Hindawi
Security and Communication Networks
Volume 2019, Article ID 7480147, 14 pages
https://doi.org/10.1155/2019/7480147

Research Article

Separable Reversible Data Hiding in Encrypted Images Based on Difference Histogram Modification

Dawen Xu and Shubing Su

School of Electronics and Information Engineering, Ningbo University of Technology, Ningbo, 315016, China

Correspondence should be addressed to Dawen Xu; dawenxu@126.com

Received 7 March 2019; Accepted 10 June 2019; Published 27 June 2019

Academic Editor: David Megias

In this paper, an efficient reversible data hiding method for encrypted image based on neighborhood prediction is proposed, which includes image encryption, reversible data hiding in encrypted domain, and hidden data extraction. The cover image is first partitioned into non-overlapping blocks, and then the pixel value in each block is encrypted by modulo operation. Therefore, the linear prediction difference in the block that satisfies the specific condition is consistent before and after encryption, ensuring that data extraction is completely separable from image decryption. In addition, by using the linear weighting of three adjacent pixels in the block to predict the current pixel, the prediction accuracy can be improved. The data-hider, who does not know the original image content, may embed additional data based on prediction difference histogram modification. Data extraction and image recovery are free of any error. Experimental results demonstrate the feasibility and efficiency of the proposed scheme.

1. Introduction

Cloud computing is revolutionizing the way digital media is stored and processed. However, the privacy and security of the digital media that resides on the cloud server may be questionable, since cloud data center is managed by a third party cloud server. One of the best ways to ensure the security and confidentiality is to encrypt the media. The user first converts the sensitive content into unintelligible form before uploading it to the cloud such that no information is revealed at all. All the processing and calculation in the cloud are performed in the cipher-text domain, and the processing result is provided to the user [1]. The authorized terminal user who has the decryption key can obtain the plaintext data after decrypting. Under this specific circumstance, the cloud service provider is not authorized to access the plaintext content. However, the effective management and reliability protection of massive cipher-text data in the cloud has become an urgent problem to be solved. Data hiding in encrypted domain is a new research field, which can directly embed some additional messages such as owner identity or authentication data, directly into an encrypted data for effective management or tamper detection purposes.

In the past few years, a considerable number of data hiding schemes for encrypted images or videos have been reported in the literature [2–10]. However, in these schemes, the original cover cannot be recovered completely without distortion due to data embedding. Strictly speaking, cloud service providers are not entitled to introduce permanent distortion, especially medical and military images. Consequently, many researchers show their interests in developing reversible data hiding in encrypted images (RDH-EI). Due to reversibility, the original image can be fully recovered after extracting the secret information [11]. In general, an RDH-EI framework has three end users, i.e., the content-owner, data-hider, and receiver. To preserve privacy, the content owner encrypts the original image before sending it to the data-hider. The data-hider embeds some additional information into the encrypted image and has no privilege to access the original content. At the receiver end, the authorized user can extract the hidden information and losslessly recover the original image. This can be used in many privacy-preserving applications such as medical cloud storage and image management.

Generally, existing RDH-EI methods can be classified into three categories, namely, methods by vacating room

after encryption (VRAE) [12–22], methods by reserving room before encryption (RRBE) [23–25], and methods based on homomorphic encryption [26–33]. The early VRAE framework is proposed by Zhang [12, 13] and Hong et al. [14]. The entire data of an uncompressed image are encrypted directly by a stream cipher. Then the data-hider embeds the additional data by modifying a small portion of encrypted data. The advantage is that the operation of end user is simple and efficient. But the embedding capacity is relatively small. More importantly, the accuracy of data extraction and the lossless recovery of original image are not satisfactory. Later, sparse coding is applied in RDH-EI to achieve high image quality [15]. Qian and Zhang [16] proposed a RDH-EI scheme using distributed source coding (DSC). Huang et al. [17] designed a new framework for RDH in encrypted domain, which integrates previous difference histogram shifting based RDH approaches via a new encryption strategy. Zhou *et al.* [18] used a public key modulation mechanism to embed additional data, without access to the encryption key. Recently, a high capacity reversible data hiding approach based on MSB (most significant bit) prediction is presented in [19]. In addition, several other VRAE schemes are reported in [20–22].

In RRBE framework, the embedding room is vacated in the plaintext domain. Ma *et al.* [23] proposed a RRBE method to reserve room from the original image before encryption. Secret information then can be embedded into the reserved space directly. Later, some RRBE methods have been proposed by reserving the space using different techniques [24, 25]. The advantages of this framework are mainly reflected in two aspects: relatively large embedding capacity and pure reversibility. But the data-hider should know the vacated room created by the content owner before encryption; otherwise he/she cannot perform information embedding with RRBE. This will undoubtedly lead to information leakage. In addition to VRAE and RRBE, another type of method is based on homomorphic encryption. Chen et al. [26] first proposed a Paillier cryptosystem based RDH-EI approach. Later, Shiu et al. [27] improved Chen et al.'s method [26] by transplanting difference expansion into homomorphic encryption. In addition, more RDH methods in homomorphic encrypted domain have been investigated in [28–33]. However, the most important problem of homomorphic encryption, such as Paillier cryptosystem, is that it will cause data expansion after encryption.

In this paper, we develop an effective and reliable framework for RDH-EI. In fact, the proposed method belongs to the first category. Its main contribution is the combination of modular addition and prediction error histogram modification. Its advantages are mainly reflected in the following aspects. First of all, room for data hiding does not need to be vacated before encryption. Secondly, it can be fully separable and fully reversible. Thirdly, modular addition operation with additive homomorphism is used for image encryption. Unlike the public key cryptosystem in [26–29], it does not result in data expansion. More importantly, the linear prediction difference in the image block that satisfies certain conditions remains the same before and after encryption, which ensures that data extraction is completely separated from image decryption. Finally, unlike the prediction technology in [31, 32], the proposed loop prediction technique can obtain the prediction difference of each pixel in the block, which greatly increases the carrier for information embedding. In addition, the prediction accuracy can be improved by using the linear weighting of the three adjacent pixel values. Thus it can effectively improve the embedding capacity. The rest of the paper is organized as follows. In Section 2, we present the proposed scheme, which includes image encryption, data embedding in encrypted image, data extraction, and original image recovery. Experimental results and analysis are given in Section 3. Finally in Section 4, conclusions and future work are drawn.

2. Proposed Scheme

The framework of the proposed scheme has been outlined in Figure 1, which shows the various steps that are performed in RDH-EI. It is composed of three phases, i.e., image encryption, data embedding in encrypted image, and data extraction and image recovery. First, the content owner encrypts the original image using an encryption key and sends the encrypted version to the data-hider. On the server side, the data-hider embeds some additional data into the encrypted image using an embedding key. Here, the data-hider is not authorized to access the original content (i.e., plaintext). At the receiving end, an authorized user can extract the hidden data and losslessly recover the original image.

2.1. Image Encryption. Let the size of an original image X be $M \times N$, and each pixel value $\mathrm{x}(i, j)$ lies in the range $[0, 255]$, $0 \le i \le M-1, 0 \le j \le N-1$. The original image is first divided into non-overlapping blocks. If both M and N are multiples of 2, the cover image is divided into W/2×H/2 image blocks with a size of 2×2 as shown in Figure 2. If M or N is not a multiple of 2, the cover image is divided into $\lceil M/2 \rceil \times \lceil N/2 \rceil$ blocks, including $\lfloor M/2 \rfloor \times \lfloor \mathrm{N}/2 \rfloor$ blocks of size 2×2. Here, $\lceil M/2 \rceil$ denotes the smallest integer greater than or equal to $M/2$, and $\lfloor M/2 \rfloor$ is a floor function to obtain the greatest integer less than or equal to $M/2$.

To ensure that pixels in the same block are encrypted with the same random value, the encryption matrix $\mathrm{R} = \{r(i, j) \mid r(i, j) \in [0, 255]\}$ is obtained using the following equation:

$$\mathrm{r}(i, j) = c\left(\left\lfloor \frac{i}{2} \right\rfloor, \left\lfloor \frac{j}{2} \right\rfloor\right) \tag{1}$$

where $\mathrm{C} = \{c(p, q) \mid c(p, q) \in [0, 255], 0 \le p \le \lceil M/2 \rceil, 0 \le q \le \lceil N/2 \rceil\}$ is a pseudo-random matrix generated using pseudo-random number generator (PRNG) with the encryption key K_{en}. After the encryption matrix is obtained, image encryption can be performed by using modulo-256 addition as follows:

$$\mathrm{S} = \mathrm{E}(X, R) = \left(x(i, j) + r(i, j)\right) \bmod 256 = s(i, j) \quad \forall i = 0, 1, \ldots, M-1, \; j = 0, 1, \ldots, N-1 \tag{2}$$

