Received September 16, 2019, accepted October 4, 2019, date of publication October 11, 2019, date of current version October 25, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2946929

Reversible Data Hiding in Encrypted Images Using POB Number System

HUA REN, SHAOZHANG NIU, AND XINYI WANG
Beijing Key Laboratory of Intelligent Telecommunication Software and Multimedia, School of Computer, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Shaozhang Niu (szniu@bupt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61370195, and in part by the Joint Funds of the National Natural Science Foundation of China under Grant U1536121.

ABSTRACT Prediction-error expansion (PEE) methods usually involve the shifting process between peak points and zero points in order to vacate room for embedding. This occurrence will inevitably impose a troublesome because overflow/underflow issue after data embedding may occur, in turn, it dramatically increases the amount of auxiliary information to be transmitted or map information to be embedded. Besides, PEE-based reversible data hiding in encrypted images (RDHEI) methods usually adopt a block-size encryption strategy, but the correlation reservation of intra-block for subsequently embedding may leave opportunities for potential attacks. To deal with these issues, we proposed a novel RDHEI method with the functionality of embedding synchronized with re-encryption, which is well carried out by a data hider using a permutation ordered binary (POB) number system. Unlike conventional PEE-based RDHEI techniques, this scheme takes advantage of two peak points for embedding, but thoroughly abandons the former idea of shifting process. Experimental results verify the superiority of the proposed method, which not only may extend the applications of PEE-based RDHEI schemes substantially, but also vastly extend POB number application scenarios in practice.

INDEX TERMS Reversible data hiding, encrypted image, prediction-error expansion, POB number system.

I. INTRODUCTION

Data hiding is a technique of embedding secret messages into given cover images by subtly modifying several selected pixels. It has been widely applied into covert communication, tamper detection, copyright protection, and etc. In some distortion-unacceptable scenarios, such as medical imagery, low forensics and military imagery, both cover image and secret messages are required to be recovered perfectly, and data hiding should be carried out in a reversible and invertible manner, which is called reversible data hiding (RDH) [1]–[3]. The existing embedding strategies of reported RDH methods are usually partitioned into three primary forms: lossless compression [4], difference expansion (DE) [5] and histogram shifting (HS) [6]. To substantially improve embedding capacity and stego-image quality, DE and HS were then evolved into a series of prediction strategies, e.g., prediction-error expansion (PEE) [7], [8], in which instead of directly referring to the original image as the cover data, the prediction error was constructed as the cover data for hiding.

Encryption is a well-known effective method for protecting the confidentiality of image owners, which aims to mask the contents of the original images, so that attackers or unauthorized entities cannot identify the contents of protected ones [9]–[11]. In some cloud-based scenarios, image owners may hold some devices that have limited storage and computation power, such as mobile devices, so they tend to seek for helps from cloud in order to conduct extra operations. However, with the aim of privacy protection, it is preferable to encrypt data before they uploaded them to the cloud. From the perspective of cloud, RDHEI techniques should be equipped to conduct certain relative operations [12].

Although it is difficult to embed secret reversibly into encrypted images using common data hiding algorithms because encryption makes the entropy of an image to be enlarged, RDHEI techniques still have received increasing concerns in image processing fields [13]–[15]. According to when the embedding space was vacated, the embedding mechanisms of the existing RDHEI schemes can

The associate editor coordinating the review of this manuscript and approving it for publication was Donghyun Kim.

be divided into two main types: reserving room before encryption (RRBE) [16] and vacating room after encryption (VRAE) [17]–[19]. The general framework of RRBE is to create embedding room in the plaintext domain, by which the content owner is expected to perform an extra preprocessing before encryption. Although a higher hiding capacity can be ensured, this scheme may be impractical, because the extra preprocessing operation will increase several extra computation burdens for the content owner. For the framework of VRAE, according to the domain in which the secret messages can be extracted, VRAE methods can be further grouped into three basic categories: data extraction in plaintext domain, in the cipher domain and in both domains. This flowchart usually includes three parts: a content owner encrypts the original signal, a data hider embeds secret messages by modifying some bits of the encrypted data, and a receiver carries out data extraction and image restoration with relative secret keys. VRAE methods are relatively practical for various scenarios; however, they offer a limited capacity due to the fact that augmented information entropy in encrypted images makes it difficult to hide secret messages into the encrypted ones. In order to balance hiding capacity and visual quality, some techniques, such as homomorphic encryption [17] and public cryptosystem [20]–[24], were also involved to achieve RDHEI. In [17], a content owner categorized an original image into a series of non-overlapping blocks, and the pixels in each block were associated with the same encryption key. Based on this, a data hider embedded secret messages into specific pixels under homomorphism encryption. This method implements greater performance in terms of visual quality and embedding capacity, but the encryption security is unsatisfactory, since the local correlations in each encrypted block before data embedding must be reserved for vacating spare room. Almost all of the current block-based RDHEI techniques perform encryption and data hiding in a separable manner, however, encryption designs before data embedding might be not fully satisfactory, and thus it is necessary and crucial to study a manner that can perform data embedding synchronized with re-encryption, in order to remedy loopholes from possible insecure cryptosystems.

Recently, several interesting researches have been conducted on secret sharing schemes based on a permutation ordered binary (POB) number system [25]–[28]. This concept is initially proposed by Sreekumar and Sundar [25] to construct a novel n out of n secret sharing. The decimal value in each share corresponds to a POB value. When needed the original secrets, the POB values in each share are converted back into corresponding decimal system, and these decimal values of all shares are then combined to reconstruct the original secret. Based on this concept, various application scenarios, e.g., image tampering detection and localization [26], video tampering detection and localization [27] and cloud-based secure data deduplication [28], have been presented and further discussed. These schemes work well in secret sharing systems in terms of tamper detection or recovery. Thus, these excellent POB-number properties are utilized here.

