applied sciences

MDPI

Article

Joint Hamming Coding for High Capacity Lossless Image Encryption and Embedding Scheme †

Yi-Hui Chen [1,2,‡], Pei-Yu Lin [3,‡], Hsin-Pei Wu [4,*,‡] and Shih-Hsin Chen [5,*,‡]

[1] Department of Information Management, Chang Gung University, Taoyuan City 33302, Taiwan; cyh@gap.cgu.edu.tw
[2] Kawasaki Disease Center, Kaohsiung Chang Gung Memorial Hospital, Kaohsiung 83301, Taiwan
[3] Department of Electrical Engineering, National Kaohsiung University of Science and Technology, Kaohsiung 80778, Taiwan; pagelin3@gmail.com
[4] Department of Business Administration, Soochow University, Taipei 10048, Taiwan
[5] Department of Computer Science and Information Engineering, Tamkang University, New Taipei City 251301, Taiwan
* Correspondence: bessiewu@scu.edu.tw (H.-P.W.); shchen@mail.tku.edu.tw (S.-H.C.)
† This paper is an extended version of our paper published in the work: Integrated Hamming Coding Operation to Reversible Data Hiding Scheme for Encrypted Images.
‡ These authors contributed equally to this work.

Abstract: Encryption is a widely used solution to prevent privacy leakage and illegal spread when sensitive images are uploaded to cloud storage. Hiding technology also allows confidential data to be embedded into encrypted images for secret communication. As image accuracy without distortion is essential within certain fields (such as medicine and the military), sensitive images must be completely decrypted back into the original images. However, an encrypted image is a noise-like pattern that is meaningless to a user; thus, it is difficult for a user to find the accurate image they desire. Take keywords as search indexes and embed them in encrypted images for encrypted image retrieval as an example. This idea has been extended by Chen and Line's scheme to achieve higher capacity with reversibility. The proposed scheme adjusts the coding results according to smooth and complex images to increase its hiding capacity. In addition, two thresholds are designed to adjust the predicted pixel value to be close to the original one. Experiments show that compared with the other schemes, the proposed method achieves superior results. In addition, a hidden encrypted image can be extracted from the cover image. Afterward, the hidden secrets can be completely extracted, and sensitive images can also be perfectly restored.

Keywords: reversible data embedding; privacy protection; privacy leakage; image encryption; hamming coding

Citation: Chen, Y.-H.; Lin, P.-Y.; Wu, H.-P.; Chen, S.-H. Joint Hamming Coding for High Capacity Lossless Image Encryption and Embedding Scheme. *Appl. Sci.* **2022**, *12*, 1966. https://doi.org/10.3390/app12041966

Academic Editor: Xin Zhong

Received: 10 December 2021
Accepted: 11 February 2022
Published: 14 February 2022

1. Introduction

Private web albums rely on a cloud service for owners to back up personal photos. This is a convenient but insecure platform that enables malicious users to illegally log in and obtain sensitive content for illegal dissemination. Image encryption offers a feasible solution to this potential security risk by allowing users to encrypt sensitive content into meaningless content before uploading it. To confirm the security and validity of image encryption, several techniques, such as compound homogeneous hyper-chaotic [1], Chaotic Map [2], and Rubik's cube method [3], are applied to image encryption. An encrypted image exists in a kind of noise-like mode. Data cannot be obtained without the secret keys, even through statistical calculations. However, meaningless image content is difficult for the owner to manage. Data hiding can cleverly embed the search index into an encrypted image for subsequent image retrieval so that owners can efficiently find specific photos from a large number of encrypted images. In addition, privileged data, such as patient name or identity, can be embedded in encrypted images to preserve a patient's privacy.

For some specific fields, such as medical and military fields, image distortion is unacceptable. Therefore, after extracting the hidden data, the medical image must be completely restored. Such methods are referred to as Reversible Data Hiding (RDH) methods [4–7]. When reversibility is applied to encrypted images, it is called the RDHEI method [8–32].

Zhang's scheme [25] encrypts the most significant five bits (MSB) in the original pixel through XOR operation with the random bit stream. Later, it compresses the least significant bit (LSB) of the encrypted image to create space for embedding the secret. However, the embedding rate of the test image "Lena" is very low, only 0.017 bpp (bits per pixel). The decrypted image is similar to but not the same as the original image. Hong et al. [11] applied side matching information to improve the quality of restoration, which is about 1.21% higher than Zhang's [25]. Qian et al. [16] used a progressive method to achieve concealment and recovery quality of 0.043 bpp and 38.1 dB, respectively. To increase confidentiality, Zhang [33] generated two different keys to encrypt images and hide messages in images; the keys were then distributed to different authorized users.