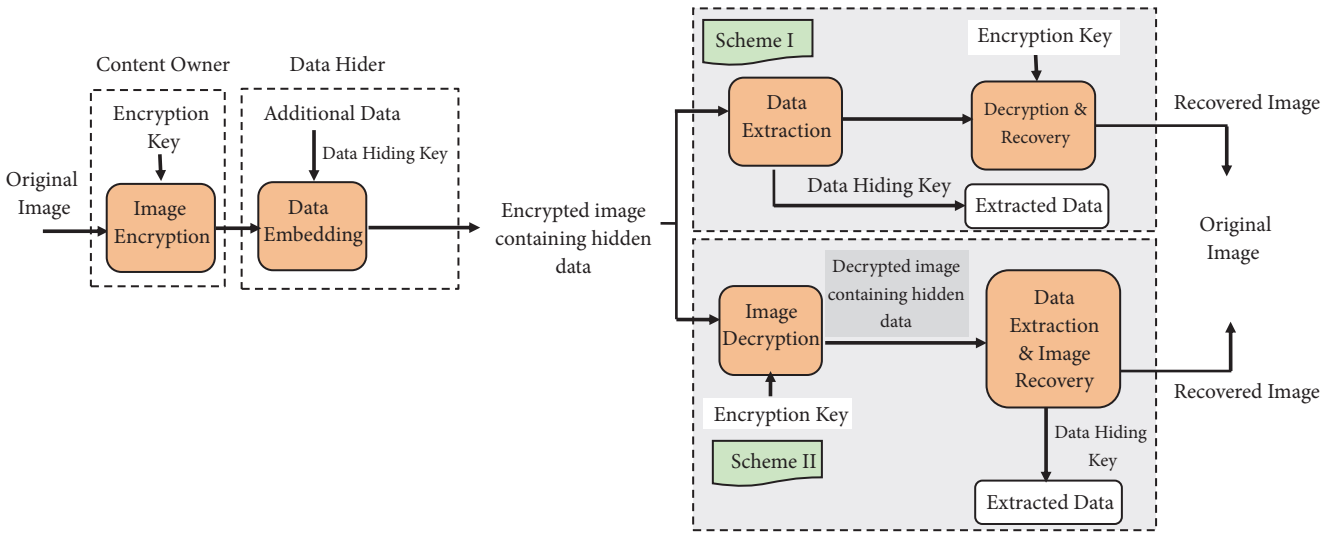


Figure 1: The framework of the proposed scheme.

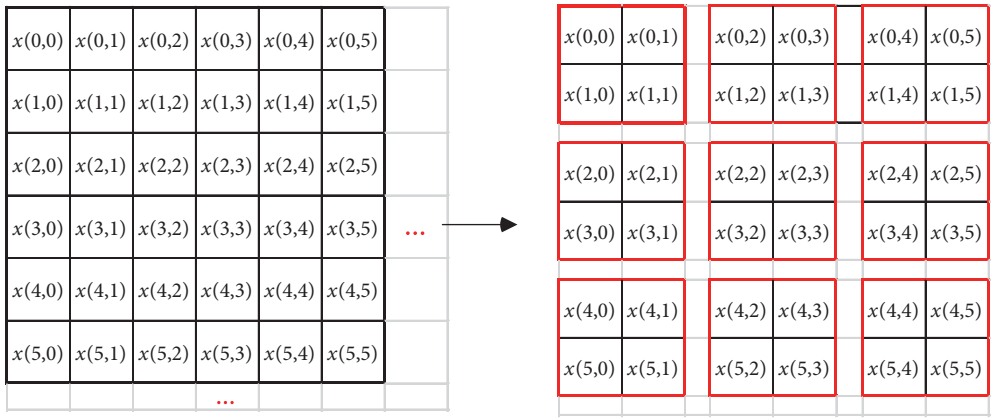


Figure 2: Example of image partition.

where S represents an encrypted image. The corresponding decryption can be done in the following manner:

$$X = D(S, R) = (s(i, j) - r(i, j)) \bmod 256 = x(i, j) \quad \forall i = 0, 1, \ldots, M-1, \; j = 0, 1, \ldots, N-1 \tag{3}$$

According to equation (2), each pixel within an image block is added by the same random integer for modulation. Thus, spatial correlations will be kept within small image blocks, and they can be exploited to embed secret data. Moreover, the main advantages of this algorithm are that they are simple and operate at a high speed.

In order to achieve error-free data extraction and complete reversibility, we also need to determine whether the current block is embeddable during the encryption process. If the current block is 2×2, the value of the current random element in R is added to the pixel values of the four points in the current block, respectively. We denote the 4 pixels in m-th original image block as $x_m{}^1$, $x_m{}^2$, $x_m{}^3$, and $x_m{}^4$. After using the modulo operation with a random value r, the 4 pixels in the new image block are calculated by $s_m{}^l = (x_m{}^l + r) \bmod 256$ (for $l = 1, 2, 3, 4$). If the block is identified as an embeddable block, the following conditions need to be satisfied:

$$x_m{}^{max} + r < 256 \tag{4}$$

and

$$x_m{}^{min} + r \geq 256 \tag{5}$$

where $x_m{}^{max} = \max\{x_m{}^1, x_m{}^2, x_m{}^3, x_m{}^4\}$ and $x_m{}^{min} = \min\{x_m{}^1, x_m{}^2, x_m{}^3, x_m{}^4\}$. Otherwise, the block is identified as a non-embeddable block.

A binary location map L_1 is used to record the locations. Specifically, if the current block is an embeddable block, the corresponding element is marked as "0" in L_1. Otherwise, the element is marked as "1". Since L_1 is mainly composed of zero, it can be compressed with a lossless compression algorithm.

Subsequently, it can be embedded in the marginal area by using LSB replacement. Alternatively, it can be saved as a part of side information and transmitted to the receiver side [34].

2.2. Data Embedding in Encrypted Image. At this phase, the data-hider can embed some secret data into the encrypted image without knowing the image content. The whole process includes difference histogram generation and difference histogram modification.

(1) Difference Histogram Generation. To generate the histogram of prediction errors in encrypted domain, an efficient loop prediction technique is adopted. The detailed procedure can be described as follows.

Step 1. After obtaining the encrypted image, the data-hider divides it into non-overlapping 2x2 blocks by the same way in image encryption. If the width or height of the image is not a multiple of 2, the right edge blocks or the bottom edge blocks whose size are not 2x2 will be ignored during data embedding.

Step 2. According to the location map L_1, it can be determined whether the current block is an embeddable block. If it is an embeddable block, go to Step 3. If it is a non-embeddable block, the next block is taken as the current block and proceed to Step 2.

Step 3. For the encrypted pixel value ${s_m}^1$ in the upper left corner of the m-th image block, the prediction difference is calculated according to the following equation:

$$\begin{aligned} &{e_m}^1 \\ &\quad = \left({s_m}^1 - \left\lfloor w_v \bullet {s_m}^2 + w_h \bullet {s_m}^4 + w_d \bullet {s_m}^3 \right\rfloor\right) \bmod 256 \end{aligned} \tag{6}$$

where ${s_m}^2$, ${s_m}^4$, and ${s_m}^3$ are the three remaining encrypted pixels located in the same column, row, and diagonal directions with ${s_m}^1$. In addition, ${e_m}^1$ is the predicted difference value. Weight coefficients w_h, w_v, and $w_d$are satisfied with $w_h + w_v + w_d = 1, 0 \le w_h, w_v, w_d \le 1$.

Although ${s_m}^1$, ${s_m}^2$, ${s_m}^3$, and ${s_m}^4$ are encrypted values, the following equation is easily proved under the condition that the embeddable block is satisfied:

$$\begin{aligned} &\left({s_m}^1 - \left\lfloor w_v \bullet {s_m}^2 + w_h \bullet {s_m}^4 + w_d \bullet {s_m}^3 \right\rfloor\right) \bmod 256 \\ &\quad = \left({x_m}^1 - \left\lfloor w_v \bullet {x_m}^2 + w_h \bullet {x_m}^4 + w_d \bullet {x_m}^3 \right\rfloor\right) \\ &\quad \cdot \bmod 256 \end{aligned} \tag{7}$$

where ${x_m}^1$ is the original pixel value in the upper left corner of the m-th image block, ${x_m}^4$ is the horizontal adjacent pixel value, ${x_m}^2$ is the vertical adjacent pixel value, and ${x_m}^3$ is the diagonal pixel value.

Proof.

$$\begin{aligned} &\left({s_m}^1 - \left\lfloor w_v \bullet {s_m}^2 + w_h \bullet {s_m}^4 + w_d \bullet {s_m}^3 \right\rfloor\right) \bmod 256 \\ &\quad = \left(\left({x_m}^1 + r_m\right) \bmod 256 - \left\lfloor w_v \right. \right. \\ &\quad \bullet \left(\left({x_m}^2 + r_m\right) \bmod 256\right) + w_h \\ &\quad \bullet \left(\left({x_m}^4 + r_m\right) \bmod 256\right) + w_d \\ &\quad \left. \left. \bullet \left(\left({x_m}^3 + r_m\right) \bmod 256\right) \right\rfloor\right) \bmod 256 \end{aligned} \tag{8}$$

According to Section 2.1, if the current block is embeddable, then ${x_m}^1 + r_m$, ${x_m}^4 + r_m$, ${x_m}^2 + r_m$, and ${x_m}^3 + r_m$ are greater than 255 or less than 256 at the same time. Consequently, the above equation can be simplified to the form of equation (7). The setting of the weighting coefficient has a certain influence on the prediction accuracy. For the sake of simplicity, we set $w_h = 0.4$, $w_v = 0.4$, and $w_d = 0.2$. □

It can be seen from the above proof that the prediction difference remains unchanged before and after encryption. All other 2x2 blocks can be processed in the same manner. There is a high degree of correlation between adjacent pixels in a local region of an image. That is, they have similar gray values, or even the same gray value. Thus, the resulting difference histogram has a higher peak than the histogram of the original image. The coefficient histogram is usually defined as

$$\mathrm{h}(t) = \#\left\{{e_m}^1 \mid {e_m}^1 = t\right\} \tag{9}$$

where # denotes the cardinal number of a set and m represents the block number. The difference histograms of some gray images in Figure 3 illustrate the distribution of the prediction errors.