Enlightened by these concerns mentioned above, we propose a novel RDHEI scheme using a POB number system, which may be viewed as an alternative means of shifting room for embedding or a lossless compression embedding. In our scheme, the content owner partitions the original image into a series of equal-size non-overlapping blocks, he/she encrypts these non-overlapping blocks by a stream cipher, and the pixels in each block are associated with the same encryption key. Based on the same block-division strategy, a data hider computes the differences between the first pixel (unchanged before and after embedding) and the other pixels in each block. The functionality, compression synchronized with re-encryption, of POB number is then leveraged to vacate redundancy room for data embedding. This will assist the data hider to embed secret messages into differences "0" and "1" (two peak points) without any shifting process by using POB-based schemes; in other words, other differences (except "0" and "1") during the embedding phase remain the same. For the data extraction and image restoration process, the embedded secret messages can be extracted without any error through an accurate calculation of the inverse POB number system, and the image restoration can be implemented perfectly. The contributions of this paper can be summarized as the following aspects:

1) Enhancement of cryptosystem security: the data embedding of our method can re-encrypt the current processed pixel automatically, which will destroy the local correlation reserved by cryptosystem design and make the marked encrypted images more unintelligible.
2) Successful elimination of overflow/underflow issue: the secret messages are embedded into two peak points (differences "0" and "1"), but do not require the shifting process between peak point and zero point; thus, it is unnecessary to introduce auxiliary information to be transmitted or map information to be embedded (used to record the boundary pixels during shifting).
3) No loss in data extraction and image recovery: data extraction is free of any error, and image restoration is achieved perfectly.
4) Low computation burden: two parameters in logistic map are utilized to be as the keys for encryption, and a POB number system is used for the lossless embedding.

II. PRELIMINARY

A. POB NUMBER SYSTEM

POB number system has already received ever-increasing concerns in the signal processing fields since Sreekumar and Sundar primitively presented the concept [25], which is especially suitable for secret sharing based application scenarios. The system is a general number system denoted by $POB(n, r)$, where n and r are non-negative integral parameters such that $n \geq r$. In this number system, all of the

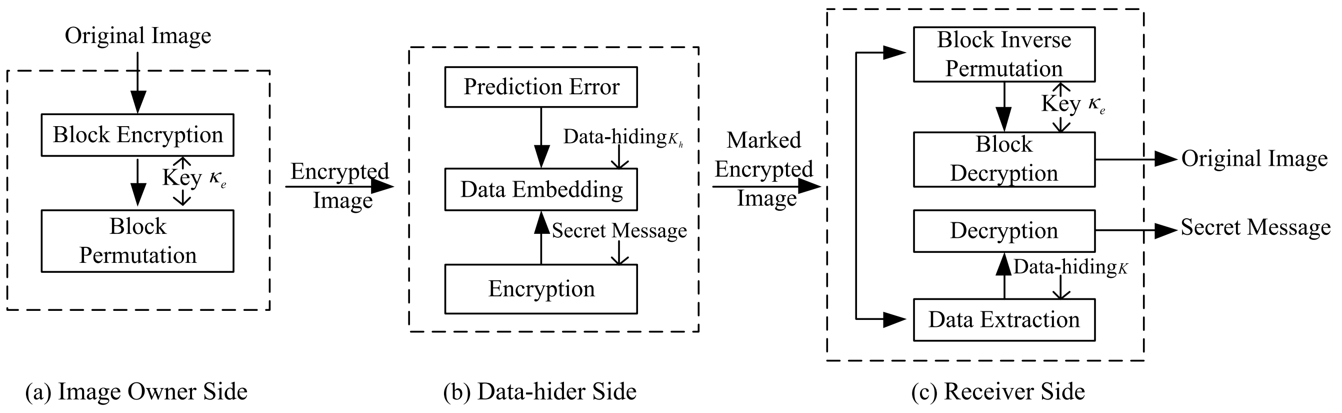

FIGURE 1. The sketch of the proposed RDHEI scheme. (a) Image encryption for the image owner side. (b) data hiding for the data hider. (c) data extraction and image recovery for the receiver.

integer values are in the range of $0, 1, \ldots, \binom{n}{r} - 1$ and can be represented as a binary string known as POB number B, in which $B = b_{n-1}b_{n-2}\ldots b_0$ is of size n having exactly r $1's$. For a given POB number system B, each digit of this number is dependent on its position value j. The POB value $V(B)$ can be generated below:

$$V(B) = \sum_{j=0}^{n-1} b_j \binom{j}{p_j} \tag{1}$$

where $p_j = \sum_{i=0}^{j} b_j$. Let us use the suffix 'p' to denote any given POB number $B = b_{n-1}b_{n-2}\cdots b_0$, e.g. $010101011p$, a specific POB number in the $POB(9, r)$ number system. It needs to be emphasized that the POB value for a given POB number $010101011p$ is 27; however, the binary representation of such a string ($010101011p$) will have a decimal value 171. The result has a vital impact on these hiding-based studies, since it ensures an invertible and reversible data embedding when carried out a lossless compression and embedding synchronized with re-encryption. The detailed properties concerning a given POB number (POB value) to its POB value (POB number) will be discussed as follows.

B. PROPERTIES OF POB NUMBER SYSTEM

For any given $POB(n, r)$ number system, it must satisfy the following properties:

1) Given two positive integral values n and r, the two values must satisfy the condition $n \geq r$; also, there will be exactly $\binom{n}{r}$ members in the $POB(n, r)$ number system.
2) The POB representation is unique; in other words, for a given $POB(n, r)$ number system, the POB number corresponding to a known POB value is one-to-one mapped [3].
3) For the $POB(n, r)$ number system, the decimal equivalents corresponding to POB values vary from 0 to $\binom{n}{r}$.

