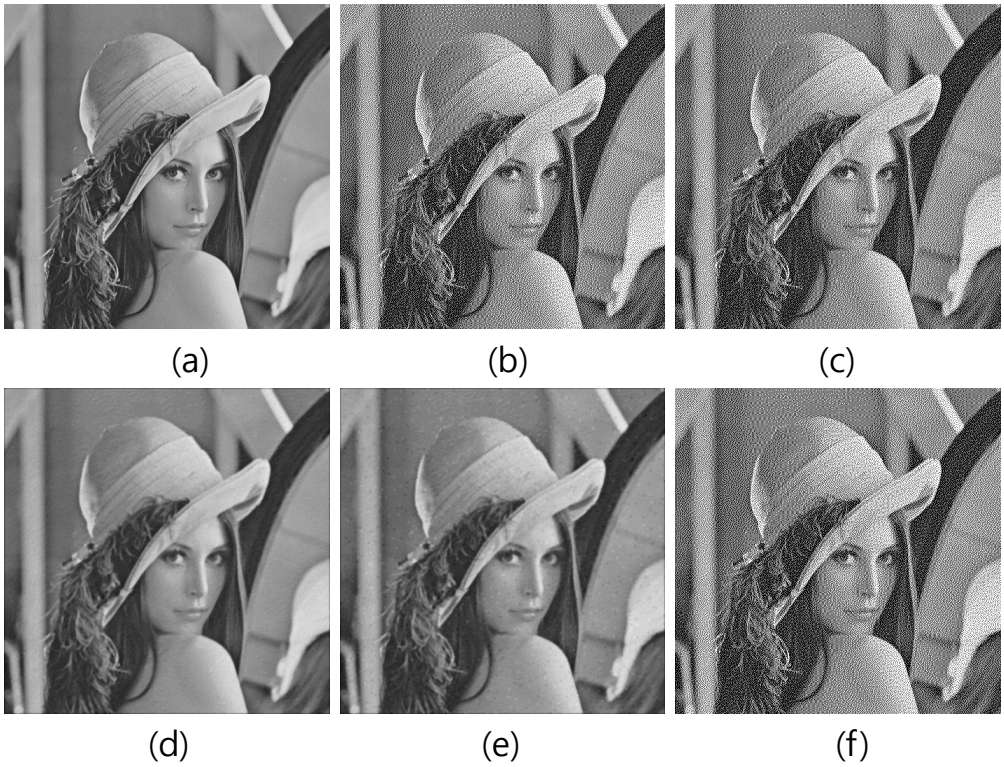


Figure 5. Restoration of an error diffusion image to a grayscale image and comparison of the visual quality between images: (**a**) original image, (**b**) halftone (cover) image, (**c**) halftone (marked) image (DH: 16 × 16 block), (**d**) inverted image with (**b**), (**e**) inverted image with (**c**), and (**f**) reconstructed image from (**c**); 7 × 7 Gaussian filter, $\sigma = 1.2$.

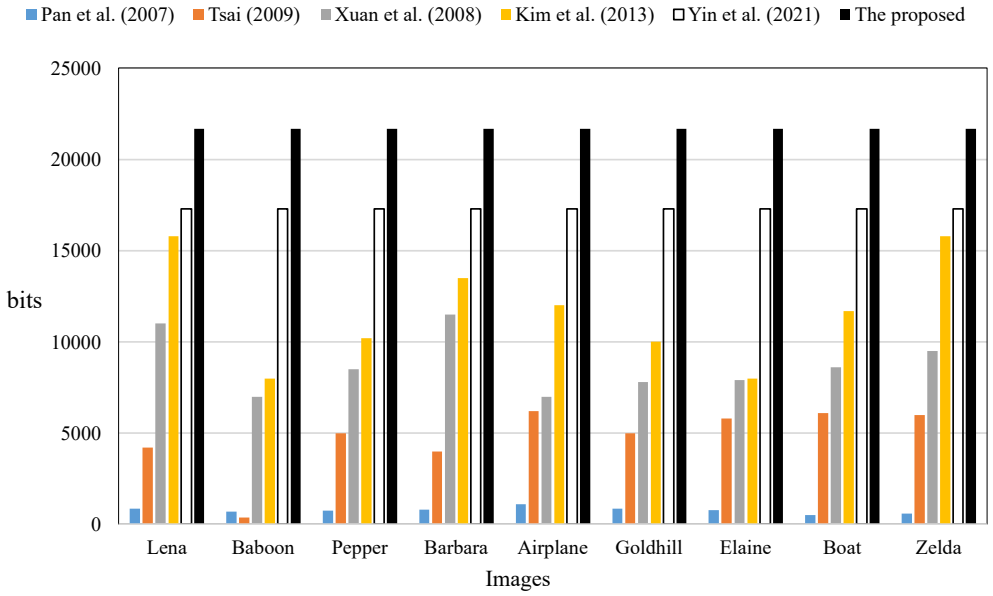


Figure 6. Comparison of the embedded bits between previous methods and the proposed method [34–38].