(2) Difference Histogram Modification. In data embedding procedure, for each block, we use equation (6) to calculate the prediction error, which is utilized for secret data embedding. After the first round of embedding, the pixel value ${s_m}^1$ in each block is changed to $\widehat{{s_m}^1}$ as shown in Figure 4(b). In the second round of embedding, the modified value $\widehat{{s_m}^1}$ will be used together with ${s_m}^3$ and ${s_m}^4$ to prediction ${s_m}^2$ and it will be changed to $\widehat{{s_m}^2}$ as given in Figure 4(c). The remaining two rounds of modification are similar. After four rounds of data embedding, all pixels in each block are modified, as shown in Figure 4(e).

Without loss of generality, we use the first round of embedding to describe the data embedding procedure. First, find the highest bins in the left and right side of the difference histogram, denoted by T_p and T_n, respectively. That is,

$$T_p = \arg\max_{h(t) \in [0,127]} num\left(h(t)\right) \tag{10}$$

and

$$T_n = \arg\max_{h(t) \in [128,255]} num\left(h(t)\right) \tag{11}$$

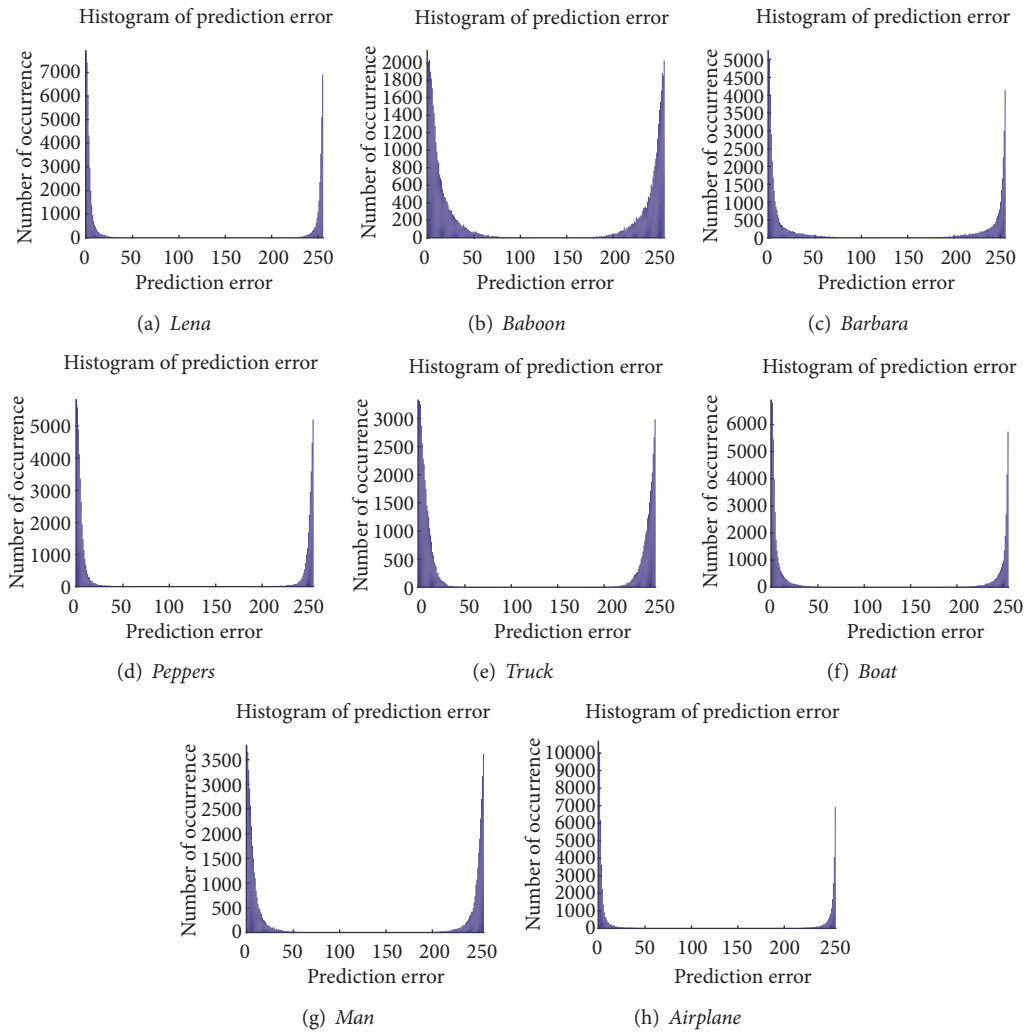


Figure 3: Histogram of prediction difference.

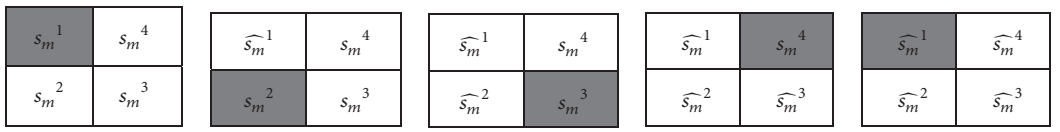


Figure 4: Data embedding using loop prediction.

The embedding zone Z_0which determines where the messages will be embedded is defined as

$$Z_0 = [T_n - \beta, T_n] \cup \left[T_p, T_p + \beta\right] \tag{12}$$

where $\beta \geq 0$ is a scale factor. In this case, the capacity C can be calculated as follows:

$$\mathrm{C} = \sum_{h(t)=T_p}^{T_p+\beta} num\left(h\left(t\right)\right) + \sum_{h(t)=T_n-\beta}^{T_n} num\left(h\left(t\right)\right) \tag{13}$$

The difference histogram modification in the encrypted domain can be described as follows:

$$\widehat{e_m}^1 = \begin{cases} e_m{}^1 + (\beta + 1) & if\ \left(T_p + \beta\right) < e_m{}^1 < (127 - \beta) \\ 2 * e_m{}^1 - T_p + w\left(l\right) & if e_m{}^1 \epsilon \left[T_p, T_p + \beta\right] \\ 2 * e_m{}^1 - T_n - w\left(l\right) & if e_m{}^1 \epsilon \left[T_n - \beta, T_n\right] \\ e_m{}^1 - (\beta + 1) & if\ (128 + \beta) < e_m{}^1 < (T_n - \beta) \\ e_m{}^1 & otherwise \end{cases} \tag{14}$$

where $w(l) \in \{0, 1\}$ is one bit of the secret data and $\widehat{e_m}^1$ is the modified prediction error. In general, to achieve a higher security, a stream cipher with key K_{em} is used to encrypt the secret data before embedded into the encrypted image. Unauthorized users will have difficulty recovering the original message because they do not have the key. Finally, the modified pixel values $\widehat{s_m}^1$ will be obtained as follows:

$$\widehat{s_m}^1 = \left(s_m{}^1 + \widehat{e_m}^1\right) \bmod 256 \tag{15}$$

According to the statistical distribution of the difference histogram in Figure 3, it can be seen that the probability of occurrence is larger when the prediction difference is closer to 0 or 255. For simplicity, we set $T_p = 0$ and $T_n = 225$. For better illustration, the graphical representation of histogram shifting is shown in Figure 5 intuitively. Although the middle part of the difference histogram is usually empty, ambiguities arise when the bins from two sides overlapped in the middle after expansion. To avoid it, the differences in $[127 - \beta, 127]$ and $[128, 128 + \beta]$ will not be shifted. However, ambiguities still arise when difference is changed from $[127-2\beta-1, 127-\beta-1]$ to $[127 - \beta, 127]$ or from $[128 + \beta + 1, 128 + 2\beta + 1]$ to $[128, 128+\beta]$ during the embedding process. The overlapping problem can be resolved by using a location map L_2. It is a binary array with its every element corresponding to $[127 - \beta, 127]$ and $[128, 128 + \beta]$, 0 for genuine and 1 for pseudo. The location map and the additional information will be embedded together into the encrypted domain.

2.3. Data Extraction and Original Image Recovery. At the receiver side, data extraction and image decryption are completely separable. In other words, the hidden data can be extracted before or after decryption. Next, we will introduce these two schemes of data extraction in detail.

(1) Scheme I: Data Extraction in the Encrypted Domain. When holding the encrypted image containing secret information and the data hiding key K_{em}, the receiver can extract the secret information directly. It can be operated in the reverse process of data embedding.