To achieve reversibility, Ma et al. [14] proposed a reversible data embedding scheme utilizing histogram shifting technology in encrypted images. Qian et al. [17] used error correction codes in Huffman coding to increase the amount of hidden information. Based on the pixel concept, which is similar to surrounding pixels, Cao et al. [8] compressed the image to make room for data embedding. However, the spatial correlation changed after the image was encrypted [12]. To resolve this shortcoming, Huang et al. [12] unitized a key design to maintain the spatial correlation even after image encryption. Zhou et al. [32] predicted pixels using neighboring pixels for secret embedding. Unfortunately, the pixels could be fully restored. In order to improve the security, the scheme [10,30] uses homomorphic encryption to encrypt the image. However, the scheme in [10,30] has low hiding capacity and encounters the problem of pixel expansion. Using the histogram shifting technique, Huang et al. [12] regard the MSB as the secret embedded in the LSB of the encrypted block.

To increase the hiding capacity, Yi et al. [22] utilized parameter binary tree labels to mark the redundant relationship between pixels and blocks. In Yi et al.'s scheme [22], the average hidden payload displayed based on the 2×2 and 3×3 block sizes were 1.752 and 2.003 bpp, respectively. Yin et al. [24] observed that the MSB of the original pixel and the predicted pixel are usually the same, which can be used to make room for secret embedding. In addition, Huffman coding is applied to reduce the size of the indicator, which is used to indicate the number of MSBs that can be leveraged to embed secrets per pixel. Yin et al.'s scheme [24] provides good pixel predictions to improve the hidden payload. However, an image encrypted through MSB prediction is easily decoded through statistical analysis [34]. To address this issue, a hiding and encryption method [35] integrated with Hamming coding is proposed in this paper. The MSB prediction can be shuffled using Hamming coding with a secret key. To achieve good pixel prediction, the proposed scheme designs two thresholds used in [24] to flexibly adjust the predicted value to be close to the original one. To improve the hiding capacity of the technique in [35], the Hamming code encoding is based on two different image types: smooth and complex images.

2. Proposed Scheme

The proposed scheme includes a secret embedding and encryption stage (in Section 2.1) and a secret extraction and decryption stage (in Section 2.2).

2.1. Secret Embedding and Encryption Stage

For clarity, the stage is divided into four steps, namely prediction, difference calculation, Hamming coding calculation and shuffling, and secret embedding and encryption. The details of each step are described as follows.

In the prediction step, since pixels are similar to their neighboring pixels, the pixels located in the first row and first column are regarded as unchanged seed pixels, named reference pixels, which are used to predict the neighboring pixels. Aside from the reference pixels, the predicted pixels represented by $\hat{p}(i, j)$ are calculated via Equation (1) based on

the neighboring pixels of the pixel $p(i,j)$, where (i,j) is the location of the pixel in the given image O, t_1 and t_2 are thresholds that control the predicted value close to $p(i,j)$. For example, as shown in Figure 1, the neighboring pixels of $p(2,2)$ are $p(1,1)$, $p(1,2)$ and $p(2,1)$ and the corresponding pixel values are 100, 102 and 105, respectively. Because $p(1,1)$ is less than $p(1,2)$ and $p(2,1)$, $\hat{p}(2,2)$ will be 106 when t_1 is 1.

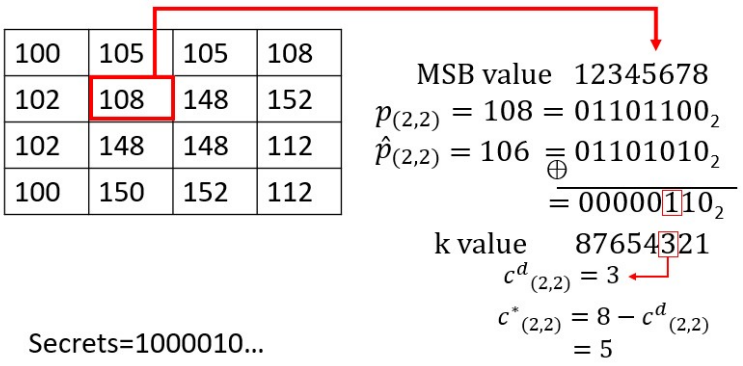


Figure 1. A sample prediction for a complex image.

Next, in the difference calculation step, we use Equation (2) to convert $p(i,j)$ and $\hat{p}(i,j)$ into two bitstreams. The k-th bits corresponding to $p(i,j)$ and $\hat{p}(i,j)$ are respectively expressed as $p^k(i,j)$ and $\hat{p}^k(i,j)$, where "mod" is a modular operation. For example, if $p(2,2)$ is equal to 108, then $p^8(2,2)$ is 0. Then, Equation (3) is used to calculate the difference between the two numbers $p(i,j)$ and $\hat{p}(i,j)$, where $\oplus$ is the exclusive OR operation. As for the example shown in Figure 1, if $p(2,2)$ and $\hat{p}(2,2)$ are 108 and 106, the result of Equation (3) is 00000110, denoted by $x(2,2)$. With Equation (4), the kth most significant bit of $x(i,j)$ equal to 1 is denoted by $c^d(i,j)$. In the same example, if $x(2,2)$ is equal to 00000110, the value $c^d(2,2)$ will be 3. Next, $c^*(i,j)$ is calculated with Equation (5) to represent the number of consecutive MSBs of the original pixel equal to that of its predicted pixel. For example, the value of $c^*(2,2)$ is 5. In other words, in the example, five MSBs of the pixel $p(2,2)$ can be predicted by $\hat{p}(2,2)$. The value of the 6th MSB of $p(i,j)$ can be recovered as the flipped value of $\hat{p}(2,2)$. Thus, the $c^*(i,j)+1$ MSB of the pixel $p(i,j)$ can be used to hide the secret.