It is in place to mention that if one embeds 1-bit secret message into each pixel of a given cover image, the binary sequence of all pixels can be converted into 9-bits POB number. These POB number can be further converted to POB values having a maximum value of 125; in other words, the maximum POB value will be 125 if a $POB(9, r)$ system is utilized. As such, if one embeds 2-bits secret messages into a given cover image, there will exist a maximum POB value 251 for the known $POB(10, r)$ number system. In the aspects of data extraction and reconstruction, the POB values are converted into a binary POB number string with 10 bits according to the reserved value r. The last 2 least significant bits (LSBs) of the POB number are taken out as the secret bits, and the remaining 8 bits are regarded as the original bits.

III. THE PROPOSED METHOD

In this section, a novel POB-based RDHEI scheme is presented. Similar to the most of existing RDHEI methods, the proposed RDHEI scheme is composed of three components: image owner side, data hider side, and receiver side, which carry out image encryption, data hiding, data extraction and image restoration independently, as shown in Figs. 1(a-c). Detailed assignments for the three components are shown as follows.

An image owner first partitions the original image into a series of non-overlapping blocks, and he/she further encrypts these blocks with a stream cipher key κ_e in the way that the pixels in each block are associated with the same key. After block encryption, all encrypted blocks are permuted with stream cipher key κ_e to provide higher levels of security. Then, a data hider enciphers the secret messages with any existing encryption methods, divides the encrypted image into several non-overlapping blocks in the same way with the image encryption phase, and computes the difference in each block in such a way that the first pixel value remains unchanged, and the other three pixels, in turn, subtract the first pixel value. After that, the secret messages can be embedded into specific pixels (difference "0" and "1") by using a data hiding key K_h. At last, after a receiver obtained the

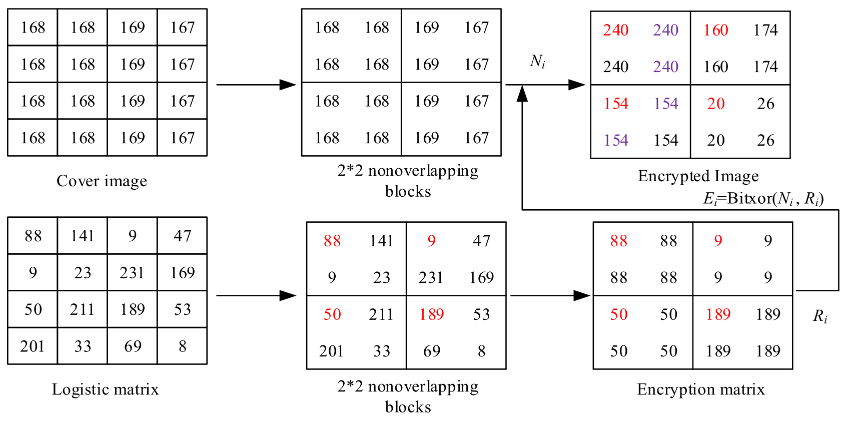

FIGURE 2. **An example illustration of encryption flowchart for local details of Lena.**

marked encrypted image from the data hider, she/he performs a joint data extraction and image decryption operation with her/his holding the encryption key and the data hiding key. Detailed procedures will be presented below.

A. IMAGE ENCRYPTION

An entire process of the image encryption is carried out by the image owner in a block-size manner. The image owner partitions an original image sized $M \times N$ into a series of 2×2 non-overlapping blocks B_m, the total blocks are $L = \lfloor M/2 \rfloor \times \lfloor N/2 \rfloor$, and m falls into $[1, L]$. If the four pixels in arbitrarily one block are represented as $B_m(x, y)$, they can be represented as the type of 8-binary bits, including $b_m(x, y, 0), b_m(x, y, 1), \ldots, b_m(x, y, 7)$. The detailed formula is presented as follows:

$$b_m(x, y, k) = \mathrm{mod}(\left\lfloor \frac{B_m(x, y)}{2^k} \right\rfloor, 2), \quad x, y = 1, 2 \text{ and } k = 0, 1, \ldots 7. \qquad (2)$$

After converting the original image into a series of bit streams, one can perform block encryption in the way that all pixels in the same block are ciphered through the same stream key. The formula is presented as follows.

$$b'_m(x, y, k) = b_m(x, y, k) \oplus \kappa_m(k), \quad x, y = 1, 2 \qquad (3)$$

where $\kappa_m(k)$ is an arbitrary random bit associated with a stream cipher key κ_e, which aims to encrypt the k-th bit plane of the current block B_m, and $b'_m(x, y, k)$ represents the corresponding encrypted resultant by XORing $b_m(x, y, k)$ and $\kappa_m(k)$. After encrypting all blocks, the image owner collects all of the encrypted binary bits $b'_m(x, y, k)$ to obtain the encrypted image.

$$B'_m(x, y) = \sum_{k=0}^{7} b'_m(x, y, k) \times 2^k, \quad x, y = 1, 2 \qquad (4)$$

Then, a permutation process controlled by the key κ_e will be utilized to enhance the security. Next, for an easy illustration of the encryption process, the first 16 pixels of a standard test image Lena are chosen for testing the encryption resultant. Rather than the use of a bit-level encryption, we present the encryption result in a pixel-level manner instead. The most well-known logistic map is utilized for generating the logistic sequence.

$$X_{n+1} = RX_n(1 - X_n) \qquad (5)$$

With the parameter $R \in (3.5699456, 4]$, the system is in chaotic state. The value of the parameter x_0 is related to a normalized hash function $\Phi^{norm}(\cdot)$ and an original image I, i.e., $x_0 = \Phi^{norm}(I)$. The encryption flowchart is shown in Fig. 2. It is apparent that the first 16 pixels of the original cover image have strong correlations, while the pseudo-random values in the logistic matrix are almost uncorrelated, and thus can be well exploited for generating a block-based pseudo-random sequence. Specifically, both the original cover image and the logistic matrix are partitioned into a series of 2×2 non-overlapping blocks firstly, and the pseudo-random values of 2×2 non-overlapping blocks in the logistic matrix are further processed in the way that the first pseudo value remains unchanged, and assign it to each of the remaining values. This process causes each block in the logistic matrix to have the same pseudo-random value. In the following, a pixel-level XOR operation between blocks, separately from the matrix and the original cover image, is carried out. Note that, although the inter-block permutation is a part of the encryption scheme, this process is not the core to present the encryption validity, thus we omit this and only show the block encryption. For the encryption resultant, one can find that, the correlations of the intra-blocks have been reserved, while these nearly exists no correlation among the inter-blocks. These well-reserved correlations in the intra-blocks are of great significance for the following data embedding.

