

Article

# Reversible Data Hiding in Encrypted Image Using Multiple Data-Hiders Sharing Algorithm

Chi-Yao Weng \*  and Cheng-Hsing Yang

Department of Computer Science and Artificial Intelligence, National Pingtung University, Pingtung 900, Taiwan  
\* Correspondence: cyweng@mail.nptu.edu.tw

**Abstract:** Reversible Data Hiding in Encrypted Image (RDHEI) is a technology for embedding secret information in an encrypted image. It allows the extraction of secret information and lossless decryption and the reconstruction of the original image. This paper proposes an RDHEI technique based on Shamir's Secret Sharing technique and multi-project construction technique. Our approach is to let the image owner hide the pixel values in the coefficients of the polynomial by grouping the pixels and constructing a polynomial. Then, we substitute the secret key into the polynomial through Shamir's Secret Sharing technology. It enables the Galois Field calculation to generate the shared pixels. Finally, we divide the shared pixels into 8 bits and allocate them to the pixels of the shared image. Thus, the embedded space is vacated, and the generated shared image is hidden in the secret message. The experimental results demonstrate that our approach has a multi-hider mechanism and each shared image has a fixed embedding rate, which does not decrease as more images are shared. Additionally, the embedding rate is improved compared with the previous approach.

**Keywords:** encrypted image; reversible data hiding; secret sharing; multi-hider mechanism



**Citation:** Weng, C.-Y.; Yang, C.-H. Reversible Data Hiding in Encrypted Image Using Multiple Data-Hiders Sharing Algorithm. *Entropy* **2023**, *25*, 209. <https://doi.org/10.3390/e25020209>

Academic Editors: Tzu Chuen Lu and David Megías

Received: 6 December 2022

Revised: 17 January 2023

Accepted: 18 January 2023

Published: 21 January 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Multimedia security technology is used to prevent unauthorized users from copying, sharing, and modifying media content. To prevent this problem, encryption and information hiding are often used to protect media content. As far as information hiding technology is concerned, traditional information hiding technology will destroy the content of the cover image. However, in some exceptional cases, such as military, medical, and legal document images, the slight distortion of the image is entirely unacceptable. Therefore, whether these images can be completely restored is very important. Reversible data hiding scheme (RDH) can correspond with the requirement of being lossless. RDH methods applied the methodology of changing context to hide the secret data in cover media. After data extracting, the changing context will be fully recovered to the cover media. On the other hand, RDHEI (Reversible Data Hiding in Encrypted Images) technology can combine encryption technology with RDH technology, which can not only hide secret information in the image, but can also encrypt the image to protect the image content.

RDH techniques can be broadly classified into three types: (1) Difference Expansion [1–3]: Difference expansion is performed by expanding the difference between adjacent pixels, and then secret information is embedded into the difference. Since the difference will be expanded after the secret information is embedded, this technique will inevitably produce a larger distortion; (2) Histogram Shift [4–6]: through Histogram Shift, the histogram is shifted by the original image or the histogram of the predicted discrepancy, and the empty position after the shift is used to embed the secret information; (3) Lossless Compression [7,8]: The secret information is hidden in the extra space after compressing the original image. Since lossless compression may lead to significant degradation of visual quality, it has received less attention. By and large, the RDH approach is based on the above with some other added strategies. For example, in 2019, Zhao et al. proposed a

method of histogram displacement [9]. First, the secret information is converted into a message with only  $-1$ ,  $0$ , and  $1$  by encoding. Then, the middle segment bin is selected for embedding on the histogram displacement. These bins can all be used to embed a  $-1$ ,  $0$ , or  $1$  message by shifting them left and right. Finally, the band size is adjusted using the Threshold, and the histogram is created using the prediction error values generated by the Chess Board Prediction method. In 2021, Gao et al. proposed a histogram displacement method for embedding medical images [10]. By dividing medical images into ROI and NROI, the embedding process will stretch the histogram created by the pixel values of ROI to the left and right, which expands the embedding capacity of ROI and enhance the contrast of the image.

RDHEI techniques can be divided into two categories: (1) Vacating Room before Encryption, VRBE [11–17]: the original image is first vacated and then encrypted; and (2) Vacating Room after Encryption, VRAE [18–21]: The image is first encrypted, and because the encrypted image retains certain properties, the encrypted image can be hidden. VRBE technology mainly focuses on reversible information hiding or compression of the original image to free up space. It encrypts the image, so there is space to hide information in the encrypted image. For example, in 2013, Ma et al.'s RDHEI method belongs to the technology of Vacating Room before Encryption [11]. They performed reversible information hiding in the original image beforehand and embedded the LSB part of a block into the image. As a result, the encrypted image can use the LSB part of this block to hide data. In 2018, Puteaux et al. performed MSB prediction and compression on the original image so that the MSB part could vacate most of the space, and the encrypted image could use this MSB part to embed the data [13]. In 2021, Yin et al. proposed a method based on pixel prediction and multi-MSB plane, which belongs to Vacating Room before Encryption [17]. First, it used the MED predictor to obtain the predicted value, and to calculate the predicted error (PE) between the predicted value and original value. Next, the sign bit of PE is stored in bit-plane 8, and the absolute value of PE is represented in bit-plane 7 (bit-plane 7) and bit-plane 1 (bit-plane 1). The bit plane is then divided into Uniform Blocks and Non-Uniform Blocks, and these blocks are rearranged. Since prediction errors are usually concentrated around zero, these bit planes, which have a large number of uniform blocks, can be encoded and compressed to free up space. On the other hand, VRAE technology mainly uses a specific encryption technique to maintain the dependency of the neighboring pixels in the encrypted image. Therefore, this feature can be used to hide the information in the encrypted image. Commonly used encryption techniques include Block-Level Scrambling [18] or Block-Level Encryption [19], in which pixels in the same block use the same random value for XOR (Exclusively-OR). For example, in 2019, Qin et al. proposed a VRAE method [18] that first encrypts the image by stirring the image between bit-planes, between blocks, and within blocks so that the pixels within the blocks retain their dependency. Then, block classification and encoding compression are used to generate the embedding space when embedding information. In 2019, Wang et al. proposed a VRAE approach, where their encryption uses inter- and intra-block stirring so that pixels within a block retain dependency [19]. Then, for the information embedding, they use the 3-LSB of all pixels in the flip block to divide the block into EB (Embeddable Block), and NEB (Non-Embeddable Block), and only the secret information is embedded in the EB. The location map of the block is embedded using the RDH method to hide 5-MSB of all pixels.

Recently, some scholars have proposed the RDHEI approach based on Secret Sharing [20–22]. In 2018, Wu et al. proposed an RDHEI method based on Secret Sharing [20]. However, their encrypted image size was more than twice the size of the original image. In 2019, Chen and other scholars proposed a secret sharing-based RDHEI method that converts original images into encrypted images via the sharing technique of Shamir's Secret Sharing and distributes the encrypted images to information hidings for information hiding [21]. Since the size of the generated encrypted image is the same as that of the original image, no information inflation occurs. All of the above methods are targeted at a single information supporter. In 2020, Chen and other scholars proposed a new secret-sharing

RDHEI approach [22], which expands the original single information hider into multiple information hidiers. The total embedding volume of Chen et al.’s method is fixed, so the embedding rate decreases as the number of shared images increases.

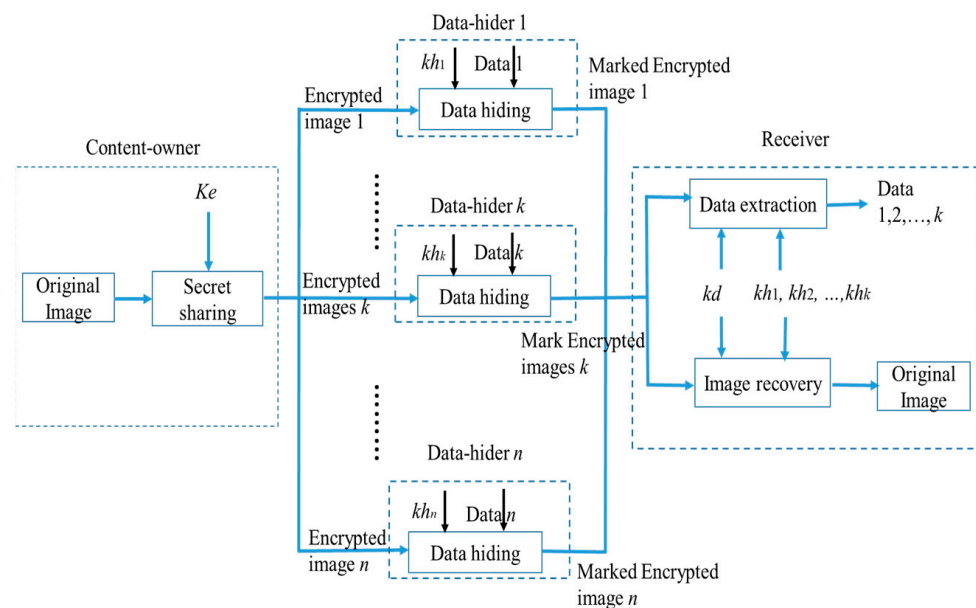
In this study, we propose a new RDHEI technique where we hide the pixel values into the polynomial coefficients by dividing multiple pixels into a group. First, the image owner confidentially shares the image, generating a shared image and leaving space in each shared image for information embedding. Then, the information recipient can embed a confidential message after receiving the shared image. Finally, the recipient can decrypt the image and remove the hidden information. The recipient needs to obtain at least a Threshold before the original image can be recovered. Our approach enables multiple information hidiers and has a more suitable fixed embedding rate. Furthermore, it solves the problem that the embedding rate becomes smaller as the number of shared images becomes larger and successfully improves the embedding rate compared with the previous method.