$$\hat{p}(i,j) = \begin{cases} p(i-1,j-1) & \text{if } p(i-1,j-1) = p(i-1,j) = p(i,j-1) \\ max(p(i-1,j), p(i,j-1)) + t_1 & \text{if } p(i-1,j-1) < min(p(i-1,j), p(i,j-1)) \\ min(p(i-1,j), p(i,j-1)) + t_2 & \text{if } p(i-1,j-1) > max(p(i-1,j), p(i,j-1)) \\ p(i,j-1) + p(i-1,j) - p(i-1,j-1) & \text{otherwise.} \end{cases} \quad (1)$$

$$p^k(i,j) = \frac{p(i,j) \bmod 2^k}{2^{k-1}}, k \text{ for 8 to 1} \quad (2)$$

$$x^k(i,j) = p^k(i,j) \oplus \hat{p}^k(i,j), k = 1, 2, ..., 8 \quad (3)$$

$$c^d(i,j) = \arg\max_k x^k(i,j) = 1, \ k = 1, 2, ..., 8 \quad (4)$$

$$c^*(i,j) = 8 - c^d(i,j) \quad (5)$$

In the Hamming coding calculation and shuffling step, when applying Hamming coding matrix [36], we convert the $c^*(i,j)$ into a bit stream as an indicator, depicted by $(w_1 w_2 w_3)$. Noted that since $(w_1 w_2 w_3)$ can only appear in eight cases, but there are actually nine cases in the difference calculation, the cases are defined according to smooth and complex images. The image judges the type of image based on the higher capacity in the smooth type encoding or complex one. The image type (smooth and complex are marked as "s" and "c", respectively) is treated as a parameter saved in the database. If it is a smooth image, values of $c^*(i,j)$ equal to 0 and 1 fall into the same case. Therefore, with regard to

the smooth image, the decimal value of the indicator $(w_1w_2w_3)$ is the value of $c^*(i,j)-1$ if $c^*(i,j)$ is larger than 0 (the fallen cases are recorded under "Label" in Table 1). If it is a complex image, the values of $c^*(i,j)$ equal to 7 and 8 fall into the same case; in such a situation, if $c^*(i,j)$ is equal to 8, then the decimal value of $(w_1w_2w_3)$ is adjusted to 7 (also recorded under "Label" in Table 1). For example, in Figure 1, the value $c^*(i,j)$ generates the indicator $(w_1w_2w_3)$ as 101 if the image is a complex image.

Essentially, the value $w_1w_2w_3$ can represent which case the pixel has fallen into, which indicates the hiding capacity. Next, the value of $w_1w_2w_3$ is assigned to Equation (7), and the values of w_1^*, w_2^* and w_3^* can be found to make the condition mC satisfy the value of $w_1w_2w_3$ (as shown in Equation (7)). The Hamming coding matrix is used as the first round to shuffle the original values of $w_1w_2w_3$ to produce the values of w_1^*, w_2^* and w_3^*. For example, if $c^*(2,2)$ is equal to 5, $w_1w_2w_3$ will be 101. While satisfying Equation (7)), the values of w_1^*, w_2^* and w_3^* will be 1, 1, and 0, respectively.

Subsequently, the value $w_1^*w_2^*w_3^*$ is recalculated as $w_1'w_2'w_3'$ for the second round of shuffling with Equation (8), where $r_1r_2r_3$ is generated by a random number generator with the key s_1. In the previous example, if the value of $r_1r_2r_3$ is 001, $w_1'w_2'w_3'$ will be $110 \oplus 001 = 111$.

In the secret embedding and encryption step, the bit stream of the embedded pixel consists of three parts as shown in Figure 2, in which the embedded pixel is represented by $p'(i,j)$. The first part is for the Hamming encoding code, which expresses which case the given pixel $p(i,j)$ falls into. The first three MSBs of the embedded pixel equate to the second-round shuffling result $w_1'w_2'w_3'$.