B. DATA EMBEDDING

Since the received image has been encrypted by a full encryption manner, data embedding must be processed directly in the encrypted domain. To meet the reversibility, a POB number system is adopted here. The difference histograms of the

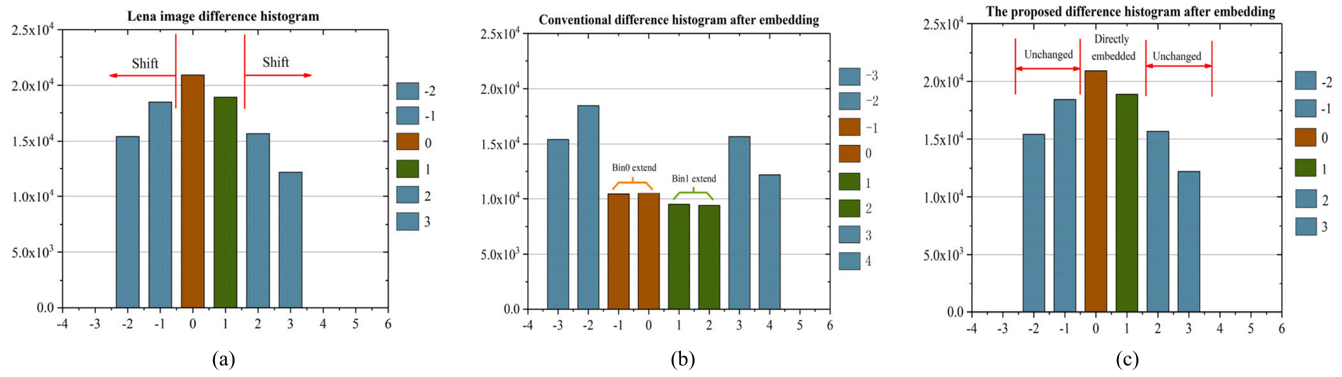

FIGURE 3. The local details of histogram shifting for conventional data embedding process. (a) Original difference histogram for encrypted Lena. (b) Shifted histogram after data embedding. (c) Data embedding of the proposed method.

encrypted images are firstly generated. Then, we embed the secret messages directly in these pixels in difference “0” and “1” by using the POB number system. The details of the data embedding are presented as follows.

1) DIFFERENCE HISTOGRAM GENERATION

When obtained the encrypted images from the image owner, a data hider also divides the encrypted images into a series of 2×2 non-overlapping encrypted blocks B'_m in the same way with the encryption phase. To facilitate description, let us denote the four pixels in one of the encrypted blocks as $B'_m(1, 1)$, $B'_m(1, 2)$, $B'_m(2, 1)$ and $B'_m(2, 2)$ $(m = 1, 2 \ldots L)$. For each block, the differences $d_{m,1}$ $d_{m,2}$ $d_{m,3}$ between the first pixel $B'_m(1, 1)$ and the remaining pixels $B'_m(1, 2)$, $B'_m(2, 1)$, and $B'_m(2, 2)$ are calculated below:

$$d_{m,1} = B'_m(1, 2) - B'_m(1, 1), \tag{6}$$

$$d_{m,2} = B'_m(2, 1) - B'_m(1, 1), \tag{7}$$

$$d_{m,2} = B'_m(2, 2) - B'_m(1, 1), \tag{8}$$

The encryption scheme mentioned above remains the correlations of the intra-blocks to some extent, all pixels in each block B'_m thus are almost kept resembling characteristics. This property makes these differences $d_{m,1}$, $d_{m,2}$ and $d_{m,3}$ equal to or approximately equal to zero, which is essential for the data hider to perform the secret messages embedding. Until now, difference histogram in the encrypted image has been generated.

From the perspective of conventional PEE-based methods, once difference histogram is generated, the next step to be done must be to shift the generated difference histogram so as to vacate room, followed by the procedure of data embedding. In order to explain this process in detail, the local details of histogram shifting for conventional data embedding methods are shown in Figs. 3(a, b), in which these differences are generated from Lena. In Fig. 3(a), the bins marked with light blue need to be shifted one step toward two sides, and the two bins separately marked with orange and green should be responsible for extension in order to carry secret messages. The shifted histogram has been illustrated in Fig. 3(b), where the bins “0” and “-1” are the extension resultant of the bin “0” in Fig. 3(a), and the bins “1” and “2” are the extensions of the original bin “1”.

The total process of these conventional PEE-based methods effectively ensures the reversibility and embedding capacity, whereas it also inevitably leads to overflow/underflow issue. A viable solution to this issue, in the most of PEE-based schemes, is to introduce boundary map to record boundary information; however, how to transmit the introduced boundary map will be one novel troublesome again, which has an irresistible tendency to cause extra expenses and costs. Thus, it is imperative and indispensable to develop further studies to resolve overflow/underflow issue in essence. Enlighten by the superiority of the properties from POB number system, a solution to this issue builds upon these differences, obtained from adjacent encrypted elements. The shifting for room is replaced with the POB number system. Moreover, with the help of POB number system, there is no need to introduce an extra shifting for vacating room process. In the proposed total data embedding process, the light blue bins which are conventionally shifted to vacate space here will remain unchanged, and we embed secret messages directly in specific pixels located in bins “0” and “1”, as shown in Fig. 3(c). The reversibility can be ensured not only due to the properties of POB number system mentioned above, but also the non-negligible characteristic, e.g., lossless compression synchronized with re-encryption.