Step 1. Divide the marked and encrypted image into non-overlapping 2x2 blocks as in data embedding phase.

Step 2. According to the location map L_1, if the current block is an embeddable block, go to Step 3. If it is a non-embeddable block, the next block is taken as the current block and proceed to Step 2.

Step 3. Note that data extraction is in the reverse order of embedding procedures, i.e., from $\widehat{s_m}^4$ to $\widehat{s_m}^1$. For each block, the remaining three pixels $\widehat{s_m}^1$, $\widehat{s_m}^2$, and $\widehat{s_m}^3$ are utilized to predict $\widehat{s_m}^4$, and the obtained prediction error values are used for data extraction. After the first round of data extraction, $\widehat{s_m}^4$ is changed as $s_m{}^4$. Then, $s_m{}^4$ together with $\widehat{s_m}^1$ and $\widehat{s_m}^2$ are utilized to predict $\widehat{s_m}^3$. After data extraction, $\widehat{s_m}^3$ is changed as $s_m{}^3$. Similarly, $s_m{}^1$ and $s_m{}^2$ can be obtained. For simplicity, the pixel in the upper left corner is still taken as an example.

For the encrypted pixel value $\widehat{s_m}^1$ in the upper left corner of the m-th image block, the prediction difference is calculated according to the following equation:

$$\begin{aligned}&\widetilde{e_m}^1 \\ &= \left(\widehat{s_m}^1 - \left\lfloor w_v \bullet s_m{}^2 + w_h \bullet s_m{}^4 + w_d \bullet s_m{}^3 \right\rfloor\right) \bmod 256\end{aligned} \tag{16}$$

According to the proof in Section 2.2, it is known that $\widetilde{e_m}^1$ is equal to $\widehat{e_m}^1$.

Step 4. According to the previous embedding process, it can be seen that the secret information can be extracted in $Z_n = [T_n - 2\beta - 1, T_n]$ and $Z_p = [T_p, T_p + 2\beta + 1]$.

If $\widetilde{e_m}^1 \in Z_p$, then

$$\widetilde{w}(l) = \begin{cases} 0 & if \bmod\left(\widetilde{e_m}^1 - T_p, 2\right) = 0 \\ 1 & if \bmod\left(\widetilde{e_m}^1 - T_p, 2\right) = 1 \end{cases} \tag{17}$$

If $\widetilde{e_m}^1 \in Z_n$, then

$$\widetilde{w}(l) = \begin{cases} 0 & if \bmod\left(T_n - \widetilde{e_m}^1, 2\right) = 0 \\ 1 & if \bmod\left(T_n - \widetilde{e_m}^1, 2\right) = 1 \end{cases} \tag{18}$$

Step 5. The extracted bits can be further decrypted by using the data-hiding key K_{em}. Thus the original secret data are obtained.

Since the whole process is entirely operated in encrypted domain, it effectively avoids the leakage of original content. After obtaining the secret data from the marked encrypted image, the prediction difference value can be further restored as follows:

$$e_m{}^1 = \begin{cases} \widetilde{e_m}^1 - \left\lceil \dfrac{\widetilde{e_m}^1 - T_p}{2} \right\rceil & \widetilde{e_m}^1 \in Z_p \\ \widetilde{e_m}^1 - (\beta + 1) & \widetilde{e_m}^1 > T_p + 2\beta + 1 \\ \widetilde{e_m}^1 + (\beta + 1) & \widetilde{e_m}^1 < T_n - 2\beta - 1 \\ \widetilde{e_m}^1 + \left\lceil \dfrac{T_n - \widetilde{e_m}^1}{2} \right\rceil & \widetilde{e_m}^1 \in Z_n \end{cases} \tag{19}$$

It should be noted that the boundary difference can be restored according to the location map L_2. The encrypted pixel values $s_m{}^1$ can be obtained as follows:

$$s_m{}^1 = \left(\widehat{s_m}^1 + e_m{}^1\right) \bmod 256 \tag{20}$$

Thus, the encrypted image without the hidden data, i.e., $S = \{s(i, j) \mid s(i, j) \in [0, 255]\}$, is obtained. With the encryption key K_{en}, the original cover image can be accurately restored by performing the decryption operation as in equation (3). As each recovery step is reversible, the final decrypted image is exactly the same as the original one.

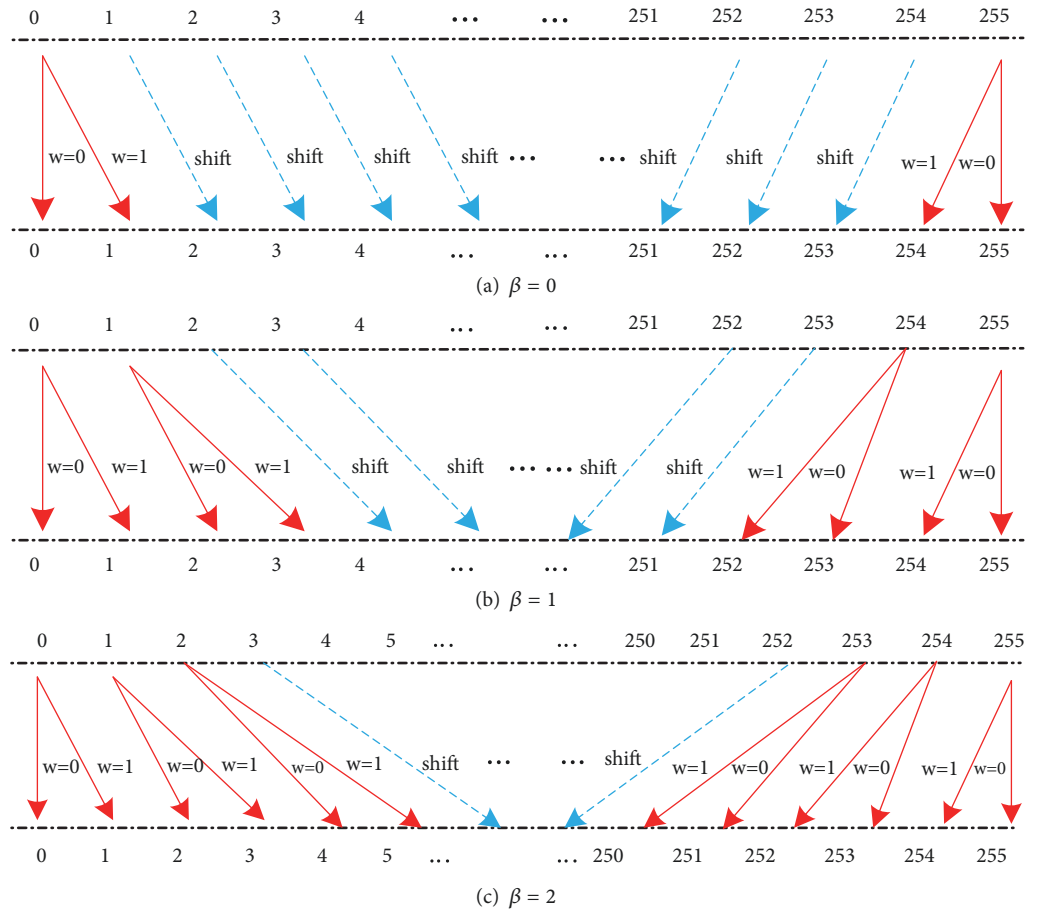


FIGURE 5: Illustration of difference histogram modification.

(2) Scheme II: Data Extraction in the Decrypted Domain. In Scheme I, both data embedding and extraction are performed in the encrypted domain. However, in some scenarios, users want to decrypt the image first and then extract the hidden data from the decrypted image when it is needed. For example, after the image being decrypted, the recipient also hopes to track the source of the image. Thus, Scheme II is introduced to perform data extraction after image decryption. The detailed process of decryption and data extraction is comprised from the following steps.

Step 1. With the encrypted image containing secret information and the encryption key K_{en}, image decryption can be accomplished according to the following equation:

$$\widetilde{\mathrm{X}} = \left(\hat{s}\left(i,j\right) - r\left(i,j\right)\right) \bmod 256 = \tilde{x}\left(i,j\right) \tag{21}$$

No visible distortions can be observed in the marked decrypted images, as will be demonstrated in later experimental results.

Step 2. Divide the marked and decrypted image $\widetilde{\mathrm{X}}$ into nonoverlapping 2x2 blocks, which is exactly the same as in Section 2.1.

Step 3. According to the location map L_1, if the current block is an embeddable block, go to Step 4. If it is a non-embeddable block, the next block is taken as the current block and proceed to Step 3.

Step 4. Calculate the prediction difference between the basic pixel and the remaining pixels in each 2x2 block. For simplicity, the pixel in the upper left corner is still taken as an example.

For the decrypted pixel value $\widehat{x_m}^1$ in the upper left corner of the m-th image block, the prediction difference is calculated according to the following equation:

$$\overline{e_m}^1 = \left(\widehat{x_m}^1 - \left\lfloor w_v \bullet {x_m}^2 + w_h \bullet {x_m}^4 + w_d \bullet {x_m}^3 \right\rfloor\right) \cdot \bmod 256 \tag{22}$$

According to the proof in Section 2.2, it is known that $\overline{e_m}^1$ is equal to $\widehat{e_m}^1$.