As for the second part in Figure 2, the values of the "Payload" column in Table 1, depicted by $payload(i,j)$, indicate the number of secret bits to-be embedded to the pixel $p(i,j)$. In this part, the hiding capacity for different types of images is listed in Table 1. Here, the fallen cases, the capacity, the code length, and pure payload are denoted by "Label", "Capacity", "Code length" and "Payload", respectively. The column "Label" indicates the decimal value of the indicator $(w_1w_2w_3)$, "Capacity" refers to the amount of changeable bits for a given pixel, "Code length" is the code length of Hamming codes, and "Payload" represents the pure hiding capacity (i.e., the result of "Capacity" value minus the "Code length" value). For example, for a complex image, the payload is 3 if Label is 5 (i.e., $6-3=3$). If $payload(i,j)$ is less than 0, no secret data are hidden. In addition, the original $|payload(i,j)|$ MSBs of the pixel $p(i,j)$ are regarded as part of the to-be hidden secret data, where $|b|$ is the absolute value of b. When $payload(i,j)$ is larger than 0, the second part of the $p'(i,j)$, bits of the secret data, are equal to the values from the 4th MSB bit to the $payload(i,j)+3$-th MSB bits of the $p'(i,j)$. The third part is the last $(5-payload(i,j))$ bits of $p(i,j)$ if $payload(i,j)$ is larger than 0; otherwise, the last five bits of the embedded pixel are kept the same as those of $p(i,j)$.

In the same example, $p'(i,j)$ can be constructed as 11100100, and the first 3 MSBs $w_1'w_2'w_3'$ as 111; the second part for embedding $payload(i,j)$ bits of secret data (e.g., obtain three bits of the to-be hidden data (001 in this example), owing to the $w_1w_2w_3$ equal to 101 in Table 1); and the third part as the last (i.e., $5-3=2$) bits of the original pixel (i.e., the last two bits of original pixel $p(2,2)$ are 00). If $payload(i,j)$ is less than 0, no data are hidden in this situation, and the 4th MSB onward to the last bit of $p(i,j)$ is equal to that of $p'(i,j)$. For clarity, the k position of the pixel $p'(i,j)$ is represented by $p'^k(i,j)$. In the image encryption stage, the key s_2 generates a random number $r(i,j)$, where the k bit of $r(i,j)$ is represented as $r^k(i,j)$. Finally, $p'(i,j)$ can be encrypted via Equation (9) with key s_2. For example, if $r^k(2,2)$ is 01011001, the encrypted pixel $p^*(2,2)$ will be $(10111101)_2 = 189$.

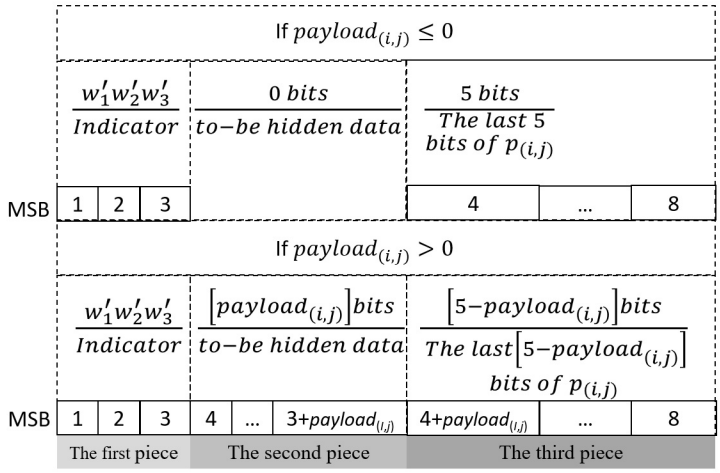


Figure 2. Three pieces of the embedded pixel.

Table 1. The pure payloads of the smooth image and complex image.

Label	Smooth Image			Complex Image		
	Capacity	Code Length	Payload	Capacity	Code Length	Payload
0	0	3	−3	1	3	−2
1	3	3	0	2	3	−1
2	4	3	1	3	3	0
3	5	3	2	4	3	1
4	6	3	3	5	3	2
5	7	3	4	6	3	3
6	8	3	5	7	3	4
7	8	3	5	7	3	4

$$m = \begin{bmatrix} (w_1^* & w_2^* & \hat{p}^5(i,j) & w_3^* & \hat{p}^6(i,j) & \hat{p}^7(i,j) & \hat{p}^8(i,j) \end{bmatrix} \quad (6)$$

$$\text{Let } w_1w_2w_3 = mC \bmod 2, \text{ where } C = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (7)$$

$$w'_1w'_2w'_3 = w_1^*w_2^*w_3^* \oplus r_1r_2r_3 \quad (8)$$

$$p^{*k}(i,j) = p'^k(i,j) \oplus r^k(i,j), \text{for } k = 1 \text{ to } 8 \quad (9)$$

2.2. Secret Extraction and Decryption Phases

This phase consists of three steps: the decryption step, data extraction step, and recovery step.

In the decryption step, the keys s_1 and s_2 will generate random bits $r_1r_2r_3$ and $r(i,j)$. Using Equation (9), the encrypted pixel $p^{*k}(i,j)$ can be decrypted as $p'^k(i,j)$, where $r^k(i,j)$ is the random bit used to decrypt the k bits of pixel $p^k(i,j)$. Thus, the $p'^k(i,j)$ can be reconstructed as $10111101 \oplus 01011001 = 11100100$.