On the other hand, the reason why we do not select all of the bins in the generated difference histogram to hide secret messages is exactly attributed to the lossless compression synchronized with re-encryption. Specifically, the selected, specific pixels, which have been encrypted during encryption phase, will be re-encrypted if we embed current secret message in them. This process will inevitably lead to a certain degree of distortion for the directly-decrypted stego-images. Based on this, in the pixel selection process, the specific pixels located at bins “0” and “1” are chosen to carry 2-bits secret messages of each pixel, instead of 1-bit message. As stated above, if one embeds 1-bit secret message into specific pixels of a cover image, the binary sequence of all

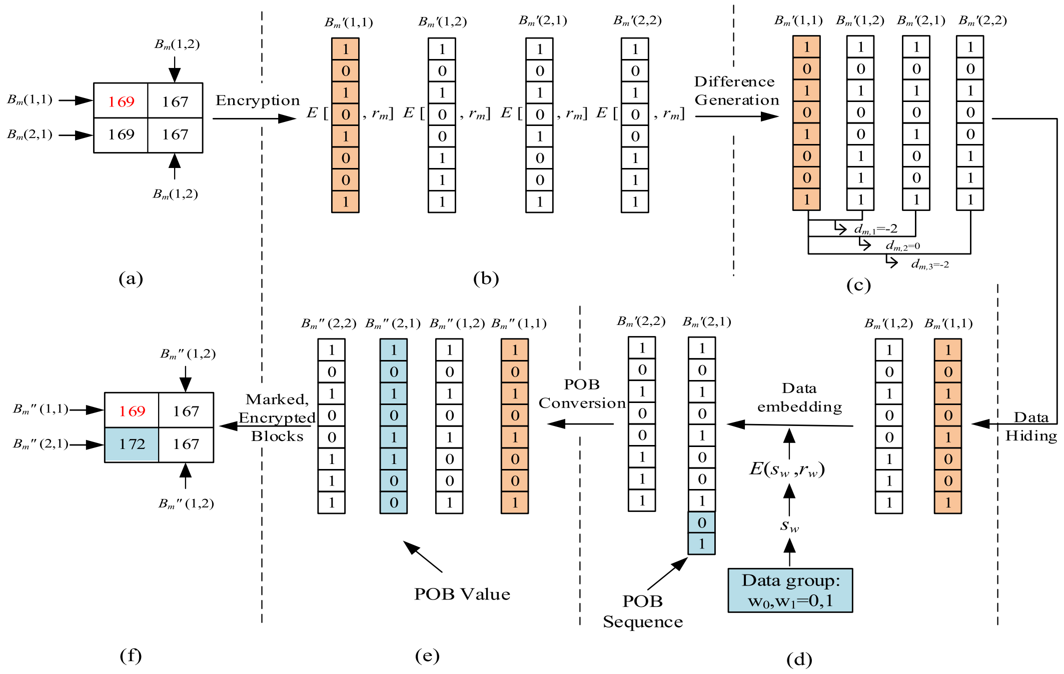

FIGURE 4. Illustration of image encryption, difference generation, data hiding (with $B_m(1,1) = 169$, $B_m(1,2) = 167$, $B_m(2,1) = 169$ and $B_m(2,2) = 167$). (a) Original pixels. (b) Encrypted pixels. (c) Difference generation. (d) Data Embedding. (e) Lossless compression with POB number system. (f) The encrypted blocks with hidden data.

pixels can be converted into 9-bit POB number, and these POB numbers can be further converted to POB values having a maximum value of 125. However, if one embeds 2-bits secret messages into a given cover image, there will exist a maximum POB value 251 for a known $POB(10, r)$ number system. To make the POB value of a given cover image much close to its decimal gray-level value, we embed 2-bits into each pixel with differences "0" and "1" in an image. In this way, the enhancement of the security and the improvement of embedding capacity can be ensured to some extent. Next, a concrete example to account for the proposed RDHEI will be given.

Here, we take an example with $B_m(1, 1) = 169$, $B_m(1, 2) = 167$, $B_m(2, 1) = 169$ and $B_m(2, 2) = 167$ in one block, originally from the local details of standard test Lena, to illustrate the total process of the proposed RDHEI method. As shown in Fig. 4(a), the first pixel marked with red plays a crucial role in the data embedding and data extraction processes since its value after encryption remains unchanged throughout the total process of the data hiding. The four pixels in the block are first converted into binary sequence, then all of them are encrypted with the same random number r_m, as shown in Fig. 4(b). After encryption, the correlations of intra-block have been reserved and can be further exploited for calculating differences, as shown in Fig. 4(c). Next, as for the data embedding, assume a data group to be embedded with two bits, e.g., $\{w_0, w_1\} = \{0, 1\}$. The two bits are encrypted with the encryption key r_w, and then they are embedded directly into the pixel with difference "0", which can be found in the last two bits marked with blue in Fig. 4(d). The total bit sequence of this marked encrypted pixel is termed as a POB sequence, followed by transforming the POB sequence into the POB value, shown in the Fig. 4(e). The conversion details between POB sequence and POB value have been discussed in the previous section. Until now, the marked encrypted block using the proposed RDHEI method has been obtained. It should be noted that, the secret message bits are embedded into specific pixels with differences "0" and "1" (bins "0" and "1"), which is different from the PEE-based RDHEI methods. The proposed method is to embed the secret messages during the generation of difference, whereas the PEE-based RDHEI methods need to shift the difference histogram before embedding secret messages.