The remaining sections of this study are organized as follows. Section 2 reviews Chen’s existing work [22]. Section 3 describes our novel secret sharing images with multiple data hidiers method. This is followed by its experimental results along with the analysis and the existing works comparisons in Section 4. Finally, the conclusion is drawn in Section 5.

### 2. Preliminary

This section introduces Chen et al.’s approach, which develops RDHEI techniques through Shamir’s Secret Sharing method. Chen et al.’s method has the role of multiple information hidiers, but their total embedding is fixed, so the embedding rate decreases as the number of shared images increases. Chen et al.’s approach is introduced in the following.

Figure 1 shows the flowchart of encryption, embedding, extraction, and decryption by Chen et al. First, the content-owner encrypts the image and uses the key  $ke$  to encrypt the original image into  $n$ . Each encrypted image is the same size as the original image. Then, each encrypted image is assigned to a different information hider. Each information hider, Data-hider  $t$ , hides the confidential information in the encrypted image  $t$  through the key  $kh_t$ . Finally, the  $1 \leq t \leq n$  receiver simply collects any  $k$  encrypted images with embedded information and related keys to extract the embedded information and decrypt it to obtain the original image.



**Figure 1.** The flowchart of Chen et al.’s method in terms of image encryption, data embedding, data extracting, and image decryption.

The content-owner performs the image encryption by first dividing image  $I$  into parts A and B. Part A is the first 10 pixels of the image, and the remaining part is part B. Figure 2

is the representation. Part A is used to store the parameters of the information hiding and the parameters of the histogram displacement process. First, the following bits are extracted from part A: all bits of the first two pixels and 3 LSBs of the last 8 pixels. Next, all the extracted bits are defined as AP. Then, scan all pixels in the part B, recording the location of the pixel values  $\geq 250$  with the location map, modify these pixel values to 249, and create a histogram of part B. From the histogram, we found the most suitable embedding point for embedding the location map and AP, which was defined as PP. In the histogram, all the values between 0 PP + 1 and 249 are shifted one place to the right so that the location map and AP can be using PP for 0 and for 1, as shown in Figure 3. Finally, the PP value is embedded into the 3-LSB part of the last 8 pixels of the A part. After the above pre-processing, image  $I$  is modified to image  $I'$ .

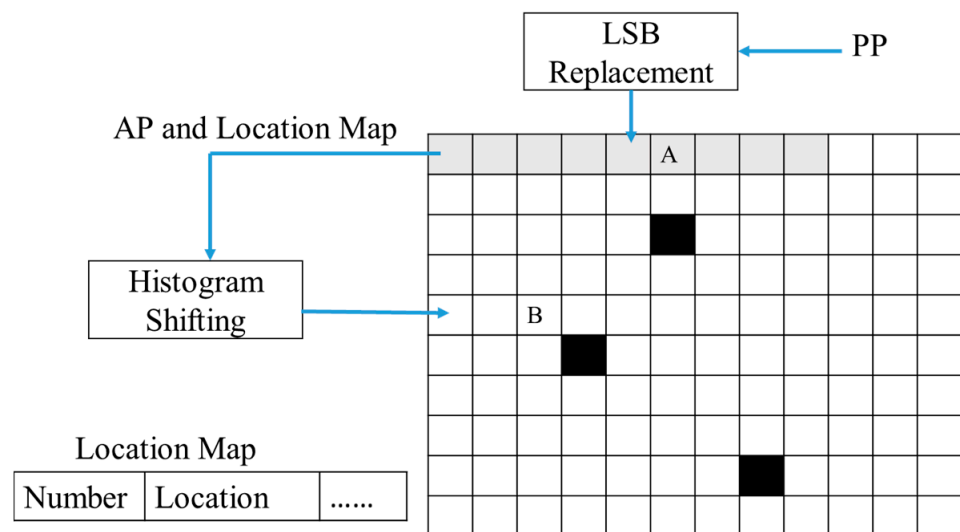


Figure 2. The image is segmented into part A and part B, where black pixels indicate their pixel values  $\geq 250$ .

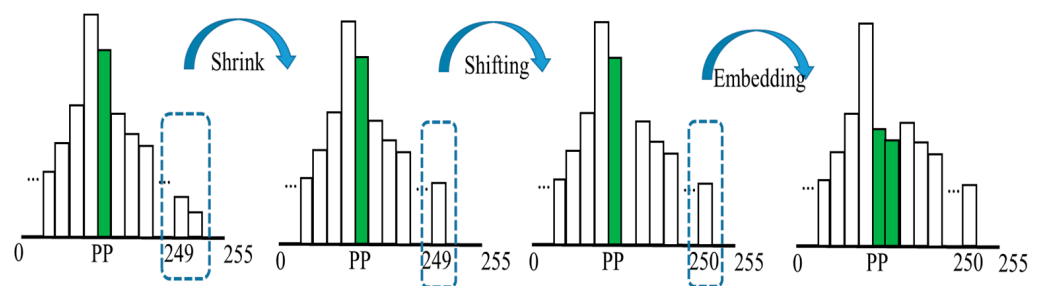


Figure 3. The diagram of pixel value reduction and parameter embedding.

Using Equation (1), the B part of the image  $I'$  is shared confidentially via Shamir's Secret Sharing, where  $T_{i,j}^{(0)}$  is a constant,  $a_{i,j}^{(a)}$  is an integer random number, and  $I'_{i,j}$  is the pixel value of position  $(i,j)$ . The image owner uses key  $ke$  to generate  $n$  non-zero random integers  $x_{i,j}^{(t)}$ ,  $t = 1, 2, \dots, n$ , to bring  $x_{i,j}^{(t)}$  into Equation (1) within  $x$  to obtain the confidential sharing result  $F_{i,j}(x_{i,j}^{(t)})$ ,  $t = 1, 2, \dots, n$ . It then hides the parameters  $t$  and  $n$  into the first two pixels of part A. From this we can obtain  $n$  encrypted images  $E^{(t)}$ ,  $t = 1, 2, \dots, n$ .

$$F_{i,j}(x) = \begin{cases} (T_{i,j}^{(0)} + I'_{i,j}x) \bmod p, & \text{if } k = 2, \\ (T_{i,j}^{(0)} + I'_{i,j}x + \sum_{a=2}^{k-1} a_{i,j}^{(a)} x^a) \bmod p, & \text{if } 2 < k \leq n \end{cases} \quad (1)$$

Data embedding is performed using the  $n$  information hidiers separately. The  $t$ -th information hider obtains the shared encrypted image  $E^{(t)}$  by scanning the first 2 pixels to obtain the  $t$  and  $n$  values. It divides the pixels into groups. Each group contains  $n$  pixels and replaces the  $l$  ( $1 \leq l \leq 7$ ) LSB of the  $t$ -th pixel in each group with the secret information. As a result, the  $n$  encrypted images  $Em^{(t)}$ ,  $t = 1, 2, \dots, n$  with the secret information embedded can be obtained. The embedding rate of this method is  $\lfloor \frac{WH-2}{n} \rfloor \cdot l \approx \frac{l}{n}$  bpp (bit per pixel), where  $W$  and  $H$  are the width and height of the image, respectively.

In this part of information extraction and image decryption, it is assumed that the first  $k$  shared images are  $Em^{(t)}$ ,  $t = 1, 2, \dots, k$  and the key is  $kd$ . In the information extraction phase, for any shared image  $Em^{(t)}$ ,  $t = 1, 2, \dots, k$ . By sharing the first 2 pixels of the image,  $t$  and  $n$  can be obtained. Next, divide all the pixels in the B part of  $Em^{(t)}$  into a group, and extract the  $l$ -LSB of the  $t$ -th pixel in each group to obtain the secret information of the shared image  $Em^{(t)}$ . In image decryption, for the pixel  $Em^{(t)}_{i,j}$ ,  $t = 1, 2, \dots, k$  at the coordinates  $(i,j)$ , and the non-zero random integers  $x_{i,j}^{(t)}$ , and  $t = 1, 2, \dots, k$  generated by the key  $kd$ , equation can be obtained through the Lagrange polynomial. (1) Polynomial  $F_{i,j}(x) = \sum_{t=1}^k \left( Em^{(t)}_{i,j} \prod_{a=1, a \neq t}^k \frac{x - X_a}{X_t - X_a} \right) \bmod 251$ , where  $X_a$  is substituted using  $x_{i,j}^{(a)}$ . However,  $Em^{(t)}_{i,j}$ ,  $t = 1, 2, \dots, k$  may not be able to obtain the correct  $F_{i,j}(x)$  because  $l$  bits are hidden into  $l$ -LSB. Without a loss of generality, assuming that the first pixel to share an image is hidden in  $l$ -bit, i.e.,  $Em^{(1)}_{i,j}$  is hidden in  $l$ -bits, then when the  $Em^{(1)}_{i,j}$  of  $l$ -LSB is modified to  $0, 1, 2, \dots, 2^l - 1$ , one of them must be  $Em^{(1)}_{i,j}$  before embedding  $l$ -LSB, which is called the correct  $Em^{(1)}_{i,j}$ . The correct  $Em^{(1)}_{i,j}$  can be identified by using  $F_{i,j}(0) = T_{i,j}^{(0)}$ , where  $T_{i,j}^{(0)}$  is the constant used in the encryption step. After constructing  $F_{i,j}(x)$ , from Equation (1) pixel  $I'_{i,j}$  of the coordinates  $(i,j)$  can be obtained and therefore the image  $I'$  can also be obtained. Finally, the image  $I'$  is restored to the original image  $I$  using the information in part A.