In the data extraction step, the value of $w_1^*w_2^*w_3^*$ can be obtained through Equation (8) key s_1. In the same example, the values of $w'_1w'_2w'_3$ and $r_1r_2r_3$ are 111 and 001, which output $w_1^*w_2^*w_3^*$ as $111 \oplus 001 = 110$. After that, the first 3 MSBs of the decrypted image are integrated as $w_1^*w_2^*w_3^*$ with predicted pixels $\hat{p}^5(i,j)$, $\hat{p}^6(i,j)$, $\hat{p}^7(i,j)$ and $\hat{p}^8(i,j)$ using Equation (7) to calculate mC. mC determines $w_1w_2w_3$. The decimal of $w_1w_2w_3$ represents

the indicator in column "Label" of Table 1. Mapping to a fallen case, if the corresponding value of $payload(i,j)$ is larger than 0, we can obtain the hidden data from the 4th MSB to the $3 + payload(i,j)$th MSB of $p'(i,j)$ as shown in the second piece of Figure 2. In contrast, if $payload(i,j)$ is less than 1, no secret data are extracted.

In the previous example, the predicted pixel is $\hat{p}(2,2) = 106$; thus, the values of $\hat{p}^5(i,j)$, $\hat{p}^6(i,j)$, $hatp^7(i,j)$ and $\hat{p}^8(i,j)$ are 0, 1, 1, 0, respectively. The value of m is constructed as [1100110] by Equation (6). With Equation (7), the value of $w_1w_2w_3$ is 101. The decimal value of $w_1w_2w_3$ is equal to 5, which indicates $payload(i,j)$ equal to 3. In the example, the hidden data are extracted as 001 (from the 4th MSB to the 6th MSB of $p'(2,2)$).

The algorithm for the recovery step is shown in Algorithm 1. The value of "Capacity" in Table 1 (for clarity, the value depicted by $payload(i,j)$, represents the number of bits that can be recovered from the predicted pixel). Furthermore, two recovery situations are for smooth images and complex images. For the smooth image, if $payload(i,j)$ equals 0, no bits of the predicted pixel $\hat{p}(i,j)$ can be used to recover the original pixel $p(i,j)$, and the original 3 MSBs of $p(i,j)$ are recovered through the hidden data extracted by other pixels. As for a complex image, if $payload(i,j)$ is less than 1, the $|payload(i,j)|$ MSBs are also recovered by the hidden data extracted by other pixels. The 4th MSB onward to the last bit of $p(i,j)$ is recovered by that of $p'(i,j)$.

If $payload(i,j)$ is larger than 0, the first $payload(i,j) - 1$ MSBs of the predicated pixel are the same as the first $payload(i,j) - 1$ MSBs of the original pixel. In addition, the $payload(i,j)$th MSB of $p(i,j)$ is recovered by the flipped value of $\hat{p}(i,j)$. The last $5 - payload(i,j)$ bits of $p'(i,j)$ can be used to recover those of $p(i,j)$. That is, the $payload(i,j)$th MSB of $p(i,j)$ is equal to $1 - \hat{p}^{8-payload(i,j)}(i,j)$.

In the example, $payload(i,j)$ is 6. Thus, the original first five MSBs are recovered as 01101 and the 6th MSB bit can be restored as 1 (flip the 6th MSB of $\hat{p}(2,2)$). The remaining bits of $p'(2,2)$ can be used to recover that of $p(2,2)$ as 00. Thus, the reconstructed pixel is 01101100 (i.e., 108).