Table 3 provides a functional comparison between the proposed method and other related methods widely used in the field. One of the unique features of our proposed method is the use of dual halftone images using RDH-EI. This aspect distinguishes it from most of the existing technologies that traditionally use grayscale images [26,27,30]. The use of the dual RDH-EI provides an additional layer of security by contributing to increased

robustness against potential attacks. In addition, our method demonstrates true reversibility, an important property that is often overlooked in the field of image steganography. The benefits of true reversibility are twofold. That is, it guarantees the integrity of the original data after extraction and offers the advantage that the carrier image can be fully recovered without any detectable distortions. This is particularly important as it protects the quality of the halftone image after extraction and preserves its esthetic value in addition to its data storage capacity. In terms of the embedding capacity, the proposed method exhibits high efficiency for halftone images [25,32]. Efficient data hiding methods can contain data more efficiently and maintain a high level of data fidelity. As summarized in Table 3, our method outperforms existing techniques by providing a surprisingly high embedding capacity for halftone images. These unique features not only solve some key problems in the field, but also open new avenues for improvement.

Table 3. Feature comparisons among the proposed method and related methods.

Methods	Image Type	# of Cover Image	DH Capacity	Encryption	Reversibility
Lu et al. (2015) [26]	Grayscale	2	High	no	yes
Yao et al. (2017) [27]	Grayscale	2	High	no	yes
Jana et al. (2018) [30]	Grayscale	2	Low	no	yes
Kim et al. (2018) [25]	Halftone	1	Low	yes	no
Sun et al. (2020) [32]	Halftone	1	Low	yes	yes
Our proposed method	Halftone	2	High	yes	yes

5. Conclusions

In this paper, we proposed a method for reversible data hiding (RDH) based on an encrypted dual halftone image using the HC (7,4). The method involves extracting data from the marked image and then image decoding to recover the original cover image (i.e., the halftone image). An advantage of our proposed method is the flexible adjustment of the block size ($M \times M$). This feature allows us to embed extensive data while enabling the production of high-quality marked images. From a security perspective, the use of the dual RDH provides a notable advantage over traditional RDH methods in that it is more robust. This strength is primarily due to the need to acquire both cover images to recover the embedded data. Moreover, since each pixel of the halftone image consists of 1 bit of data, our method has the advantage of enabling fast transmission over low-power networks. Thus, our proposed approach to obfuscate data in encrypted images not only ensures sufficient invisibility of the data for transmission to the receiver, but also eliminates the need to disturb the visible image or subsequently recover the cover image. However, applying only the decoding algorithm to the stego image may result in an unsatisfactory image quality due to a distorted, marked image. However, this problem can be circumvented by applying a data extraction and image restoration algorithm that produces a restored image. The experimental results confirm that our proposed method has a high embedding capacity and moderate computational complexity while ensuring image reversibility. The image quality was evaluated by converting halftone images to grayscale images using a Gaussian low-pass filter (LPF). In future research, we aim to explore techniques superior to low-pass filtering to improve the image quality when converting halftone images to 8-bit gray images.

Author Contributions: Each author discussed the details of the manuscript. C.K. designed and wrote the manuscript. C.K. implemented the proposed technique and provided the experimental results. N.-N.D., K.-H.J. and L.L. reviewed and revised the article. L.L. and C.K. drafted and revised the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) under a grant funded by the Korea government (MSIT) (No. 2021R1G1A1008105) (N.D.). This research was supported by the Brain Pool program funded by the Ministry of Science and ICT through the National Research Foundation of Korea (2019H1D3A1A01101687) and Basic Science Research Program

through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2021R1I1A3049788) (K.J.). This research was supported by the National Natural Science Foundation of China (61866028), Technology Innovation Guidance Program Project (Special Project of Technology Cooperation, Science and Technology Department of Jiangxi Province) (20212BDH81003) (L.L.)