### 3. Proposed Method

In this section, we propose a new approach to the interplay of confidential image sharing and information hiding. Our approach is also based on Shamir’s Secret Sharing technique of confidential sharing. The traditional Shamir ( $k, n$ )-threshold secret sharing technique shares confidential information  $D$  into  $n$  shared information and the original confidential information  $D$  can be calculated by taking out  $k$  of the shared information. This method requires the construction of a  $k-1$  polynomial, as follows:

$$q(x) = a_0 + a_1x + \dots + a_{k-1}x^{k-1} \bmod p \tag{2}$$

where  $a_0 = D$ ,  $p$  is a prime, constant, or coefficient  $a_i < p$ ,  $i = 0, 1, \dots, k - 1$ . If this mechanism is applied to image sharing, for example, for a grayscale image with a gray scale of 256, to share a pixel value  $P$ , the above polynomial is usually used with  $a_0 = P$  and  $p = 251$ . Because the value range of pixel value  $P$  is  $0 \sim 255$ , all pixel values of the image must be pre-processed so that the range of all pixel values becomes  $0 \sim 250$ .

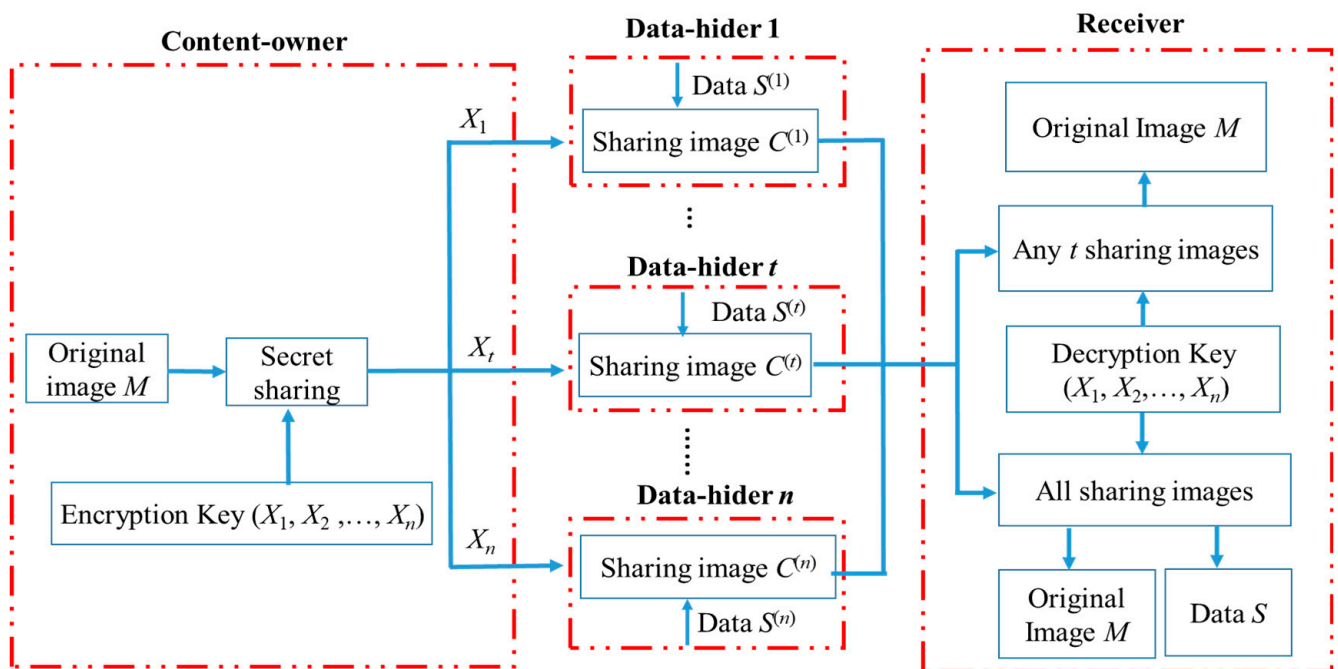
We propose the finite field ( $GF(2^n)$ ) strategy, modifying Shamir’s polynomial formulation, as follows:

$$q(x) = a_0 + a_1x + \dots + a_{k-1}x^{k-1} \tag{3}$$

where the constants or coefficients are  $a_i \in GF(2^8)$ ,  $i = 0, 1, \dots, k - 1$  and both multiplication and addition are performed using the  $GF(2^n)$  operator. Since the  $GF(2^n)$  quadrature is closed, i.e., if  $a, b \in GF(2^n)$ , then  $a + b \in GF(2^n)$ ,  $a - b \in GF(2^n)$ ,  $a \times b \in GF(2^n)$ , and  $a/b \in GF(2^n)$ , the polynomial can be directly applied to image sharing, and in the case of grayscale images with  $256 (=2^8)$  gray levels, no pixel-specific pre-processing is required at all.

Our method has three roles: Content-owner, Data-hiders, and Receivers, as shown in Figure 4, which shows the operation flow of our method in image encryption, data

embedding, data extracting, and image decryption. We take Shamir’s  $(k, n)$ -threshold as an example, where  $n$  means  $n$  shared images are generated and  $k$  means  $k$  shared images are obtained to decrypt the original image. First of all, the image owner can generate  $n$  shared images through Shamir’s Secret Sharing technology and Galois field 256 for confidential sharing of the original image  $M$ . The key  $X_1, \dots, X_n$  are the keys required to generate  $C^{(1)}, C^{(2)}, \dots, C^{(n)}$ , respectively. Next, the data-hider has a separate pair of  $n$  shared images to embed the confidential data  $S^{(n)}$  embedded in  $n$  in a shared image to generate an embedded shared image  $C'^{(1)}, C'^{(2)}, \dots, C'^{(n)}$ . Lastly,  $n$  image recipients receive the embedded shared image, respectively  $C'^{(1)}, C'^{(2)}, \dots, C'^{(n)}$ . Any image recipient with an embedded key  $X_1, \dots, X_n$  can retrieve the confidential information embedded by the data-hider from the shared image. In addition, as long as any  $k$  of the embedded shared images can be decrypted, assuming that the  $k$  embedded shared images are  $C'^{(1)}, \dots, C'^{(k)}$  then using the key  $X_1, \dots, X_k$  can decrypt the original image  $M$  and retrieve confidential data  $S$ . Linear operations on polynomials through a Galois domain, whose addition and multiplication are similar to normal addition and multiplication, except that the result of the operations are elements of the domain.



**Figure 4.** Our proposed diagram in image encryption, data embedding, data extraction and image decryption.

### 3.1. Image Encryption

For the image encryption part, we use Shamir’s confidential sharing and pixel grouping strategy. Figure 5 shows our confidential sharing process. We group the original pixels, if for Shamir’s sharing  $(k, n)$ -threshold, each group contains  $k$  pixels, in the sharing process of group  $(P_{i,j}, \dots, P_{i,j+k-1})$ , we use these pixel values to build  $k - 1$  times polynomial, then use  $X_1, X_2, \dots, X_n$  substituted into the polynomial to obtain the shared pixels  $y^{(1)}_{i,j}, \dots, y^{(n)}_{i,j}$ . Each shared pixel,  $y^{(t)}_{i,j}$ , can be split into  $k$  pixels to produce  $C^{(t)}_{i,j}, \dots, C^{(t)}_{i,j+k-1}$ . Since the group  $(C^{(t)}_{i,j}, \dots, C^{(t)}_{i,j+k-1})$  holds only  $y^{(t)}_{i,j}$  with 8 bits, so the group can be empty of  $8k - 8 = 8(k - 1)$  bits of space.

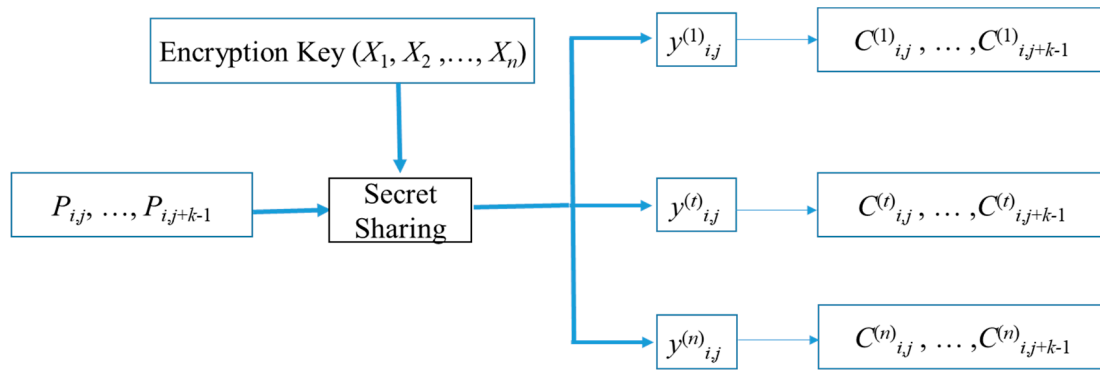


Figure 5. Secret sharing process for pixel group  $(P_{i,j}, \dots, P_{i,j+k-1})$ .