Algorithm 1 Recovering pseudo code

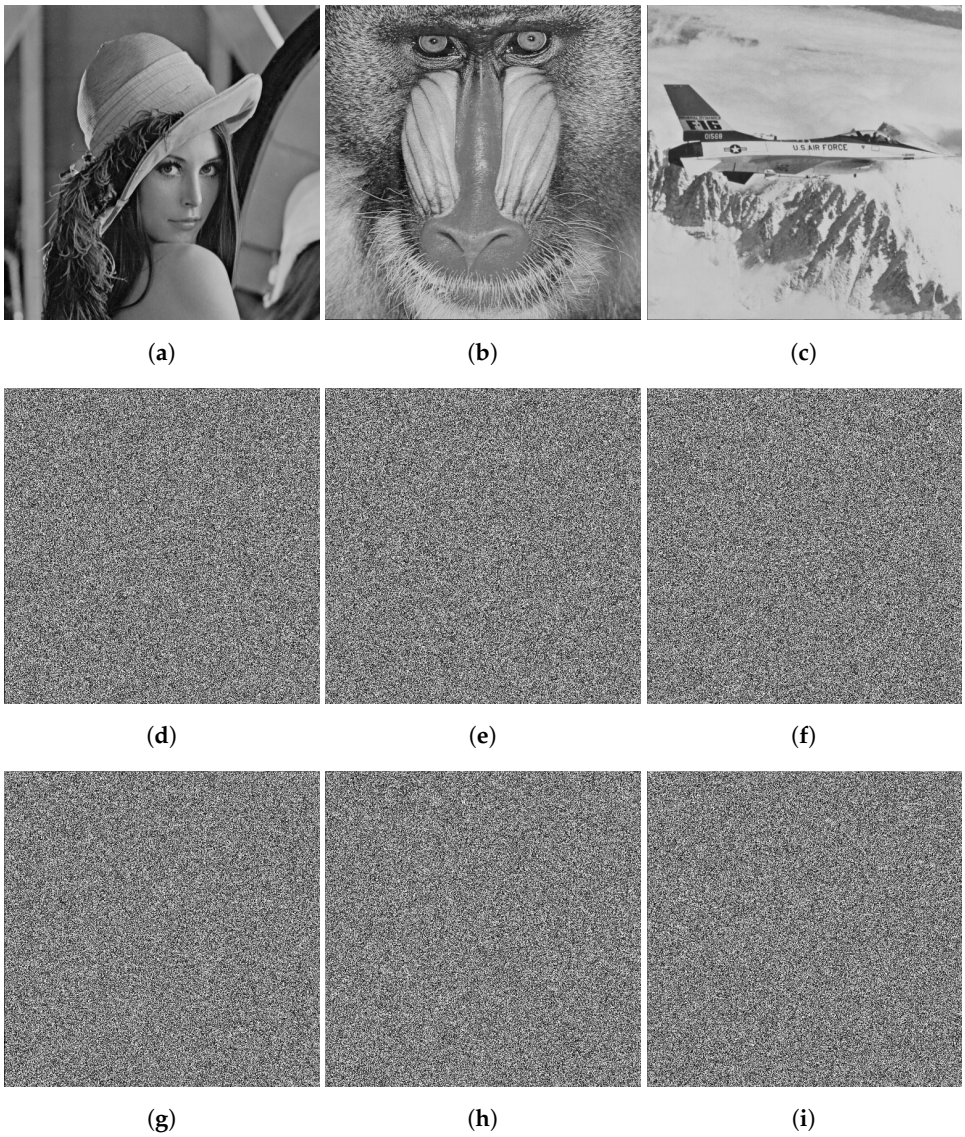
```
procedure RECOVERY(payload(i,j), p̂(i,j))
    if payload(i,j) > 0 then
        the payload(i,j) − 1 MSBs of p(i,j) are recovered by that of p̂(i,j).
        the payload(i,j)th MSB of p(i,j) = 1 − p̂^{8−payload(i,j)}(i,j).
        the last 5 − payload(i,j) bits of p(i,j) are recovered by that of p'(i,j).
    else if payload(i,j) ≤ 0 then
        if Image is tagged as s then
            the original 3 MSBs of p(i,j) can be recovered by the other pixels.
        else if Image is tagged as c then
            |payload(i,j)| MSBs are recovered by the other pixels.
        the last 5 bits of p(i,j) are recovered by that of p'(i,j).
```

3. Experimental Results

This section shows the experimental results and compares them with the results of [9,15,22,35,37,38]. Three test images: Lena, Baboon and Jetplane, as shown in Figure 3. In addition, the measured metric PSNR (Peak Signal-to-Noise Ratio), as seen in Equation (10), is used to evaluate the visual quality. After the image is encrypted, the encrypted results of Figure 3a–c are shown in Figure 3d–f, respectively. The encryption result is meaningless to users. The PSNRs of Figure 3d–f are 8.816, 9.524 and 8.677 dB, respectively. Figure 3g–i shows that the attacker used the wrong key to decrypt the image, and the PSNRs are 8.797, 9.507 and 8.656 dB, respectively.

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

$$MSE = \frac{\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (p^*(i,j) - p(i,j))}{m \times n} \tag{11}$$



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

(**g**) (**h**) (**i**)

Figure 3. Test images and the corresponding encrypted and decrypted with wrong key images. (**a**) Lena. (**b**) Baboon. (**c**) Jetplane. (**d**) Encrypted result of (**a**). (**e**) Encrypted result of (**b**). (**f**) Encrypted result of (**c**). (**g**) Illegal decryption of (**d**). (**h**) Illegal decryption of (**e**). (**i**) Illegal decryption of (**f**).

Hidden capacity is expressed in bits per pixel (bpp). Table 2 shows the hiding capacity of the test images Lena, Baboon and Jetplane. Here, the field "Label" is mapped to the falling case according to the value of $c^*(i,j)$ and the image type (smooth or complex image), and the "distribution" is the number of pixels classified into the same "Label". "Payload" refers to pure payload. The payloads of the images Lena, Baboon and Jetplane are 2.58, 1.19 and 3.43 bpp, respectively, which are higher than those in [35]. In addition, the experiments showed that the prediction results for smooth images (such as Lena and Jetplane) were better than those for complex images (such as Baboon), which resulted in smooth images having greater hiding capacity than complex images did. Moreover, thresholds t_1 and t_2 settings are the flexible factors for different images to make the prediction pixel close to the original ones. The close prediction causes a better hiding capacity. Table 3 compares the payload (bpp) of the proposed scheme with the methods in [9,15,22,35,37,38]. The results show that the proposed method outperforms the six other algorithms.