After that, the secret messages have been hidden in those pixels with differences "0" and "1" (bins "0" and "1"), and the original cover content will be not exposed to the data hider. When needed to extract the embedded secret bits and decrypt the original bits, the value r that records the number of $1's$ in the marked encrypted binary POB sequence and the length of the embedded secret messages should be known at the receiver side. In order to achieve this, a map recording the value r needs to be losslessly compressed in the data hider side, and the compressed map as well as the length of the secret messages can be embedded into the LSBs of the first pixel in the selected blocks using the data hider key K_h. In view of the true reversibility of the proposed method, the original LSBs of the first pixel in the selected blocks are concatenated at the front of the secret messages and then embedded into the encrypted images with differences "0" and "1". Note that, by using the proposed method, even though the correlations of intra-blocks after encryption are reserved to a certain extent, the embedding synchronized with re-encryption property is able to repair the previous encryption loopholes in time. Unlike the previously block-based RDHEI scheme [17], the correlations of intra-block after data embedding were still reserved, and thus might be restricted for the application scenarios having a higher security level. To sum up, the security performance with the embedding synchronized with re-encryption is better than the previous block-based RDHEI schemes.

2) THE STEPS OF THE DATA EMBEDDING

Once generated the difference histogram of the encrypted image, the next task for the data hider is to embed secret messages into these pixels with bins "0" and "1". The detailed steps can be presented below.

Step 1: Divide the encrypted image into 2×2 blocks.

Step 2: Estimate the embedding capacity through summing up bins "0" and "1" of the encrypted image. If the embedding capacity is less than the length of the additional data, there is no embedding, else complete the following steps.

Step 3: According to the data-hiding key K_h, embed the compressed r value map as well as the length of secret messages obtained from the previous Step 2, into the LSBs of the first pixel in the selected blocks.

Step 4: Embed the secret messages into the reserved room of the bins according to the data-hiding key K_h until the secret messages are fully embedded.

C. DATA EXTRACTION AND IMAGE RECOVERY

When needed to decrypt the encrypted image and extract the embedded secret messages, the following steps can be carried out.

Step 1: Divide the marked encrypted image into 2×2 blocks.

Step 2: Obtain the compressed r value map as well as the length of secret messages from the LSBs of the first pixel in the selected blocks using the data-hiding key K_h.

Step 3: Transform the decimal values $B''(1, 2)$, $B''(2, 1)$ and $B''(2, 2)$ in the current block into POB sequences with the total 10 bits according to the obtained uncompressed r value map, and take the first 8 bits to compare with the first pixel value in this block. If the difference equals to "0" or "1", go to Step 4; else, go to step 5.

Step 4: Extract the remaining 2 bits as the secret data, and decode the pixel using the encryption stream key.

Step 5: Judge whether all of the three decimal values in one block have been completely proceed. If not, repeat Steps 3-5; else, complete the decryption of the first pixel using the same encryption key.

Step 6: Repeat Steps 3-6 until all of the given secret messages are extracted.

Step 7: Convert the decrypted blocks into the inverse permutation phase using permutation key.

After that, the original image can be truly reconstructed and the secret messages can be extracted with no errors.

IV. EXPERIMENTS AND ANALYSES

Extensive experiments are conducted to validate the performance of the proposed RDHEI scheme. Particularly, we focus on evaluating the security for encryption design as well as the performance for RDH respectively.

A. THE PERFORMANCE OF ENCRYPTION

For the purpose of clear demonstration, we choose six standard test images from the dataset [29], to be as the cover images involved in the encryption and the data hiding phases, as shown in Fig. 5. Perceptual security is firstly exploited in our experiments to evaluate the encryption performance. The perceptual security aims to measure the amount of details loss in a more subjective manner. As shown in Fig. 6, the encrypted results reveal completely incomprehensible images, thus it is impossible for the naked eye to acquire the original plaintext contents merely from these encryption counterparts.

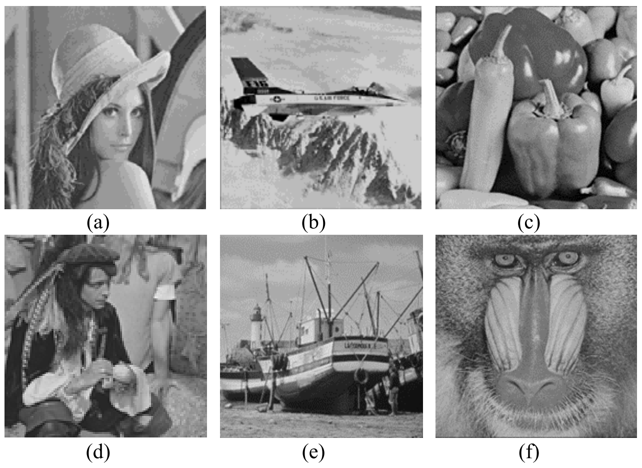

FIGURE 5. Six test images sized 512 × 512. (a) Lena. (b) Plane. (c) Peppers. (d) Man. (e) Boat. (f) Baboon.

1) CORRELATION ANALYSIS

Correlation coefficient depicts the correlation among adjacent pixels in an image, which is used to test the linear relationship between two images. The value of correlation coefficient should be theoretically 0 for an ideal encryption method. Another objective parameter to evaluate the error between the original image and encrypted image is peak signal to noise rate (PSNR).Correlation coefficient and PSNR are defined as follows:

$$\rho_{XY} = \frac{Cov(X, Y)}{\sqrt{D(X)}\sqrt{D(Y)}} \tag{9}$$

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

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} [X(i, j) - Y(i, j)]^2 \tag{11}$$

where X and Y denote the matrix of the original image and the encrypted image, respectively, Cov means the covariance

TABLE 1. Correlation coefficients and PSNR between cover images and encrypted images under different size.