Here is our cryptographic algorithm. Our Algorithm 1 is illustrated with Shamir  $(k, n)$ -thresholds, so the polynomial is a  $(k - 1)$ -th polynomial.

---

**Algorithm 1:** Image Sharing

---

**Input:** Image  $M$ , Encryption keys  $X_t, t = 1, 2, \dots, n$ .

**Output:** Sharing images  $C^{(t)}, t = 1, 2, \dots, n$ .

Step 1: A set of pixels  $(P_{i,j}, \dots, P_{i,j+k-1})$  is extracted from Image  $M$ . The polynomial  $P_{i,j+k-1}x^{k-1} + P_{i,j+k-2}x^{k-2} + \dots + P_{i,j}$  is generated using the set of pixels  $(P_{i,j}, \dots, P_{i,j+k-1})$ .

Step 2: Substitute the Encryption key  $X_t, t = 1, 2, \dots, n$  through 256 Galois field into the polynomial  $P_{i,j+k-1}x^{k-1} + P_{i,j+k-2}x^{k-2} + \dots + P_{i,j}$  to obtain  $n$  encrypted pixel values  $y^{(t)}_{i,j}$ , where  $y^{(t)}_{i,j} = P_{i,j+k-1}x_t^{k-1} + P_{i,j+k-2}x_t^{k-2} + \dots + P_{i,j}, t = 1, 2, \dots, n$ .

Step 3: The encrypted pixel values  $y^{(t)}_{i,j}$  are divided into 8-bits, and each bit is put into the pixels of the shared image  $C^{(t)}$  in order  $(C^{(t)}_{i,j}, \dots, C^{(t)}_{i,j+k-1})$ , and it is called the P part, if  $k \leq 7$ , because  $k < 8$  cannot put all the bits at once, so the remaining bits are put in order again and the remaining vacant part is called Part B.

Step 4: Repeat Step 2 and Step 3 until all set of pixels  $(P_{i,j}, \dots, P_{i,j+k-1})$  have been processed.

Step 5: Output  $n$  encrypted shared images  $C^{(t)}, t = 1, 2, \dots, n$ .

---

3.2. Data Embedding

In the image embedding part, when the  $t_{th}$  data-hider receives the shared image  $C^{(t)}$ , it embeds each pixel of  $C^{(t)}$ , and through the B part, it embeds the secret message  $S^{(t)}$  into  $C^{(t)}$  one after another. Figure 6 shows the embedding process of data-hider  $t$  for the pixel group  $(C^{(t)}_{i,j}, \dots, C^{(t)}_{i,j+k-1})$ . Since group  $(C^{(t)}_{i,j}, \dots, C^{(t)}_{i,j+k-1})$  has a space of  $8(k - 1)$  bits, part of the confidential data  $S^{(t)}$  can be directly embedded in this space, resulting in the embedded shared group  $(C'^{(t)}_{i,j}, \dots, C'^{(t)}_{i,j+k-1})$ . The embedding rate of our method is fixed, and the embedding rate does not decrease as the number of shared images increases. The following is our embedding Algorithm 2.

---

**Algorithm 2:** Data Embedding

---

**Input:** Sharing images  $C^{(t)}, t = 1, 2, \dots, n$ , Data  $S^{(t)}$

**Output:** Marked sharing images  $C'^{(t)}, t = 1, 2, \dots, n$

Step 1: For a pixel  $C^{(t)}_{i,j}$  of Sharing images  $C^{(t)}$ , the message is taken from the secret message  $S^{(t)}$  and embedded in the B part of pixel  $C^{(t)}_{i,j}$ .

Step 2: Repeat Step 1 until all the B parts of  $C^{(t)}_{i,j}$  are embedded in the secret message.

Step 3: The output of the embedded shared image  $C'^{(t)}$ .

---

3.3. Data Extracting and Image Decryption

In the part of data extraction and image decryption, the  $t_{th}$  image recipient receives the shared image  $C'^{(t)}$ , and can extract the confidential information  $S^{(t)}$  embedded by the  $t_{th}$  data-hider. In addition, by collecting the shared images and decryption keys above the

Threshold, a polynomial can be constructed for each set of pixels, through which the pixels of the original image and the secret information of the image owner can be extracted and the original image  $M$  can be recovered.

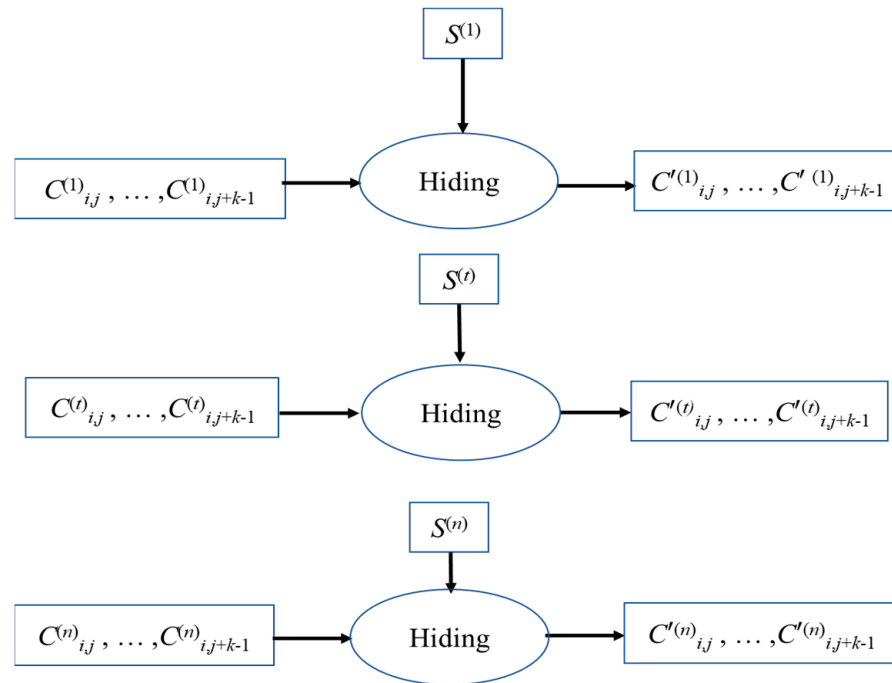


Figure 6. Embedding process for pixel group  $(C^{(t)}_{ij}, \dots, C^{(t)}_{ij+k-1}), t = 1, 2, \dots, n$ .

Figure 7 shows the process of extracting the cluster  $(C'^{(t)}_{ij}, \dots, C'^{(t)}_{ij+k-1})$  by Receiver  $t$ . The pixel group  $(C'^{(t)}_{ij}, \dots, C'^{(t)}_{ij+k-1})$  has been embedded in a confidential message of  $8(k - 1)$  bits and can be taken out directly into a partial  $S^{(t)}$ . After extraction, there is no need to reduce this group to  $(C^{(t)}_{ij}, \dots, C^{(t)}_{ij+k-1})$ , because the group can obtain  $y^{(t)}_{ij}$  whether it is reduced or not.

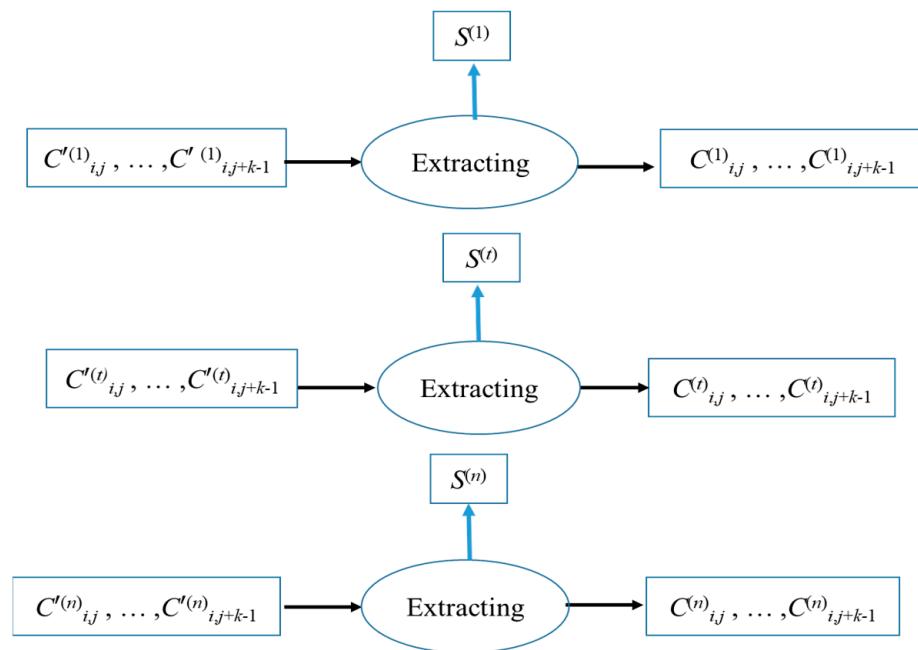


Figure 7. Extraction process for the pixel group  $(C'^{(t)}_{ij}, \dots, C'^{(t)}_{ij+k-1}), t = 1, 2, \dots, n$ .