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are unavailable.

Acknowledgments: We thank the anonymous reviewers for their valuable suggestions that improved the quality of this article.

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

Abbreviations

The following abbreviations are used in this manuscript:

DH	Data Hiding.
RDH	Reversible Data Hiding.
Dual RDH	Dual Reversible Data Hiding.
HC	Hamming Code.
$\mathcal{I}_1$	Cover Image 1.
$\mathcal{I}_2$	Cover Image 2.
$\tilde{\mathcal{I}}_1$	Marked Image 1.
$\tilde{\mathcal{I}}_2$	Marked Image 2.
m	Secret bits.
$N \times N$	Image Size.
$M \times M$	Block Size.
ξ	Encrypted Data.
$\mathcal{H}$	A parity check matrix.
D	A function that converts a binary number to decimal number.
η	Syndrome.
y	A codeword.
B	Read Block.
k	A random binary stream cipher.
ek	A random binary stream cipher for image encryption.

References

1. Bender, W.; Gruhl, D.; Morimote, N.; Lu, A. Techniques for data hiding. *IBM Syst. J.* **1996**, *35*, 313–336.
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.
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.
5. Fridrich, J.; Goljan, M.; Du, R. Lossless data embedding—New paradigm in digital watermarking. *EURASIP J. Adv. Signal Process.* **2002**, *2002*, 986842.
6. Tian, J. Reversible data embedding using a difference expansion. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 890–896. [CrossRef]
7. Alattar, A. M. Reversible watermark using the difference expansion of a generalized integer transform. *IEEE Trans. Image Process.* **2004**, *13*, 1147–1156.
8. Ni, Z.; Shi, Y.-Q.; Ansari, N.; Su, W. Reversible data hiding. *IEEE Trans. Circuits Syst. Video Technol.* **2006**, *16*, 354–362.
9. Qin, C.; Chang, C.-C.; Huang, Y.-H.; Liao, L.-T. An inpaintingassisted reversible steganographic scheme using a histogram shifting mechanis. *IEEE Trans. Circuits Syst. Video Technol.* **2013**, *23*, 1109–1118. [CrossRef]
10. Dragoi, I.-C.; Coltuc, D. Local-Prediction-Based Difference Expansion Reversible Watermarking. *IEEE Trans. Image Process.* **2014**, *23*, 1779–1790. [CrossRef]
11. Abanda, Y.; Tiedeu, A. Image encryption by chaos mixing. *IET Image Process.* **2016**, *10*, 742–750. [CrossRef]
12. Li, C.; Lin, D.; Lü, J. Cryptanalyzing an image-scrambling encryption algorithm of pixel bits. *IEEE Trans. Multimedia* **2017**, *24*, 64–71. [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. Shamir, A. How to share a secret. *Commun. Assoc. Comput. Mach.* **1979**, *22*, 612–613. [CrossRef]
15. Naor, M.; Shamir, A. Visual cryptography. In Proceedings of the Advances in Cryptology—EUROCRYPT'94, Perugia, Italy, 9–12 May 1994; De Santis, A., Ed.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1995; Volume 950.
16. Zhang, X. Reversible data hiding in encrypted image. *IEEE Signal Process. Lett.* **2011**, *18*, 255–258. [CrossRef]
17. Zhang, X. Separable reversible data hiding in encrypted image. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 826–832. [CrossRef]
18. Wu, X.; Sun, W. High-capacity reversible data hiding in encrypted images by prediction error. *Signal Process.* **2014**, *104*, 387–400. [CrossRef]
19. Shiu, C.-W.; Chen, Y.-C.; Hong, W. Encrypted image-based reversible data hiding with public key cryptography from difference expansion. *Signal Process. Image Commun.* **2015**, *39*, 226–233. [CrossRef]
20. Yi, S.; Zhou, Y. Separable and Reversible Data Hiding in Encrypted Images Using Parametric Binary Tree Labeling. *IEEE Trans. Multimed.* **2019**, *21*, 51–64. [CrossRef]
21. 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]
22. Yin, Z.; Xiang, Y.; Zhang, X. Reversible data hiding in encrypted images based on multi-MSB prediction and Huffman coding, *IEEE Trans. Multimed.* **2020**, *22*, 874–884.