Table 2. The pure payloads of images "Lena", "Baboon" and "Jetplane".

Label	Lena ($t_1 = -1$, $t_2 = 1$)		($t_1 = 0$, $t_2 = 0$)		Jetplane ($t_1 = 0$, $t_2 = -1$)	
	Distribution	Payload	Distribution	Payload	Distribution	Payload
−1	1023	–	1023	–	1023	–
0	8200 7493	−24,600 −22,479	34,591	−69,182	4457 9990	−13,371 −29,970
1	18,984	0	22,574	−22,574	8427	0
2	28,742	28,742	40,173	0	16,430	16,430
3	47,653	95,306	44,464	44,464	26,250	52,500
4	53,002	159,006	41,394	82,788	33,346	100,038
5	42,304	169,216	32,230	96,690	36,543	146,172
6	25,060	125,300	20,949	83,796	36,952	184,760
7	29,683	148,415	12,033 12,713	48,132 50,852	88,726	443,630
Total	-	677,883	-	313,943	-	899,166

Table 3. Comparison of payload on three images.

Method	Payload		
	Lena	Baboon	Jetplane
Scheme [35]	2.42	1.19	2.98
Scheme [15]	0.977	0.838	0.983
Scheme [37]	1.636	0.833	1.717
Scheme [38]	1.156	0.372	1.294
Scheme [22]	2.014	0.462	2.008
Scheme [9]	1.928	0.480	2.254
Proposed Scheme	2.58	1.19	3.43

4. Conclusions

Image encryption aims to encrypt sensitive data to protect privacy for the image owner. This paper proposes a reversible data embedding scheme using Hamming coding in encrypted images. On average, the code length is 3 bpp to expand the hiding capacity. In the experiment, the hiding capacity of the proposed scheme is greater than that in [9,15,22,35,37,38] for both smooth and complex types of images. In addition, the proposed technique enables an encrypted image to be completely restored to its original state after the hidden data are extracted.

Author Contributions: Formal analysis, P.-Y.L.; Methodology, Y.-H.C. and S.-H.C.; Visualization, S.-H.C.; Writing–original draft, Y.-H.C., P.-Y.L. and H.-P.W.; Writing–review and editing, H.-P.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the Ministry of Science and Technology of the Republic of China, Taiwan, under Grant MOST 107-2221-E-182 -081 -MY3 and MOST 110-2221-E-182-026-MY3, and in part by the Kaohsiung Chang Gung Memorial Hospital with grant number CMRPD3M0011.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

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

References

1. Zhu, H.; Zhang, X.; Yu, H.; Zhao, C.; Zhu, Z. An image encryption algorithm based on compound homogeneous hyper-chaotic system. *Nonlinear Dyn.* **2017**, *89*, 61–79. [CrossRef]
2. Zhu, H.; Zhao, Y.; Song, Y. 2D logistic-modulated-sine-coupling-logistic chaotic map for image encryption. *IEEE Access* **2019**, *7*, 14081–14098. [CrossRef]