Figure 8 shows the response flow diagram of the pixel group  $(P_{i,j}, \dots, P_{i,j+k-1})$ . Assuming that the first  $k$  shared images are obtained from  $C^{(1)}, C^{(2)}, \dots, C^{(k)}$ , the pixels of each shared image are also grouped, and each group contains  $k$  pixels. For the 7th shared image  $C^{(t)}$  for which the pixel group  $(C^{(t)}_{i,j}, \dots, C^{(t)}_{i,j+k-1})$  can be taken out to share pixels  $y^{(t)}_{i,j}$ . Using  $y^{(t)}_{i,j}, t = 1, 2, \dots, k$  and  $X_t, t = 1, 2, \dots, k$  and Lagrange polynomials, the originally constructed  $(k - 1)$ -th polynomial can be recovered, and therefore the pixel group  $(P_{i,j}, \dots, P_{i,j+k-1})$  can be recovered.

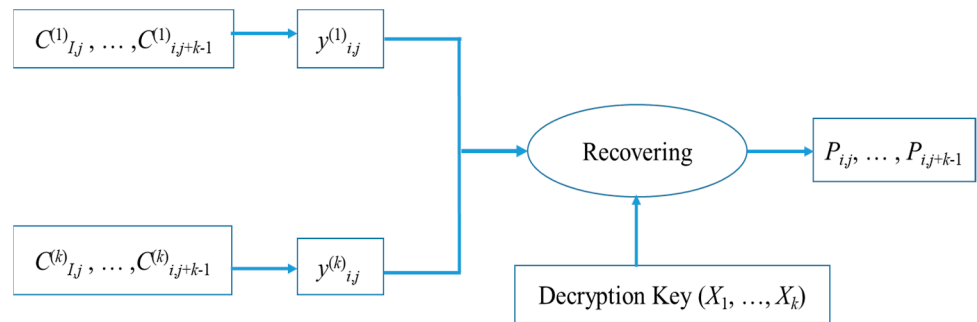


Figure 8. Flowchart of recovery of pixel group  $(P_{i,j}, \dots, P_{i,j+k-1})$ .

The following is our Algorithm 3 for recovering images and extracting secret information. For the sake of illustration, we assume that we obtain the first  $k$  marked shared images  $C^{(1)}, \dots, C^{(k)}$  and use the key  $X_1, \dots, X_k$  to decrypt the original images  $M$  and extract the secret information  $S^{(1)}, \dots, S^{(k)}$ .

$$F_{i,j}(x) = \sum_{t=1}^k \left( y^{(t)}_{i,j} \prod_{a=1, a \neq t}^k \frac{x - X_a}{X_t - X_a} \right) \tag{4}$$

where  $y^{(t)}_{i,j}, X_t \in GF(256), t = 1, 2, \dots, k$ , and add, subtract, multiply, and divide are all in  $GF(256)$ .

**Algorithm 3:** Data extracting and image decryption

- Input:** Marked sharing images  $C^{(1)}, \dots, C^{(k)}$ , Decryption keys  $X_1, \dots, X_k$   
**Output:** Data  $S^{(1)}, S^{(2)}, \dots, S^{(k)}$ , Original image  $M$
- Step 1. Take part B of the image  $C^{(1)}, \dots, C^{(k)}$  to obtain the secret information  $S^{(1)}, \dots, S^{(k)}$ .
  - Step 2. For sharing a set of pixels of the image  $C^{(t)}$  ( $C^{(t)}_{i,j}, \dots, C^{(t)}_{i,j+k-1}$ ), extract the P part of the pixel set and merge it into the encrypted pixel values  $y^{(t)}_{i,j}, t = 1, 2, \dots, k$ .
  - Step 3. Bring the encrypted pixel values  $y^{(1)}_{i,j}, y^{(2)}_{i,j}, \dots, y^{(k)}_{i,j}$  with the decryption keys  $X_1, \dots, X_k$  into Lagrange polynomial of Equation (4), build the polynomial  $P_{i,j+k-1}x^{k-1} + P_{i,j+k-2}x^{k-2} + \dots + P_{i,j}$ .
  - Step 4. The coefficients of the polynomial  $P_{i,j+k-1}x^{k-1} + P_{i,j+k-2}x^{k-2} + \dots + P_{i,j}$  are used to obtain the pixel sets  $(P_{i,j}, \dots, P_{i,j+k-1})$  of the original images.
  - Step 5. Repeat Step 2 to Step 4 until all pixel groups  $(P_{i,j}, \dots, P_{i,j+k-1})$  have been processed.
  - Step 6. Combine all pixel groups  $(P_{i,j}, \dots, P_{i,j+k-1})$  into the original image  $M$ .
  - Step 7. Output the original image  $M$  and the secret information  $S^{(1)}, S^{(2)}, \dots, S^{(k)}$ .

3.4. Example

We give an example of Shamir’s (3, 3)-threshold. Assuming the secret information  $S^{(1)} = 10000\ 10000\ 010000\ (2), S^{(2)} = 01000\ 01000\ 001000\ (2), S^{(3)} = 00100\ 00100\ 000100\ (2)$ , the pixel values of the pixel group  $(P_{i,j}, \dots, P_{i,j+k-1})$  of the original image are  $P_{(i,j)} = 2\ (10), P_{(i,j+k)} = 4\ (10), P_{(i,j+2)} = 8\ (10)$ . The pixel group  $(P_{i,j}, \dots, P_{i,j+k-1})$  generates 3 pixel groups of shared images after embedding secret information  $(C^{(1)}_{i,j}, \dots, C^{(1)}_{i,j+k-1}), (C^{(2)}_{i,j}, \dots, C^{(2)}_{i,j+k-1}), (C^{(3)}_{i,j}, \dots, C^{(3)}_{i,j+k-1})$ .

### 3.4.1. Encryption and Embedding Steps

The set of pixel values of the original image are  $P_{(i,j)} = 2_{(10)} = 10_{(2)}$ ,  $P_{(i,j+k)} = 4_{(10)} = 100_{(2)}$ , and  $P_{(i,j+2)} = 8_{(10)} = 1000_{(2)}$ , respectively. Using the pixel values  $P_{(i,j)}$ ,  $P_{(i,j+k)}$ , and  $P_{(i,j+2)}$  to construct the polynomial  $P_{i,j+k-1}x^{k-1} + P_{i,j+k-2}x^{k-2} + \dots + P_{i,j}$ , we obtain  $8x^2 + 4x + 2$ . Then, we start the encryption, at this time we assume the encryption keys  $X_t$  are  $X_1 = 10$ ,  $X_2 = 20$ ,  $X_3 = 30$ , substitute them into  $8x^2 + 4x + 2$  and get 3 encrypted shared pixels by  $GF(256)$ , respectively,  $y^{(1)} = 60_{(10)} = 00111100_{(2)}$ ,  $y^{(2)} = 10_{(10)} = 00001010_{(2)}$ ,  $y^{(3)} = 52_{(10)} = 00110100_{(2)}$ . Finally, the encrypted pixel values are divided into 8 bits and stored in the pixel group  $(C^{(t)}_{i,j}, \dots, C^{(t)}_{i,j+k-1})$ ,  $t = 1, 2, 3$ , called the P part. We can get

$$\begin{aligned} C^{(1)}_{i,j} &= \underline{010}00000_{(2)}, C^{(1)}_{i,j+1} = \underline{010}00000_{(2)}, C^{(1)}_{i,j+2} = \underline{11}000000_{(2)}. \\ C^{(2)}_{i,j} &= \underline{001}00000_{(2)}, C^{(2)}_{i,j+1} = \underline{010}00000_{(2)}, C^{(2)}_{i,j+2} = \underline{00}000000_{(2)}. \\ C^{(3)}_{i,j} &= \underline{010}00000_{(2)}, C^{(3)}_{i,j+1} = \underline{000}00000_{(2)}, C^{(3)}_{i,j+2} = \underline{11}000000_{(2)}. \end{aligned}$$

Assuming  $S^{(1)} = 1000010000_{(2)}$ ,  $S^{(2)} = 0100001000_{(2)}$ ,  $S^{(3)} = 0010000100_{(2)}$ , the data-hider targets the pixels of the three shared images  $C^{(1)}$ ,  $C^{(2)}$ , and  $C^{(3)}$ , respectively, the secret information  $S^{(1)}$ ,  $S^{(2)}$ , and  $S^{(3)}$  embedded in part B to obtain the images  $C'^{(1)}$ ,  $C'^{(2)}$ ,  $C'^{(3)}$ :

$$\begin{aligned} C'^{(1)}_{i,j} &= 010\underline{10000}_{(2)}, C'^{(1)}_{i,j+1} = 010\underline{10000}_{(2)}, C'^{(1)}_{i,j+2} = 11\underline{010000}_{(2)}. \\ C'^{(2)}_{i,j} &= 001\underline{01000}_{(2)}, C'^{(2)}_{i,j+1} = 010\underline{01000}_{(2)}, C'^{(2)}_{i,j+2} = 00\underline{001000}_{(2)}. \\ C'^{(3)}_{i,j} &= 010\underline{00100}_{(2)}, C'^{(3)}_{i,j+1} = 000\underline{00100}_{(2)}, C'^{(3)}_{i,j+2} = 11\underline{000100}_{(2)}. \end{aligned}$$