Step 5. The hidden data $\widetilde{w}(l)$can be extracted in a manner similar to equation (17) and equation (18). That is, it is only necessary to replace $\widetilde{e_m}^1$ in equation (17) and equation (18) with $\overline{e_m}^1$.

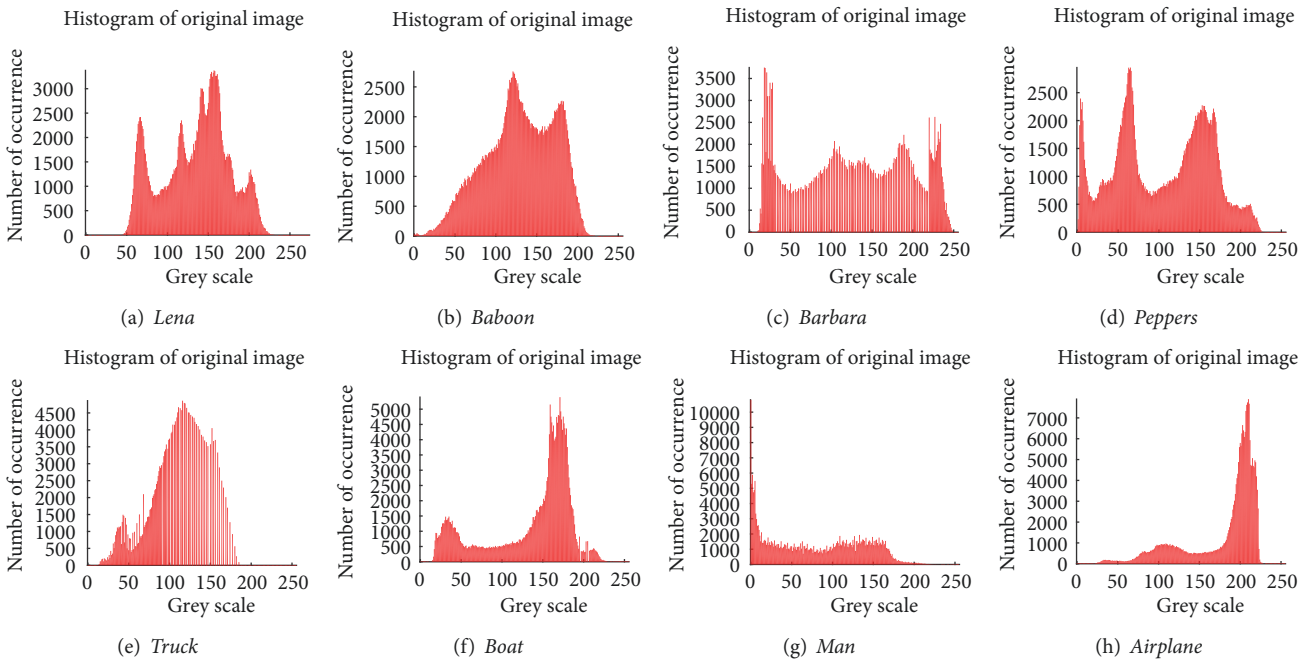


Figure 6: Histogram of the original image.

Step 6. The image difference can also be restored in the same way as in equation (19). The original pixel values x_m^1 can be obtained as follows:

$$x_m^1 = \left(\widetilde{x_m}^1 + e_m^1\right) \bmod 256 \tag{23}$$

Therefore, the original image, i.e., $X = \{x(i, j)\}$, is successfully recovered. Due to the reversibility, the original image and secret data can be completely restored without any error.

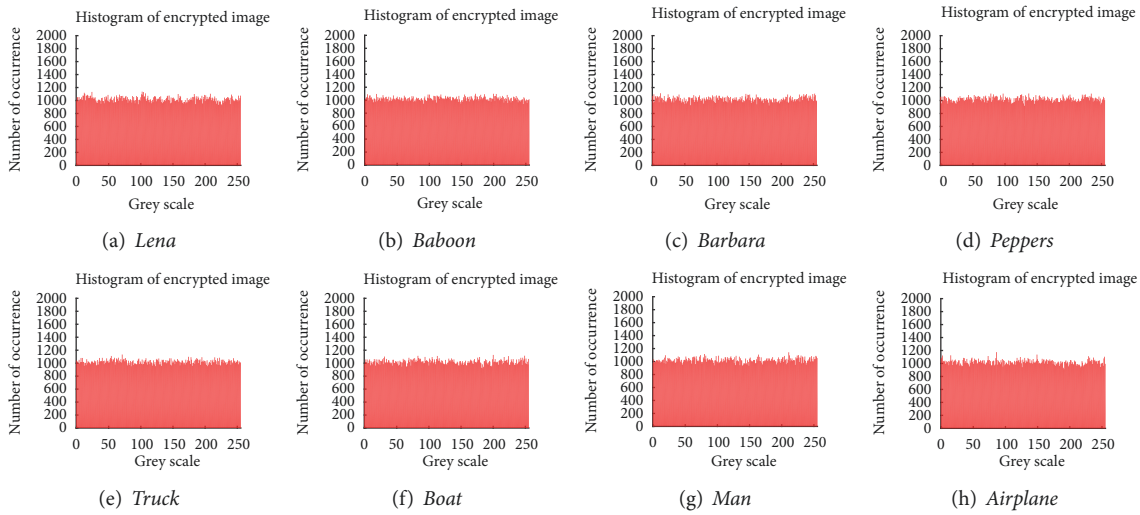
3. Experimental Results and Analysis

In this section, the experimental results obtained by applying our method will be present. Eight standard images of size 512 × 512×8, i.e., *Aerial, Barbara*, *Lena*, *Lighthouse*, *Tank, Truck*, *Zelda*, *and boats* [35], are used to compare our proposed method with the related state-of-the-art works. In addition, 80 images selected from a popular gray-scale image database [36] are also used to further demonstrate the effectiveness of our method. The secret data is a binary sequence generated by pseudo-random number generator. For data hiding in encrypted images, we have to measure different performances which are the scrambling effect, the payload (i.e., embedding rate), and the reconstructed image quality.

3.1. Scrambling Effect and Security Analysis. In RDH-EI, unauthorized user is not allowed to access the original image and secret data. Thus both the original image and secret data should be protected. To protect the secret data, a stream cipher is applied to change the bits. Without the data hiding key K_{em}, it is extremely difficult to reveal the secret data. And a pseudo-random matrix generated by PRNG is used to encrypt image. A statistical analysis of histogram is employed to verify the security level. Generally, histogram of an image demonstrates the distribution of the pixels based on the intensity values. Figure 6 illustrates the histograms of the original image. After encryption, the corresponding histograms are shown in Figure 7. It can be observed that the histograms of the encrypted image obtained with our approach are uniformly distributed in comparison with the original image. It is impossible to exploit them to obtain information about the original content of the image. In addition, the histogram statistics of the original images and the corresponding encrypted images are also given in Table 1.

Perceptual security refers to the encrypted image being unintelligible. The original images are given in Figure 8, and their corresponding encrypted results are shown in Figure 9. As can be observed, the visual content of the plaintext images has been completely blurred by the proposed encryption scheme. In addition, for standard gray images, i.e., *Lena*, *Baboon*, *Barbara*, *Peppers*, *Truck*, *Boat*, *Man*, *and Airplane*, PSNR (Peak Signal to Noise Ratio) values are 9.53dB, 9.53dB, 7.85dB, 8.45dB, 9.61dB, 8.96dB, 7.56dB, and 8.05dB, respectively. The PSNR values of the encrypted images are relatively low. Obviously, scrambling performance of the described encryption system is more than adequate and the visual security is guaranteed. To more comprehensively validate the proposed method, another 80 images are selected from [36] for testing. The *PSNR* values of 80 encrypted images are shown in Figure 11. In [37], bit plane disordering, block, and pixel scrambling are performed, which may provide a reference to further enhance security for our future work.