23. Mohammadi, A.; Nakhkash, M.; Akhaee, M.A. A high-capacity reversible data hiding in encrypted images employing local difference predictor. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *30*, 2366–2376. [CrossRef]
24. Feng, Q.; Leng, L.; Chang, C.C.; Horng, J.H.; Wu, M. Reversible data hiding in encrypted images with extended parametric binary tree labeling. *Appl. Sci.* **2023**, *13*, 245. [CrossRef]
25. Kim, C.; Shin, D.; Leng, L.; Yang, C.-N. Separable reversible data hiding in encrypted halftone image. *Displays* **2018**, *55*, 71–79. [CrossRef]
26. Lu, T.C.; Wu, J.H.; Huang, C.C. Dual-image-based reversible data hiding method using center folding strategy. *Signal Proc.* **2015**, *115*, 195–213. [CrossRef]
27. Yao, H.; Qin, C.; Tang, Z.; Tian, Y. Improved dual-image reversible data hiding method using the selection strategy of shiftable pixel's coordinates with minimum distortion. *Signal Proc.* **2017**, *135*, 26–35. [CrossRef]
28. Lee, C.F.; Huang, Y.L. Reversible data hiding scheme based on dual stegano-images using orientation combinations. *Telecommun. Syst.* **2013**, *52*, 2237–2247. [CrossRef]
29. Lu, T.C.; Chi, L.P.; Wu, C.H.; Chang, H.P. Reversible data hiding in dual stego-images using frequency-based encoding strategy. *Multimed. Tools Appl.* **2017**, *76*, 23903–23929. [CrossRef]
30. Jana, B.; Giri, D.; Mondal, S.K. Dual image based reversible data hiding scheme using (7,4) hamming code. *Multimed. Tools Appl.* **2018**, *77*, 763–785. [CrossRef]
31. Kang, H.; Leng, L.; Chang, C.C. Overlapped (7,4) hamming code for large-capacity and low-loss data hiding. *Multimed. Tools Appl.* **2023**. [CrossRef]
32. Sun, Y.X.; Li, Q.; Yan, B.; Pan, J.-S.; Yang, H.-M. Reversible data hiding in dual encrypted halftone images using matrix embedding. *Multimed. Tools Appl.* **2020**, *79*, 27659–27682. [CrossRef]
33. Ulichney, R.A. *Digital Halftoning*; MIT Press: Cambridge, MA, USA, 1987.
34. Pan, J.S.; Luo, H.; Lu, Z.H. Look-up table base reversible data hiding for error diffused halftone image. *Informatica* **2007**, *18*, 615–628. [CrossRef]
35. Tsai, P. Histogram-based reversible data hiding for vector quantization-compressed images. *IET Image Process.* **2009**, *3*, 100–114. [CrossRef]
36. Xuan, G.; Shi, Y.-Q.; Chai, P.; Tong, X.; Teng, J.; Li, J. Reversible binary image data hiding by run-length histogram modification. In Proceedings of the 2008 19th International Conference on Pattern Recognition, Tampa, FL, USA, 8–11 December 2008; pp. 1–4.
37. Kim, C.; Choi, Y.S.; Kim, Y.J.; Shin, D.; Shin, D.; Yang, C.N. Reversible data hiding for halftone images using histogram modification. *Information* **2013**, *16*, 1861–1872.
38. Yin, X.L.; Lu, W.; Liu, W.T.; Guo, J.M.; Huang, J.W.; Shi, Y.G. Reversible Data Hiding in Halftone Images Based on Dynamic Embedding States Group. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 2631–2645. [CrossRef]
39. Fu, M.S.; Au, O.C. Data hiding watermarking for halftone image. *IEEE Trans. Image Process.* **2002**, *11*, 477–484. [PubMed]
40. 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.
41. Moon, T.K. *Error Correction Coding–Mathematical Methods and Algorithms*; John Wiley & Sons: Hoboken, NJ, USA, 2005; pp. 2001–2006.
42. Bulan, O.; Sharma, G.; Monga, V. Orientation modulation for data hiding in clustered-dot halftone prints. *IEEE Trans. Image Process.* **2010**, *19*, 2070–2084. [CrossRef]
43. Floyd, R.W.; Steinberg, L. An adaptive algorithm for spatial grey scale. *Proc. Soc. Inf. Disp.* **1976**, *17*, 75–77.

44. Image Databases. Available online: https://www.imageprocessingplace.com/root_files_V3/image_databases.htm (accessed on 5 January 2023).
45. Gonzalez, R.C.; Woods, R.E. *Digital Image Processing*, 2nd ed.; Prentice-Hall, Inc.: Upper Saddle River, NJ, USA, 2002.