### 3.4.2. Decryption and Extracting Steps

We use 3 shared images to illustrate the steps of decryption and extracting. Take out the pixel sets of 3 shared image images  $(C'^{(1)}_{i,j}, \dots, C'^{(1)}_{i,j+k-1})$ ,  $(C'^{(2)}_{i,j}, \dots, C'^{(2)}_{i,j+k-1})$ ,  $(C'^{(3)}_{i,j}, \dots, C'^{(3)}_{i,j+k-1})$  and the decryption keys  $X_1, X_2, X_3$  and take out each pixel set of B to obtain the secret information  $S$ . The P parts of each pixel group are merged individually to obtain the shared image pixels  $y^{(1)}$ ,  $y^{(2)}$ , and  $y^{(3)}$ . Substitute  $X_1, X_2, X_3$ , and  $y^{(1)}, y^{(2)}, y^{(3)}$  into Equation (4) to obtain the original polynomial, where  $y^{(1)} = 60$ ,  $y^{(2)} = 10$ ,  $y^{(3)} = 52$ ,  $X_1 = 10$ ,  $X_2 = 20$ ,  $X_3 = 30$ ,

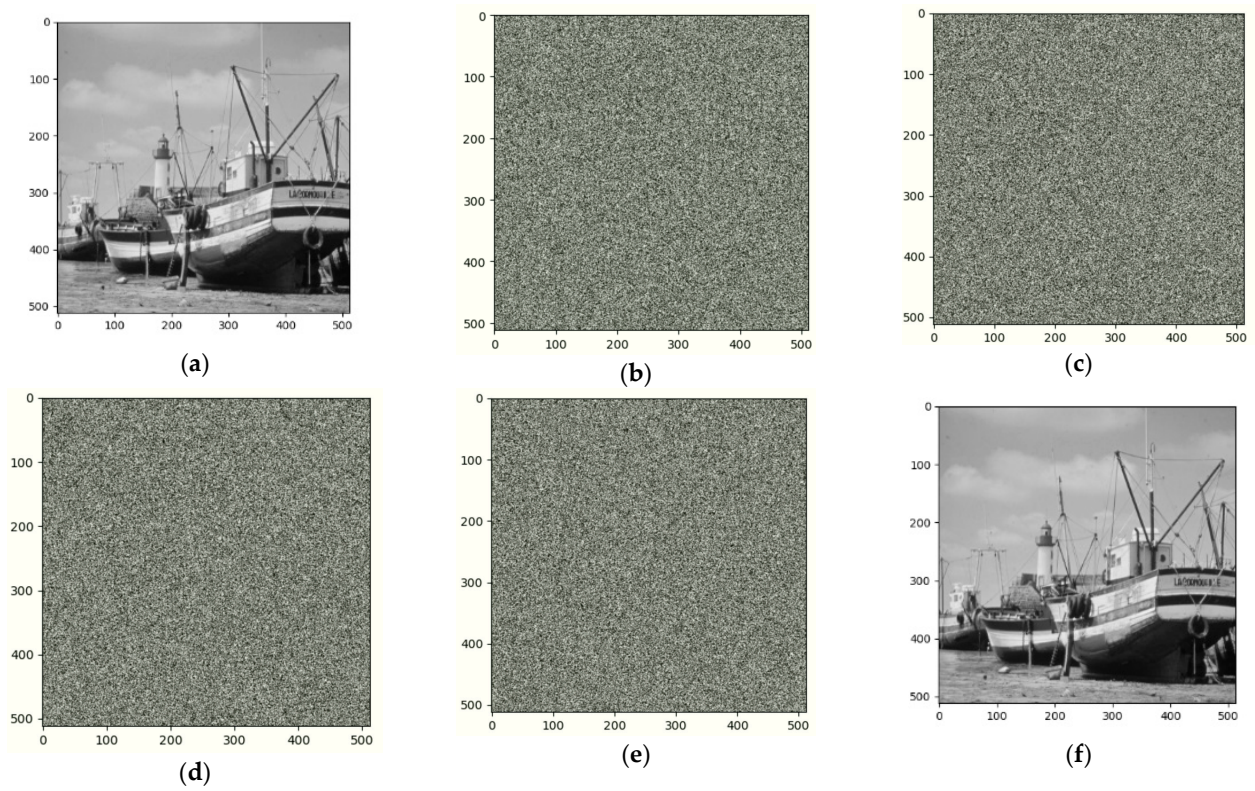
$$\begin{aligned} 60 \times \frac{x-20}{10-20} \times \frac{x-30}{10-30} + 10 \times \frac{x-10}{20-10} \times \frac{x-30}{20-30} + 52 \times \frac{x-10}{30-10} \times \frac{x-20}{30-20} \\ = 8x^2 + 4x + 2 \end{aligned}$$

The polynomial  $P_{i,j+k-1}x^{k-1} + P_{i,j+k-2}x^{k-2} + \dots + P_{i,j} = 8x^2 + 4x + 2$  is obtained to obtain the pixel group of the original image  $(P_{i,j}, \dots, P_{i,j+k-1}) = (2, 4, 8)$ .

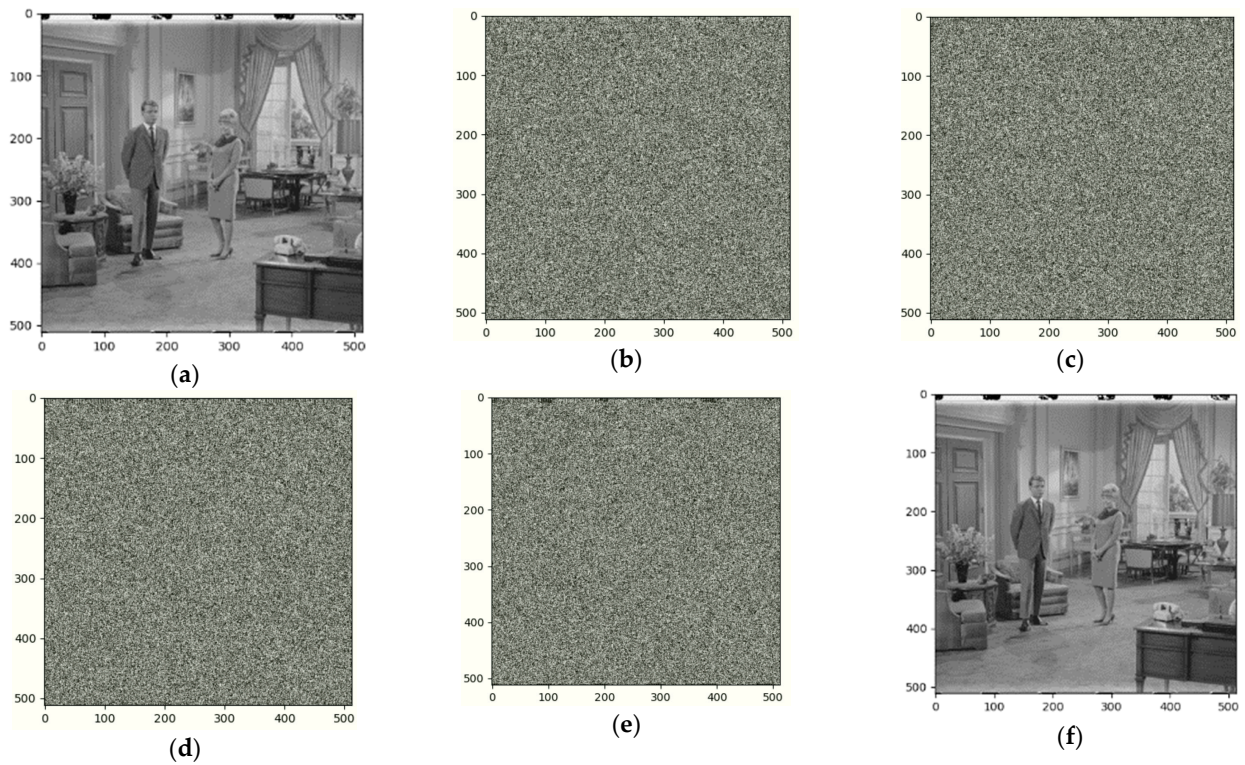
## 4. Experimental Results

In this section, we perform experiments and analysis. All the tested images are gray 425 level sized by  $512 \times 512$ . Figure 9 shows the experimental results of Image Boat. We use the three-out-of-four threshold secret sharing method to group every three original images and encrypt them into four shared images. Figure 9a is the original image, Figure 9b–e are different shared images, and Figure 9f is the image after decryption and information retrieval. From the image we can see that our method can fully recover. Similarly, we did the same for image Couple1 and the result is shown in Figure 10.