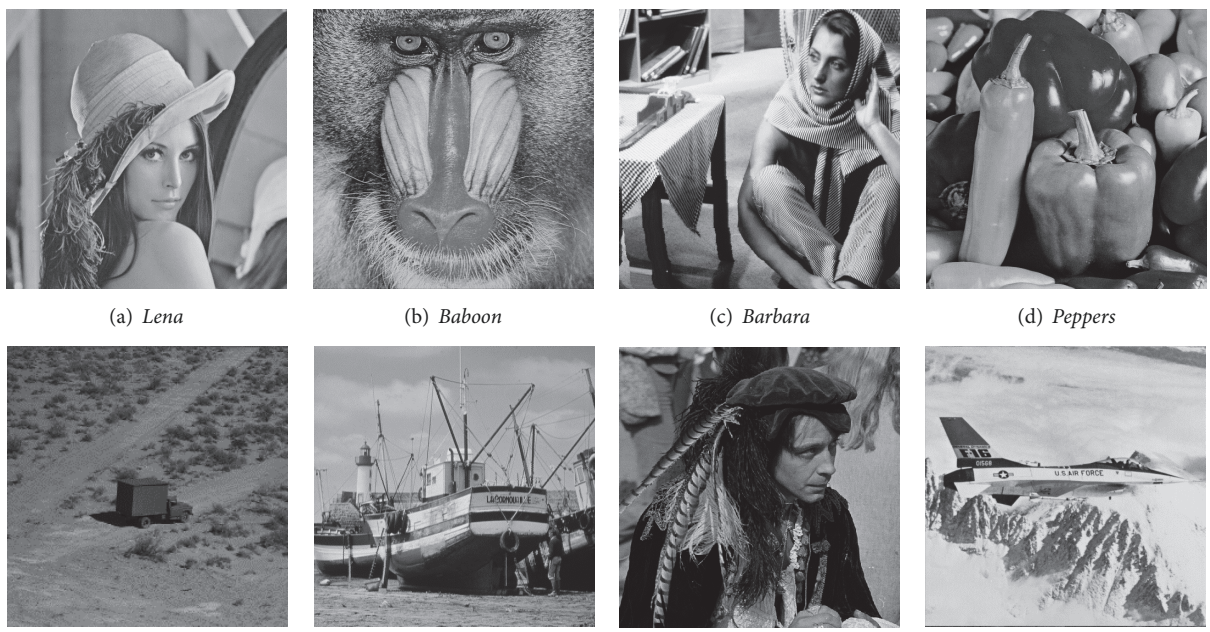
3.2. Visual Quality of Marked and Decrypted Image. When both of the keys K_{en} and $K_{em}$are obtained, the original image



(a) *Lena* (b) *Baboon* (c) *Barbara* (d) *Peppers*

(e) *Truck* (f) *Boat* (g) *Man* (h) *Airplane*

Figure 7: Histogram of the corresponding encrypted image.



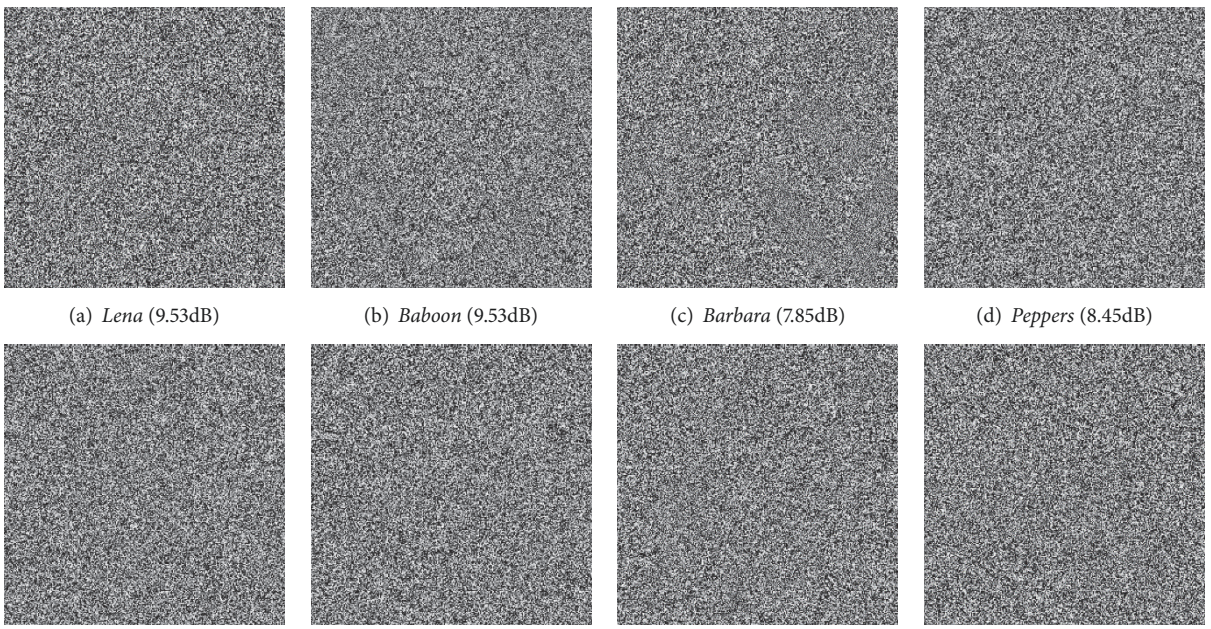
(a) *Lena* (b) *Baboon* (c) *Barbara* (d) *Peppers*

(e) *Truck* (f) *Boat* (g) *Man* (h) *Airplane*

Figure 8: Original images.

Table 1: Comparison of histogram statistics.

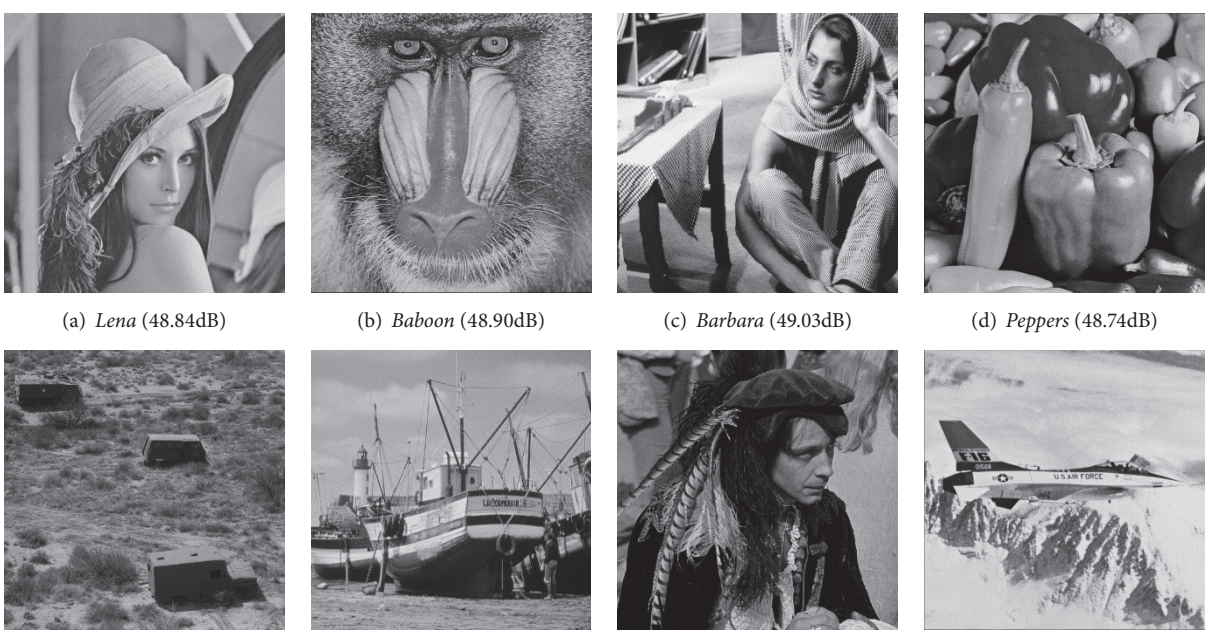
	Original Image			Encrypted Image		
	Mean	Standard Deviation	Median	Mean	Standard Deviation	Median
Lena	134.40	41.51	141	127.05	73.97	127
Baboon	129.15	42.30	130	127.33	73.78	127
Barbara	126.59	72.06	127	127.55	74.10	127
Peppers	104.21	57.40	108	127.55	73.84	127
Boat	136.13	52.32	158	127.51	73.93	127
Truck	106.39	34.97	109	127.42	73.89	127
Airplane	179.20	45.12	200	127.10	73.92	127
Man	77.82	58.25	74	127.80	73.99	127



(a) *Lena* (9.53dB) (b) *Baboon* (9.53dB) (c) *Barbara* (7.85dB) (d) *Peppers* (8.45dB)

(e) *Truck* (9.61dB) (f) *Boat* (8.96dB) (g) *Man* (7.56dB) (h) *Airplane* (8.05dB)

Figure 9: The corresponding encrypted images.



(a) *Lena* (48.84dB) (b) *Baboon* (48.90dB) (c) *Barbara* (49.03dB) (d) *Peppers* (48.74dB)

(e) *Truck* (48.69dB) (f) *Boat* (48.83dB) (g) *Man* (49.12dB) (h) *Airplane* (49.00dB)

Figure 10: Decrypted images containing the hidden data.

can be losslessly recovered by decrypting and extracting the hidden information, as indicated by a *PSNR* which tends to$+\infty$. Equality between the original images and restored images has proved the reversibility of the proposed scheme. Note that the hidden message is always extracted without any error. In some scenarios, the authorized user can decrypt the marked encrypted image to get an approximated original image. Therefore, the visual quality of the decrypted image containing the hidden data is also expected to be equivalent or very close to that of the original image. Since each pixel is altered at most by $\beta + 1$, the introduced distortion will not be perceptible when β is small. To verify this, the original images and their corresponding decrypted versions containing the hidden data are shown in Figures 8 and 10, respectively. Just as shown, the recovery version is very identical to the original image visually.

Table 2: Test results for eight test images.