3. Zhu, H.; Dai, L.; Liu, Y.; Wu, L. A three-dimensional bit-level image encryption algorithm with Rubik's cube method. *Math. Comput. Simul.* **2021**, *185*, 754–770. [CrossRef]
4. Coatrieux, G.; Le Guillou, C.; Cauvin, J.M.; Roux, C. Reversible watermarking for knowledge digest embedding and reliability control in medical images. *IEEE Trans. Inf. Technol. Biomed.* **2008**, *13*, 158–165. [CrossRef]
5. Lee, S.; Yoo, C.D.; Kalker, T. Reversible image watermarking based on integer-to-integer wavelet transform. *IEEE Trans. Inf. Forensics Secur.* **2007**, *2*, 321–330. [CrossRef]
6. Celik, M.U.; Sharma, G.; Tekalp, A.M.; Saber, E. Lossless generalized-LSB data embedding. *IEEE Trans. Image Process.* **2005**, *14*, 253–266. [CrossRef]
7. Celik, M.U.; Sharma, G.; Tekalp, A.M. Lossless watermarking for image authentication: A new framework and an implementation. *IEEE Trans. Image Process.* **2006**, *15*, 1042–1049. [CrossRef]
8. Cao, X.; Du, L.; Wei, X.; Meng, D.; Guo, X. High capacity reversible data hiding in encrypted images by patch-level sparse representation. *IEEE Trans. Cybern.* **2015**, *46*, 1132–1143. [CrossRef]
9. Chen, K.; Chang, C.C. High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement. *J. Vis. Commun. Image Represent.* **2019**, *58*, 334–344. [CrossRef]
10. Chen, Y.C.; Shiu, C.W.; Horng, G. Encrypted signal-based reversible data hiding with public key cryptosystem. *J. Vis. Commun. Image Represent.* **2014**, *25*, 1164–1170. [CrossRef]
11. 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]
12. Huang, F.; Huang, J.; Shi, Y.Q. New framework for reversible data hiding in encrypted domain. *IEEE Trans. Inf. Forensics Secur.* **2016**, *11*, 2777–2789. [CrossRef]
13. 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]
14. 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]
15. Puteaux, P.; Puech, W. An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 1670–1681. [CrossRef]
16. Qian, Z.; Zhang, X.; Feng, G. Reversible data hiding in encrypted images based on progressive recovery. *IEEE Signal Process. Lett.* **2016**, *23*, 1672–1676. [CrossRef]
17. Qian, Z.; Zhang, X.; Wang, S. Reversible data hiding in encrypted JPEG bitstream. *IEEE Trans. Multimed.* **2014**, *16*, 1486–1491. [CrossRef]
18. Qian, Z.; Zhang, X. Reversible data hiding in encrypted images with distributed source encoding. *IEEE Trans. Circuits Syst. Video Technol.* **2015**, *26*, 636–646. [CrossRef]
19. Wu, X.; Sun, W. High-capacity reversible data hiding in encrypted images by prediction error. *Signal Process.* **2014**, *104*, 387–400. [CrossRef]
20. Xu, D.; Wang, R. Separable and error-free reversible data hiding in encrypted images. *Signal Process.* **2016**, *123*, 9–21. [CrossRef]
21. Yi, S.; Zhou, Y. Binary-block embedding for reversible data hiding in encrypted images. *Signal Process.* **2017**, *133*, 40–51. [CrossRef]
22. 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]
23. Yin, Z.; Abel, A.; Tang, J.; Zhang, X.; Luo, B. Reversible data hiding in encrypted images based on multi-level encryption and block histogram modification. *Multimed. Tools Appl.* **2017**, *76*, 3899–3920. [CrossRef]
24. 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. [CrossRef]
25. Zhang, X. Reversible data hiding in encrypted image. *IEEE Signal Process. Lett.* **2011**, *18*, 255–258. [CrossRef]
26. Zhang, X. Separable Reversible Data Hiding in Encrypted Image. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 826–832. [CrossRef]
27. Zhang, W.; Ma, K.; Yu, N. Reversibility improved data hiding in encrypted images. *Signal Process.* **2014**, *94*, 118–127. [CrossRef]
28. Zhang, X.; Qian, Z.; Feng, G.; Ren, Y. Efficient reversible data hiding in encrypted images. *J. Vis. Commun. Image Represent.* **2014**, *25*, 322–328. [CrossRef]
29. Zhang, X.; Long, J.; Wang, Z.; Cheng, H. Lossless and reversible data hiding in encrypted images with public-key cryptography. *IEEE Trans. Circuits Syst. Video Technol.* **2015**, *26*, 1622–1631. [CrossRef]
30. Zhang, W.; Wang, H.; Hou, D.; Yu, N. Reversible data hiding in encrypted images by reversible image transformation. *IEEE Trans. Multimed.* **2016**, *18*, 1469–1479. [CrossRef]
31. Zheng, S.; Li, D.; Hu, D.; Ye, D.; Wang, L.; Wang, J. Lossless data hiding algorithm for encrypted images with high capacity. *Multimed. Tools Appl.* **2016**, *75*, 13765–13778. [CrossRef]
32. Zhou, J.; Sun, W.; Dong, L.; Liu, X.; Au, O.C.; Tang, Y.Y. Secure reversible image data hiding over encrypted domain via key modulation. *IEEE Trans. Circuits Syst. Video Technol.* **2016**, *26*, 441–452. [CrossRef]
33. Zhang, X.; Feng, G.; Ren, Y.; Qian, Z. Scalable coding of encrypted images. *IEEE Trans. Image Process.* **2012**, *21*, 3108–3114. [CrossRef]
34. Dragoi, I.C.; Coltuc, D. On the security of reversible data hiding in encrypted images by MSB prediction. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 187–189. [CrossRef]

35. Chen, Y.H.; Lin, P.Y. Integrated Hamming Coding Operation to Reversible Data Hiding Scheme for Encrypted Images. In Proceedings of 22nd IEEE/ACIS International Fall Virtual Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD2021-Fall), Taichung, Taiwan, 24–26 November 2021.
36. Greenwald, S.W. Matrix Multiplication with Asynchronous Logic Automata. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2010.
37. Puteaux, P.; Puech, W. EPE-based huge-capacity reversible data hiding in encrypted images. In Proceedings of the 2018 IEEE International Workshop on Information Forensics and Security (WIFS), Hong Kong, China, 11–13 December 2018; pp. 1–7.
38. Puyang, Y.; Yin, Z.; Qian, Z. Reversible data hiding in encrypted images with two-MSB prediction. In Proceedings of the 2018 IEEE International Workshop on Information Forensics and Security (WIFS), Hong Kong, China, 11–13 December 2018; pp. 1–7.