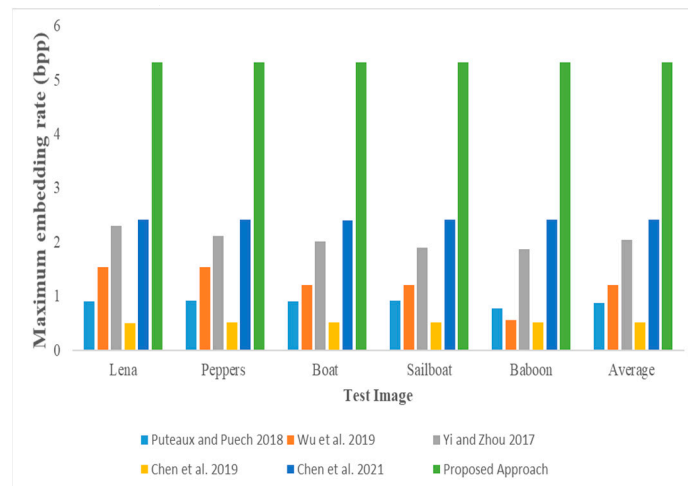
Figure 11 shows the maximum embedding rate comparison among the proposed method and state-of-the-art methods. We use three-out-of-three, three-out-of-four, and three-out-of-five threshold secret sharing to make the comparison. Here is a three-out-of-four to illustrate that, in our way, a shared image pixel can be embedded with 5 or 6 bits of secret information at an embedding rate of 5.3 bpp (Bit Per Pixel). In Chen et al.'s method, their embedding rate is  $\lfloor \frac{WH-2}{WH} \rfloor \cdot l \approx \frac{l}{n}$  bpp, which is about 1.75 bpp since  $n = 4$  and  $l = 7$ . From Figure 11, we can see that the embedding rate of my method is larger than other methods, and it will be more obvious as the number of shared images increases.



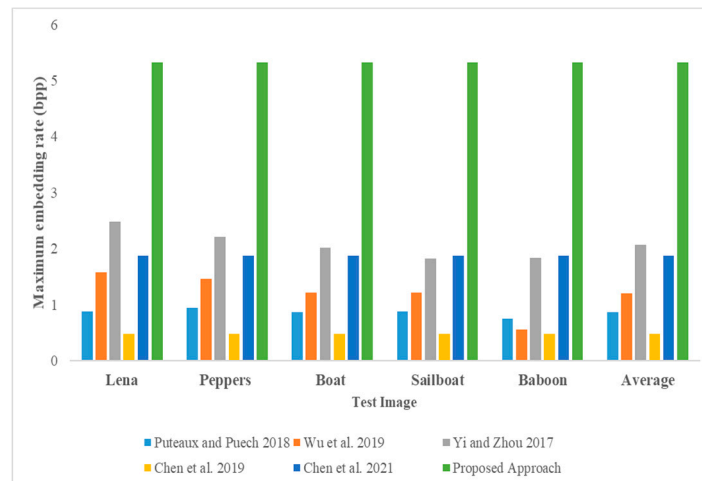
**Figure 9.** Encrypted results of Boat and its recovered image:(a) Original (b) Sharing image 1 (c) Sharing image 2 (d) Sharing image 3 (e) Sharing image 4 (f) Recovered.



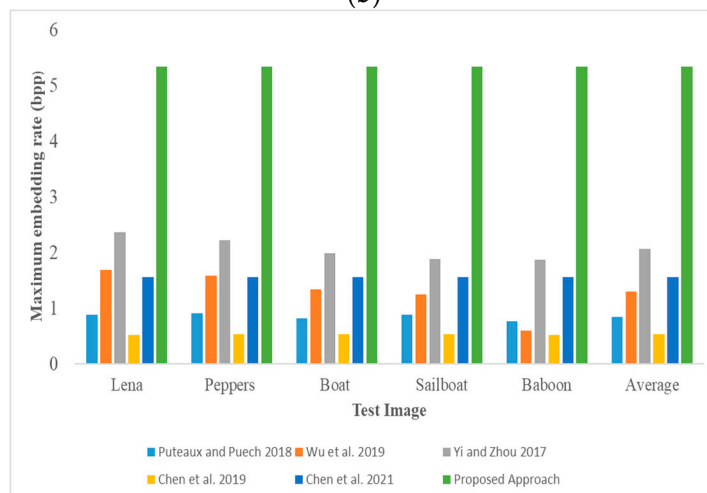
**Figure 10.** Encrypted results of Couple1 and its recovered image: (a) Original (b) Sharing image 1 (c) Sharing image 2 (d) Sharing image 3 (e) Sharing image 4 (f) Recovered.



(a)



(b)



(c)

**Figure 11.** Maximum embedding rate comparison among the proposed method based on Shamir (3,  $n$ ) and state-of-the-art methods [13–15,21,22]: (a)  $n = 3$ ; (b)  $n = 4$ ; (c)  $n = 5$ .

Table 1 shows the feature comparison among the proposed scheme and state-of-the-art schemes, which shows the comparison of the features of our approach with other approaches.

**Table 1.** Feature comparison among the proposed scheme and state-of-the-art schemes.

Scheme	Separable	Vacating Room before Encryption	Encryption Strategy	Participant (Data-Hider)
Puteaux and Puech [13]	Yes	Yes	Stream cipher	Single
Wu et al. [14]	Yes	Yes	Stream cipher	Single
Yi and Zhou [15]	Yes	Yes	Block permutation and modulation	Single
Chen et al. [21]	No	Yes	Secret sharing	Single
Chen et al. [22]	Yes	No	Secret sharing	Multiple
Proposed	Yes	No	Secret sharing	Multiple

Table 2 shows the comparisons of embedding capacity (bits) and embedding rate (bpp), here the experimental data is used in a three-out-of-four threshold secret sharing approach, in our method, a pixel of a shared image can embed 5 or 6 bits of secret information, so a shared image can embed 1,398,096 bits. The bpp (bit per pixel) for embedding rate is  $\frac{1,398,096}{(512 \times 512)} \cong 5.3$ . We share the image as four shared images, so you can embed  $1,398,096 \times 4 = 5,592,384$  bits. Figure 12 shows the images used in Table 2.

**Table 2.** Comparisons of Embedding capacity (bits) and Embedding rate (bpp).

Test Images	Embedding Capacity (bits)	Embedding Rate (bpp)
Lena	5,592,384	5.3
Peppers	5,592,384	5.3
Boat	5,592,384	5.3
Sailboat	5,592,384	5.3
Baboon	5,592,384	5.3
Couple1	5,592,384	5.3
Toys	5,592,384	5.3
Girl	5,592,384	5.3

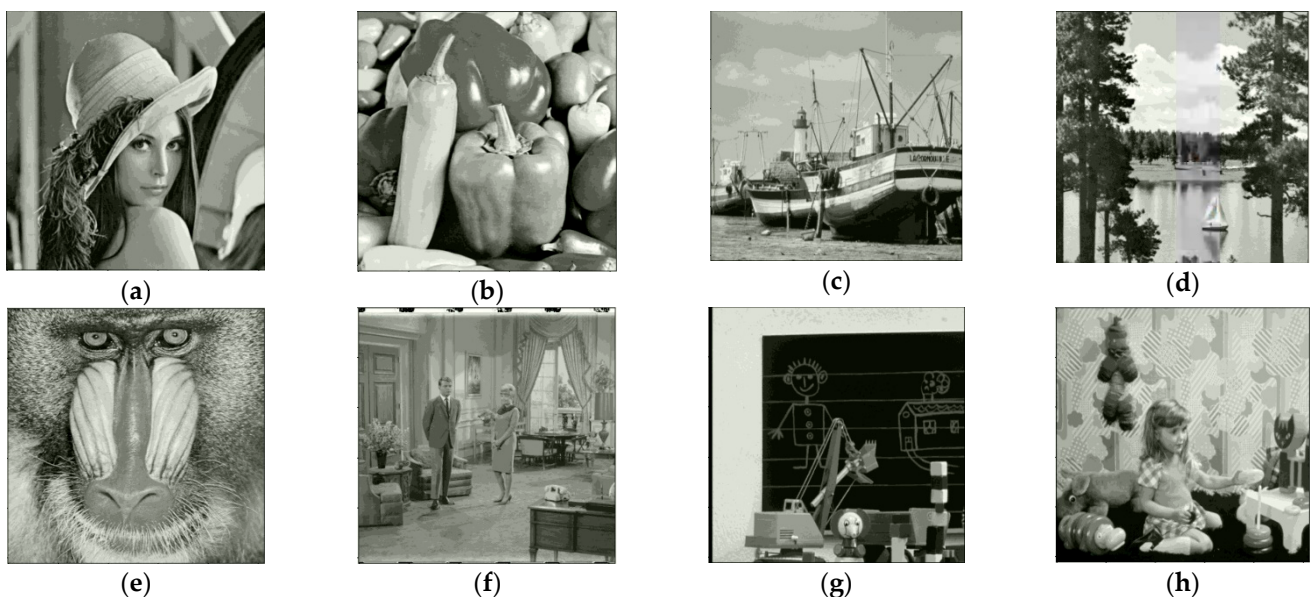
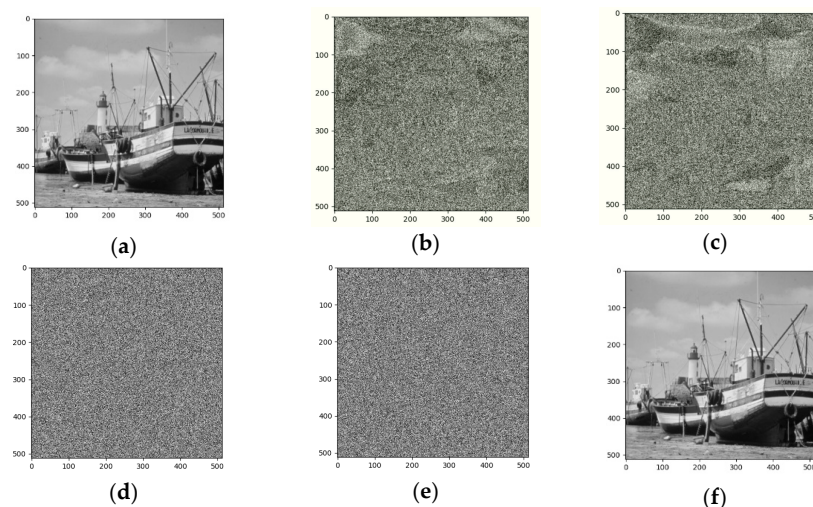
**Figure 12.** Test images: (a) Lena; (b) Peppers; (c) Boat; (d) Sailboat; (e) Baboon; (f) Couple1; (g) Toys; (h) Girl.