Images	Embedding rate			PSNR		
	$\beta = 0$	$\beta = 1$	$\beta = 2$	$\beta = 0$	$\beta = 1$	$\beta = 2$
Lena	0.2222	0.4036	0.5407	48.8411	43.6649	40.9351
Baboon	0.0629	0.1223	0.1779	48.9038	43.1374	39.8523
Barbara	0.1434	0.2571	0.3426	49.0294	43.5576	40.5109
Peppers	0.1639	0.3034	0.4118	48.7383	43.3371	40.3917
Boat	0.1892	0.3447	0.4630	48.8325	43.5223	40.6503
Truck	0.0943	0.1810	0.2600	48.6871	43.0213	39.8344
Airplane	0.2709	0.4691	0.5935	48.9975	44.0302	41.4274
Man	0.1103	0.1951	0.2588	49.1221	43.6780	40.6730

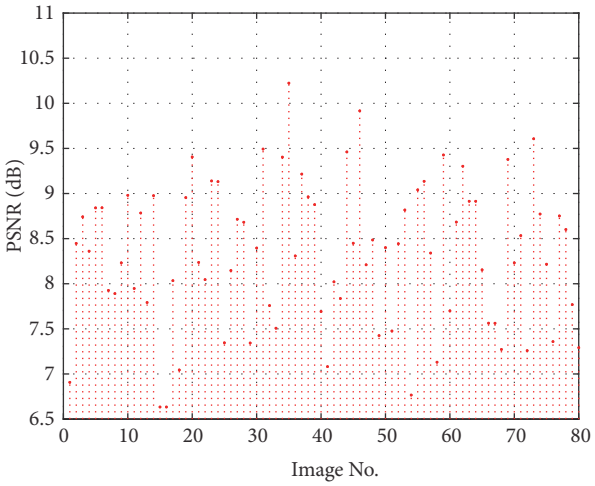


Figure 11: *PSNR* values of 80 encrypted images.

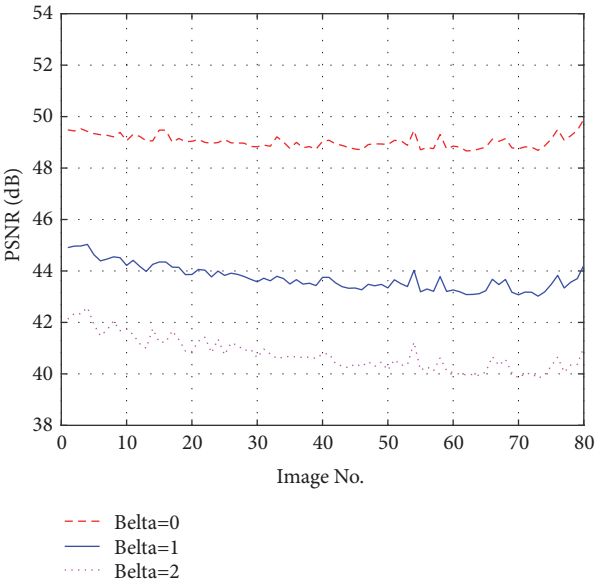


Figure 12: PSNR values of 80 decrypted images.

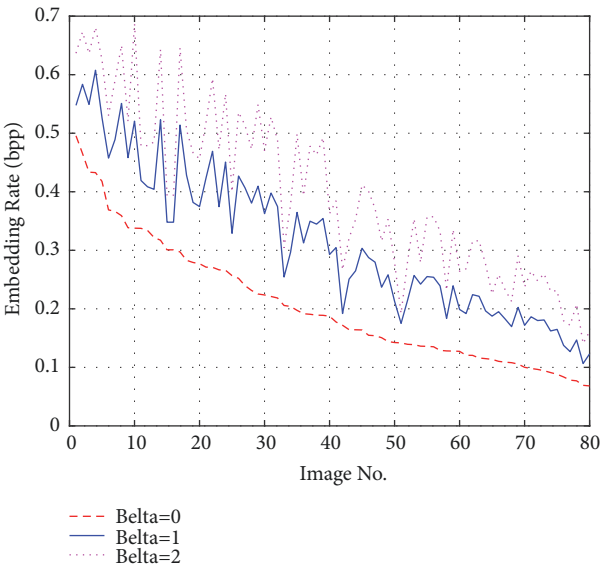


Figure 13: Embedding rates of 80 test images.

To quantitatively evaluate the reconstructed image quality in comparison to the original one, the PSNR values with different embedding rate are given in Table 2. When $\beta = 0$, the *PSNR* values are all above 48 dB. Even when $\beta = 2$, the *PSNR* values are all around 40 dB. In addition, the *PSNR* values for the other 80 images [36] are also given in Figure 12, which have similar results. These test results indicate that it is almost impossible to detect the degradation in image quality caused by data hiding.

3.3. Embedding Capacity. In our experiments, the embedding capacity is measured in bit per pixel (bpp), which is expected to be as large as possible in order to embed the maximum amount of information. Obviously, the embedding capacity in each image is determined by the number of prediction errors within $Z_0 = [T_n - \beta, T_n] \cup [T_p, T_p + \beta]$. As the side information is extremely small with respect to the capacity, the side bits are not counted in the following experimental results. For eight standard gray images, the embedding rates are shown in Table 2. It can be observed that the embedding capacity of the proposed scheme depends strongly on the characteristics of

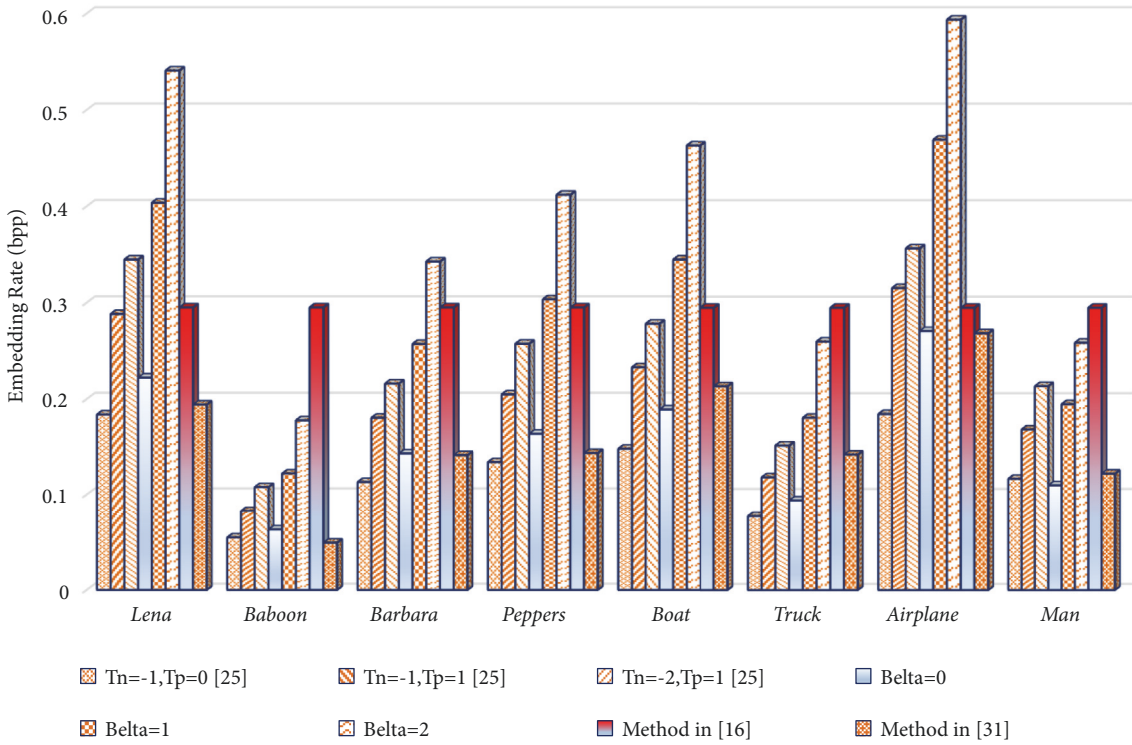


Figure 14: The comparison results of embedding capacity.

the original cover image, as each image has a different number of prediction errors associated with the embedding process. As expected, images with less texture in the original version (e.g., *Lena* and *Airplane*) have higher prediction accuracy and thus can contribute higher number of differences associated with the peak point. Therefore, they can be embedded with more data, achieving a larger embedding rate. On the other hand, images with higher spatial activity (e.g., *Baboon* and *Truck*) achieve lower embedding rate.

As can be seen from equation (14), the embedding capacity is related to the parameter β which can be used for adjusting the embedding capacity flexibly. When a low capacity is needed, e.g., for content authentication purpose, we can narrow the embedding range, e.g., $\beta = 0$. In this way, a higher quality of stego-image can be achieved, as depicted in Figure 12. As the parameter β increases, it means that the more prediction errors can be exploited for embedding. As a result, the embedding capacity will be higher. But on the other hand, more shifting and embedding operations will make more changes to the pixel values, which will lead to lower PSNR.

Table 2 illustrates the embedding capacity for three cases, i.e., $\beta = 0$, $\beta = 1$, and $\beta = 2$, which demonstrates the effectiveness of β on improving the embedding capacity. It is obvious that there is no perfect solution to achieve high payload and low distortion simultaneously. The more flexible capacity control is achieved in our framework, which is helpful to make a tradeoff between the capacity and the visual quality according to the different practical requirements. Besides eight standard images, the embedding rates of other 80 test images [36] are plotted in Figure 13.