Test images	Size	Correlation coefficients	PSNR	Test images	Size	Correlation coefficients	PSNR
	(64×64)	0.0177	9.7094		(64×64)	0.0095	9.4597
Lena	(128×128)	0.0082	9.6008	Man	(128×128)	0.0011	9.2132
	(256×256)	0.0109	9.5974		(256×256)	0.0130	9.2532
	(512×512)	0.0032	9.5294		(512×512)	0.0069	9.1213
	(64×64)	0.0193	9.1048		(64×64)	-0.0365	9.0973
Plane	(128×128)	-0.0063	8.8206	Boat	(128×128)	0.0127	9.0745
	(256×256)	-0.0025	8.6981		(256×256)	0.0133	9.0661
	(512×512)	0.0016	8.6297		(512×512)	0.0071	8.9518
	(64×64)	0.0331	9.0434		(64×64)	0.0075	10.2620
Peppers	(128×128)	-0.0205	8.9554	Baboon	(128×128)	0.0216	10.1166
	(256×256)	0.0206	9.0325		(256×256)	0.0101	10.0002
	(512×512)	0.0059	8.8777		(512×512)	-5.6140e-04	9.8092

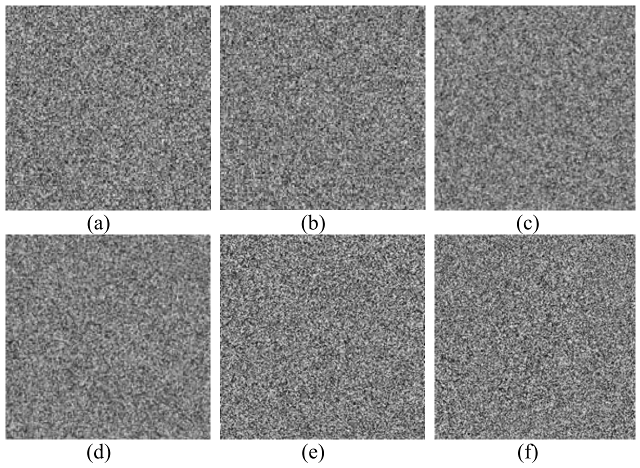

FIGURE 6. Encrypted images. (a) Encrypted Lena (*psnr* = 9.5294). (b) Encrypted Plane (*psnr* = 8.6297). (c) Encrypted Peppers (*psnr* = 8.8777). (d) Encrypted Man (*psnr* = 9.1213). (e) Encrypted Boat (*psnr* = 8.9518). (f) Encrypted Baboon (*psnr* = 9.8092).

and D means the variance, and M and $N$are the weight and height of an image. In the following, we will test the above two parameters with different image sizes, including 64×64, 128×128, 256×256, and 512×512, in order to present the encryption result. The results are listed in Table 1. It can be seen that the correlation coefficients for different image size are almost zeros and the PSNR values are very small, which implies that there is no information leakage from the encrypted images.

2) HISTOGRAM AND ENTROPY ANALYSES

In a specific statistical attack, a deliberate attacker or cryptanalyst may devote to analyzing the predictability of special elements or predictable relations between data segments of original bit stream and cipher code stream. These relationships subsequently obtained can be exploited either to determine the original plaintext contents without knowing the decryption key or to substantially cut down the search space in order to make a brute attack viable. Minimum or no relation between the ciphered image and the original image provides an indicator of the effectiveness of an encryption algorithm resistant statistical attack. In order to detect the existence of the relationships, histogram and information entropy of the original images and its encrypted counterparts are adopted. Information entropy $H(s)$ of a source s is calculated as follows:

$$H(s) = -\sum_{i=1}^{K} p(s_i)\log_2 p(s_i) \tag{12}$$

where $p(s_i)$ represents the probability of the pixel s_i, and K is the number of pixels in an image. For an image of 8-bits in depth, the entropy of an ideal encryption should be 8. It can be seen from Table 2 that a plain image has a smaller entropy value with respect to the corresponding encrypted image. And the entropy value is enlarged with the increase of an image size, i.e., a larger image has a higher entropy value. Since the entropy value for an ideal random image should be 8, a larger image provides a relatively secure encryption. For the histogram distributions shown in Fig. 7, it is obvious that the histograms of all of the encrypted images are uniform, indicating that the encryption is secure in terms of histogram distribution.

3) ROBUSTNESS ANALYSIS FOR DIFFERENT ATTACKS

Known plaintext attack refers to a cryptanalysis model in which an attack has known the plaintexts and corresponding ciphertexts in advance, and she/he tries to reveal key-related crucial information. Chosen plaintext attack has a more powerful strength than known plaintext attack, in which an attacker can choose any plaintexts and generate the corresponding ciphertexts. If an encryption method has a capability to withstand chosen plaintext attack, it is certainly secure against known plaintext attack. To test chosen plaintext attack of the proposed method, we perform differential attack on the standard test image Lena. The cover image Lena and one modified bit of Lena are shown in Figs. 8(a, b). The two images having one bit difference are encrypted with the same key and embedded to generate the marked encrypted results,

TABLE 2. Information entropy between plain images and encrypted images under different image size.

Information entropy							
Test images	Size	Plain image	Encrypted image	Test images	Size	Plain image	Encrypted image
	(64×64)	7.1435	7.9441		(64×64)	7.2512	7.9593
Lena	(128×128)	7.1889	7.9843	Man	(128×128)	7.2906	7.9870
	(256×256)	7.2046	7.9981		(256×256)	7.2882	7.9965
	(512×512)	7.2185	7.9989		(512×512)	7.1926	7.9987
	(64×64)	6.7465	7.9436		(64×64)	7.0277	7.9428
Plane	(128×128)	6.7537	7.9863	Boat	(128×128)	7.0650	7.9833
	(256×256)	6.7332	7.9963		(256×256)	7.0944	7.9954
	(512×512)	6.7059	7.9988		(512×512)	7.1238	7.9987
	(64×64)	7.5283	7.9490		(64×64)	6.8108	7.9567
Peppers	(128×128)	7.5675	7.9864	Baboon	(128×128)	6.9133	7.9868
	(256×256)	7.5797	7.9966		(256×256)	7.0092	7.9967
	(512×512)	7.5925	7.9989		(512×512)	7.1391	7.9990