Table 3 shows the comparison of embedding rate (bpp) with different  $k$  based on Shamir ( $k, n$ ). In our approach, each original pixel  $P_{i,j}$  could be reconstructed by collecting  $k$  sharing pixels. It means that the  $k$  sharing pixels holds only  $P_{i,j}$  with 8 bits, so  $k$  sharing pixels has a space of  $8 \times (k - 1)$  bits, thus the embedding ratio is estimated by  $\frac{8 \times (k-1)}{k}$ . Assume that  $k = 8$  indicates that a pixel  $P_{i,j}$  can be split into 8 pixels to produce shared pixels  $C^{(1)}_{i,j}, \dots, C^{(8)}_{i,j}$  by Shamir's confidential sharing. Therefore, the embedding ratio is  $\frac{8 \times (8-1)}{8} = 7$ . In the same cases, assume that  $k = 6$  and the embedding ratio is  $\frac{8 \times (6-1)}{6} \cong 6.6$ . From Table 3, we can see that our method increases the embedding rate (bpp) as  $k$  increases for each  $k$  binning of the original images.

**Table 3.** Comparison of Embedding rate (bpp) with different  $k$  based on Shamir( $k, n$ ).

Test Image	$k$					
	3	4	5	6	7	8
Lena	5.3	6	6.4	6.6	6.8	7
Peppers	5.3	6	6.4	6.6	6.8	7
Boats	5.3	6	6.4	6.6	6.8	7
Sailboat	5.3	6	6.4	6.6	6.8	7
Baboon	5.3	6	6.4	6.6	6.8	7
Average	5.3	6	6.4	6.6	6.8	7

If two-out-of-two threshold secret sharing is used for the experiment, the maximum number of entries in the multiplex is only one, which may make the sharing images appear contoured and therefore the encryption effect is not satisfactory. In this case, you just need to add a procedure to encrypt the shared image once more, and then the encrypted image will have the encryption effect. Of course, the decryption process should also add one additional step to the image decryption. We experiment with image Boat. Figure 13a is the original image, Figure 13b,c are the two-out-of-two threshold secret sharing different sharing images, and we can find some contours appear in Figure 13b,c. Figure 13d,e are the results of re-encryption of Figure 13b,c, respectively, which already have the effect of image encryption, while Figure 13d,e are the results of the re-encryption of Figure 13b,c. Figure 13f is the image after decryption and information retrieval.



**Figure 13.** Encrypted results of Boat and its recovered image by 2-out-of-2 threshold secret sharing: (a) Original (b) Sharing image 1 (c) Sharing image 2 (d) Encrypted Sharing image 1 (e) Encrypted Sharing image 2 (f) Recovered.

## 5. Conclusions

This paper proposes an RDHEI technique based on the polynomial construction technique of Shamir's Secret Sharing technique, which divides  $k$  pixels into a group and uses  $k$  pixels to construct a polynomial. Then, our approach uses the polynomial and Galois Field calculation to generate  $n$  sharing pixels. Because of the finite field ( $GF(2^n)$ ) strategy, our method can avoid the pixel-specific pre-processing. The 8 bits of each shared pixel are split into  $k$  small parts and placed in each of the  $k$  pixels of the shared image so that the constructed shared image has B part of the embedding space, and each shared image is encrypted. Information hidiers can hide secret information in Part B of the shared image. Compared with other methods, our method has a higher embedding rate, and the embedding rate does not decrease due to more shared images. In the future work, we will consider how to apply the proposed model to other applications or to some specified multimedia such as video or audio.

**Author Contributions:** Conceptualization, C.-Y.W.; Methodology, C.-Y.W. and C.-H.Y.; Writing—original draft, C.-Y.W.; Supervision, C.-H.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was partially supported by the National Science and Technology Council of the Republic of China under the Grant No. MOST 110-2221-E-153-002-MY2 and MOST 111-2221-E-153-005. And the APC was funded by the National Science and Technology Council of the Republic of China under the Grant No. MOST 110-2221-E-153-002-MY2.

**Institutional Review Board Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** The authors would like to thank Chia-Chun Chiang who is our ISLAB member for providing the part of simulations.

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

## References

1. Luo, L.; Chen, Z.; Chen, M.; Zeng, X.; Xiong, Z. Reversible image watermarking using interpolation technique. *IEEE Trans. Inf. Forensics Secur.* **2010**, *5*, 187–193.
2. Tian, J. Reversible Data Embedding Using a Difference Expansion. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 890–896. [[CrossRef](#)]
3. Wu, H.T.; Huang, J. Reversible image watermarking on prediction errors by efficient histogram modification. *Signal Process.* **2012**, *92*, 3000–3009. [[CrossRef](#)]
4. Xuan, G.; Yao, Q.; Yang, C.; Gao, J.; Chai, P.; Shi, Y.Q.; Ni, Z. Lossless Data Hiding Using Histogram Shifting Method Base on Integer Wavelets. In Proceedings of the International Workshop on Digital Watermarking, LNCS, Jeju Island, Republic of Korea, 8–10 November 2006.
5. Tai, W.L.; Yeh, C.M.; Chang, C.C. Reversible data hiding based on histogram modification of pixel differences. *IEEE Trans. Circuits Syst. Video Technol.* **2009**, *19*, 906–910.
6. Ni, Z.; Shi, Y.; Ansari, N.; Su, W. Reversible data hiding. *IEEE Trans. Circuits Syst. Video Technol.* **2006**, *16*, 354–362.
7. Celik, M.U.; Sharma, G.; Tekalp, A.M.; Saber, E. Lossless generalized-LSB data embedding. *IEEE Trans. Image Process.* **2005**, *14*, 253–266. [[CrossRef](#)] [[PubMed](#)]
8. Fridrich, J.; Goljan, M.; Du, R. Lossless data embedding for all image formats. In Proceedings of the SPIE- International Society for Optical Engineering, San Jose, CA, USA, 19 January 2002.
9. Peng, F.; Zhao, Y.; Zhang, X.; Long, M.; Pan, W.Q. Reversible data hiding based on RSBEMD coding and adaptive multi-segment left and right histogram shifting. *Signal Process. Image Commun.* **2020**, *81*, 1–9. [[CrossRef](#)]
10. Gao, G.; Tonga, S.; Xiaa, Z.; Xu, B.W.L.; Zhao, Z. Reversible data hiding with automatic contrast enhancement for medical images. *Signal Process.* **2021**, *178*, 1–14. [[CrossRef](#)]
11. 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](#)]
12. Cao, X.; Du, L.; Wei, X.; Meng, D.; Guo, X. High capacity reversible data hiding in encrypted image by patch-level sparse representation. *IEEE Trans. Cybern.* **2016**, *46*, 1132–1143. [[CrossRef](#)] [[PubMed](#)]
13. 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](#)]

14. Wu, H.T.; Yang, Z.; Cheung, Y.M.; Xu, L.; Tang, S. High-capacity reversible data hiding in encrypted images by bit plane partition and MSB prediction. *IEEE Access* **2019**, *7*, 62361–62371. [[CrossRef](#)]
15. Yi, S.; Zhou, Y. Adaptive code embedding for reversible data hiding in encrypted images. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017.
16. 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](#)]
17. Yin, Z.; She, X.; Tang, J.; Luo, B. Reversible data hiding in encrypted images based on pixel prediction and multi-MSB planes rearrangement. *Signal Process.* **2021**, *187*, 1–11. [[CrossRef](#)]
18. Qin, C.; Qian, X.; Hong, W.; Zhang, X. An efficient coding scheme for reversible data hiding in encrypted image with redundancy transfer. *Inf. Sci.* **2019**, *487*, 176–192. [[CrossRef](#)]
19. Wang, Y.; Cal, Z.; He, W. A New High Capacity Separable Reversible Data Hiding in Encrypted Images Based on Block Selection and Block-Level Encryption. *IEEE Access* **2019**, *7*, 175671–175680. [[CrossRef](#)]
20. Wu, X.; Weng, J.; Yan, W. Adopting secret sharing for reversible data hiding in encrypted images. *Signal Process.* **2018**, *143*, 269–281. [[CrossRef](#)]
21. Chen, Y.C.; Hung, T.H.; Hsieh, S.H.; Shiu, C.W. A new reversible data hiding in encrypted image based on multi-secret sharing and lightweight cryptographic algorithms. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 3332–3343. [[CrossRef](#)]
22. Chen, B.; Lu, W.; Huang, J.; Weng, J.; Zhou, Y. Secret sharing based reversible data hiding in encrypted images with multiple data-hiders. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 978–991. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.