3.4. Comparison and Discussion. As mentioned in Section 1, all methods in [12–14] may introduce some errors on data extraction and/or image recovery, while complete reversibility and error-free extraction can be achieved in the proposed method. For methods in [23–25], error-free data extraction and image lossless recovery can be obtained. But histogram shifting should be done prior to encrypting the image. On the contrary, in our method, the image is directly encrypted, which is more reasonable. In addition, several comparisons are made between our proposed method and several previous methods [16, 25, 31] in terms of embedding rate. To do this, eight standard images are taken as examples, and the comparison of the embedding capacity is shown in Figure 14. First of all, it can be seen that our method has a larger payload than the others. In fact, the maximal payload for these methods, obtained by Xu *et al.* [25] is 0.3565bpp. Specifically, the proposed method reaches a maximal payload of 0.5935bpp, which is far higher than what the other compared algorithms can achieve. When we examine the reconstructed image quality, our method can perfectly reconstruct the original image (PSNR$\longrightarrow +\infty$) using both the encryption key and the data hiding key.

4. Conclusions and Future Work

In this paper, an efficient framework for RDH-EI is presented. A specific modulo operation is utilized to encrypt the image, which can preserve some correlation between the neighboring pixels. With the preserved correlation, the data-hider can embed the additional data into the encrypted

image by using difference histogram modification. Since the embedding process is done on encrypted data, our scheme preserves the confidentiality of content. Data extraction is separable from image decryption; i.e., the additional data can be extracted either in the encrypted domain or in the decrypted domain. Experimental results show that the visual quality of marked decrypted image is very high and that the achieved payload is enough to embed some additional data. On the other hand, real reversibility can be achieved, which means that the secret data and original image can be restored without any error. Future works will focus on determining the optimal modification on the histogram to achieve the best rate-distortion performance.

Data Availability

The [.xlsx] data and MATLAB source code used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61771270), Zhejiang Provincial Natural Science Foundation of China (LY17F020013), and Ningbo Natural Science Foundation (2018A610054).

References

[1] Q. Wang, W. Zeng, and J. Tian, "A compressive sensing based secure watermark detection and privacy preserving storage framework," *IEEE Transactions on Image Processing*, vol. 23, no. 3, pp. 1317–1328, 2014.

[2] B. Zhao, W. D. Kou, H. Li, L. Dang, and J. Zhang, "Effective watermarking scheme in the encrypted domain for buyer-seller watermarking protocol," *Information Sciences*, vol. 180, no. 23, pp. 4672–4684, 2010.

[3] A. V. Subramanyam, S. Emmanuel, and M. S. Kankanhalli, "Robust watermarking of compressed and encrypted JPEG2000 images," *IEEE Transactions on Multimedia*, vol. 14, no. 3, pp. 703–716, 2012.

[4] T. Bianchi and A. Piva, "Secure watermarking for multimedia content protection: a review of its benefits and open issues," *IEEE Signal Processing Magazine*, vol. 30, no. 2, pp. 87–96, 2013.

[5] J. Guo, P. Zheng, and J. Huang, "Secure watermarking scheme against watermark attacks in the encrypted domain," *Journal of Visual Communication and Image Representation*, vol. 30, pp. 125–135, 2015.

[6] H. Liu, D. Xiao, R. Zhang, Y. Zhang, and S. Bai, "Robust and hierarchical watermarking of encrypted images based on Compressive Sensing," *Signal Processing: Image Communication*, vol. 45, pp. 41–51, 2016.

[7] D. Xu, R. Wang, and Y. Q. Shi, "Data hiding in encrypted H.264/AVC video streams by codeword substitution," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 4, pp. 596–606, 2014.

[8] D. Xu, R. Wang, and Y. Q. Shi, "An improved scheme for data hiding in encrypted H.264/AVC videos," *Journal of Visual Communication and Image Representation*, vol. 36, pp. 229–242, 2015.

[9] D. Xu, R. Wang, and Y. Zhu, "Tunable data hiding in partially encrypted H.264/AVC videos," *Journal of Visual Communication and Image Representation*, vol. 45, pp. 34–45, 2017.

[10] P. Singh, B. Raman, and N. Agarwal, "Towards encrypted video tampering detection and localization based on POB number system over cloud," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 9, pp. 2116–2130, 2018.

[11] Y.-Q. Shi, X. Li, X. Zhang, H.-T. Wu, and B. Ma, "Reversible data hiding: Advances in the past two decades," *IEEE Access*, vol. 4, pp. 3210–3237, 2016.

[12] X. P. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Processing Letters*, vol. 18, no. 4, pp. 255–258, 2011.

[13] X. P. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 826–832, 2012.

[14] W. Hong, T.-S. Chen, and H.-Y. Wu, "An improved reversible data hiding in encrypted images using side match," *IEEE Signal Processing Letters*, vol. 19, no. 4, pp. 199–202, 2012.

[15] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High capacity reversible data hiding in encrypted images by patch-level sparse representation," *IEEE Transactions on Cybernetics*, vol. 46, no. 5, pp. 1132–1143, 2016.

[16] Z. Qian and X. Zhang, "Reversible data hiding in encrypted images with distributed source encoding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 4, pp. 636–646, 2016.

[17] F. Huang, J. Huang, and Y.-Q. Shi, "New framework for reversible data hiding in encrypted domain," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 12, pp. 2777–2789, 2016.

[18] J. Zhou, W. Sun, L. Dong, X. Liu, O. C. Au, and Y. Y. Tang, "Secure reversible image data hiding over encrypted domain via key modulation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 3, pp. 441–452, 2016.

[19] P. Puteaux and W. Puech, "An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1670–1681, 2018.

[20] C. Qin, Z. He, X. Luo, and J. Dong, "Reversible data hiding in encrypted image with separable capability and high embedding capacity," *Information Sciences*, vol. 465, pp. 285–304, 2018.

[21] S. Yi and Y. Zhou, "Separable and reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Transactions on Multimedia*, vol. 21, no. 1, pp. 51–64, 2019.

[22] H. L. Ge, Y. Chen, Z. X. Qian, and J. J. Wang, "A high capacity multi-level approach for reversible data hiding in encrypted images," in *IEEE Transactions on Circuits and Systems for Video Technology*, 2018.

[23] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 3, pp. 553–562, 2013.

[24] W. M. Zhang, K. D. Ma, and N. H. Yu, "Reversibility improved data hiding in encrypted images," *Signal Processing*, vol. 94, no. 1, pp. 118–127, 2014.

[25] D. Xu and R. Wang, "Separable and error-free reversible data hiding in encrypted images," *Signal Processing*, vol. 123, pp. 9–21, 2016.

[26] Y.-C. Chen, C.-W. Shiu, and G. Horng, "Encrypted signal-based reversible data hiding with public key cryptosystem," *Journal of Visual Communication and Image Representation*, vol. 25, no. 5, pp. 1164–1170, 2014.

[27] C.-W. Shiu, Y.-C. Chen, and W. Hong, "Encrypted image-based reversible data hiding with public key cryptography from difference expansion," *Signal Processing: Image Communication*, vol. 39, pp. 226–233, 2015.

[28] X. Zhang, J. Long, Z. Wang, and H. Cheng, "Lossless and reversible data hiding in encrypted images with public-key cryptography," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1622–1631, 2016.

[29] H.-T. Wu, Y.-M. Cheung, and J. Huang, "Reversible data hiding in Paillier cryptosystem," *Journal of Visual Communication and Image Representation*, vol. 40, pp. 765–771, 2016.

[30] S. Xiang and X. Luo, "Reversible data hiding in homomorphic encrypted domain by mirroring ciphertext group," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, pp. 3099–3110, 2017.

[31] D. W. Xu, K. Chen, R. D. Wang, and S. B Su, "Separable reversible data hiding in encrypted images based on two-dimensional histogram modification," *Security and Communication Networks*, vol. 2018, Article ID 1734961, p. 14, 2018.

[32] K. Chen and D. Xu, "An efficient reversible data hiding scheme for encrypted images," *International Journal of Digital Crime and Forensics*, vol. 10, no. 2, pp. 1–22, 2018.

[33] D. Xiao, Y. Xiang, H. Zheng, and Y. Wang, "Separable reversible data hiding in encrypted image based on pixel value ordering and additive homomorphism," *Journal of Visual Communication and Image Representation*, vol. 45, pp. 1–10, 2017.

[34] L. Luo, Z. Chen, M. Chen, X. Zeng, and Z. Xiong, "Reversible image watermarking using interpolation technique," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 1, pp. 187–193, 2010.

[35] "Test Images," http://www.hlevkin.com/TestImages/.

[36] "Test Images," http://decsai.ugr.es/cvg/dbimagenes/g512.php.

[37] C. Qin, X. Qian, W. Hong, and X. Zhang, "An efficient coding scheme for reversible data hiding in encrypted image with redundancy transfer," *Information Sciences*, vol. 487, pp. 176–192, 2019.