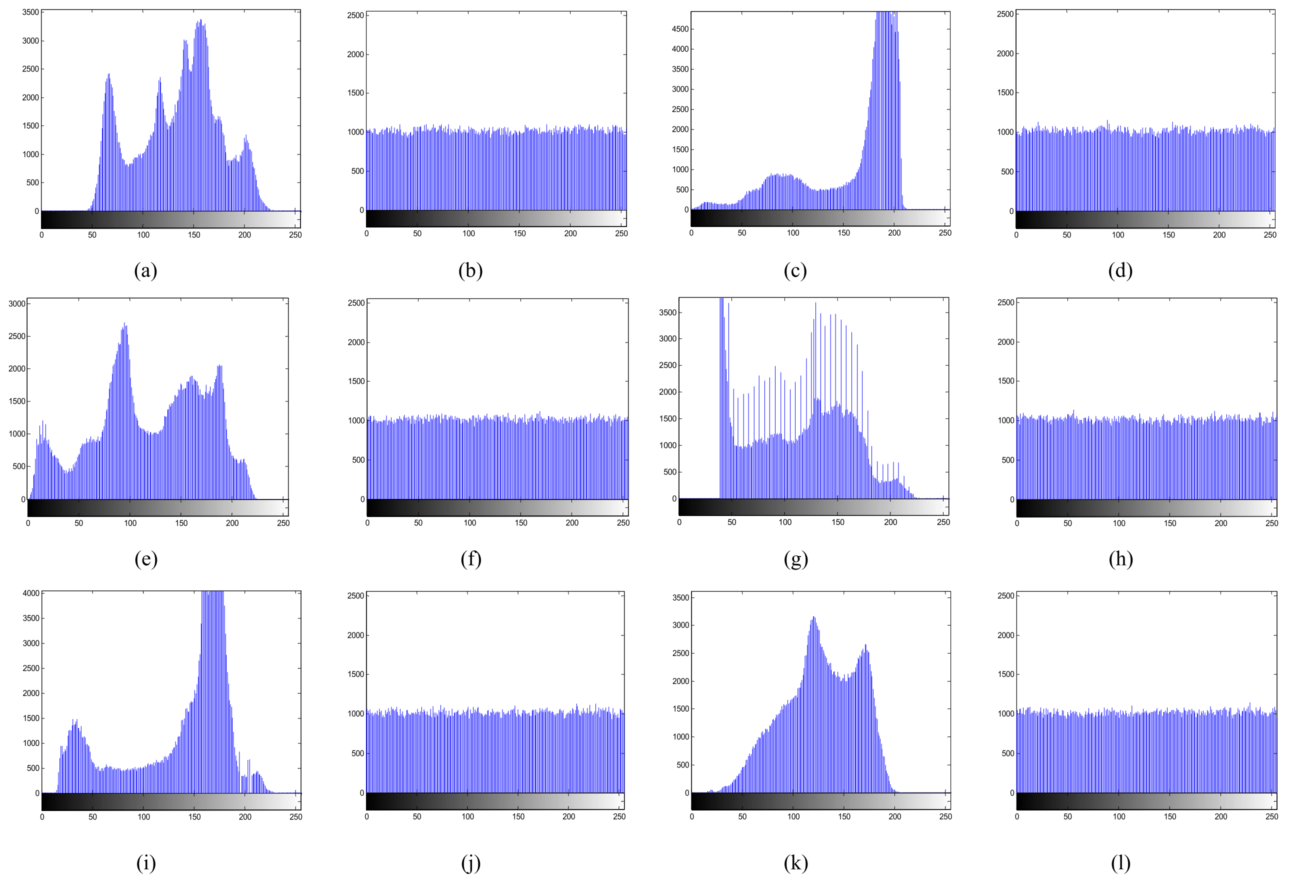

FIGURE 7. Histograms of the original images and encrypted images. (a, c, e, g, i, k) Histograms of original Lena, Plane, Peppers, Man, Boat, Baboon. (b, d, f, h, j, l) Histograms of encrypted Lena, Plane, Peppers, Man, Boat, Baboon.

as shown in Figs. 8(c, d). As seen, only one bit difference on the two images brings two completely different marked encrypted results. In other words, no matter which image is selected, the marked encrypted result is completely different from each other. Thus, the proposed method can withstand known/chosen plaintext attack.

In addition, we also discuss the robustness after suffering from common image processing attacks, by adding 1% Salt & Noise, 1% Gaussian noise, 80×80 image cut, LSB plane removed, MSB plane removed, Gaussian filter with sigma $= 2.5$ and filter size $= 2 \times 2$, and Mean filter with filter size $= 2 \times 2$, to the marked encrypted image Lena, as shown in Fig. 9. As seen, Figs. 9(b-d, i-l) show the marked encrypted results separately with 1% Salt & Noise, 1% Gaussian noise, 80×80 image cut, LSB plane removed, MSB plane removed, Gaussian filter with sigma $= 2.5$ and

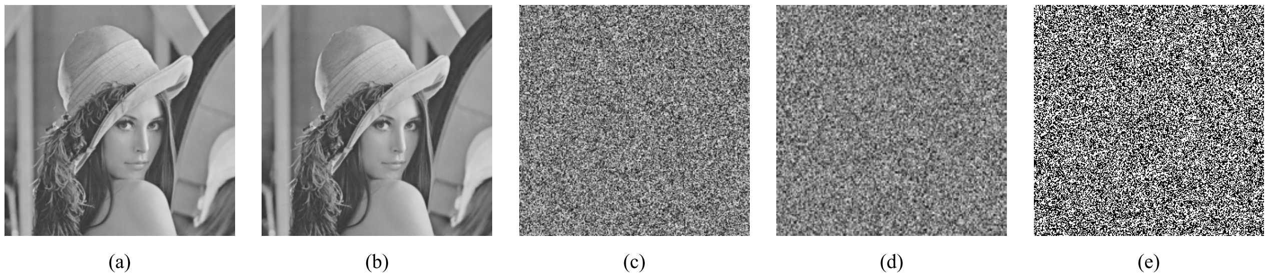

FIGURE 8. Differential analysis with Lena. (a) The original Lena. (b) Only one modified bit of (a). (c) Marked encrypted result of (a). (d) Marked encrypted result of (b). (e) The difference between (c) and (d).

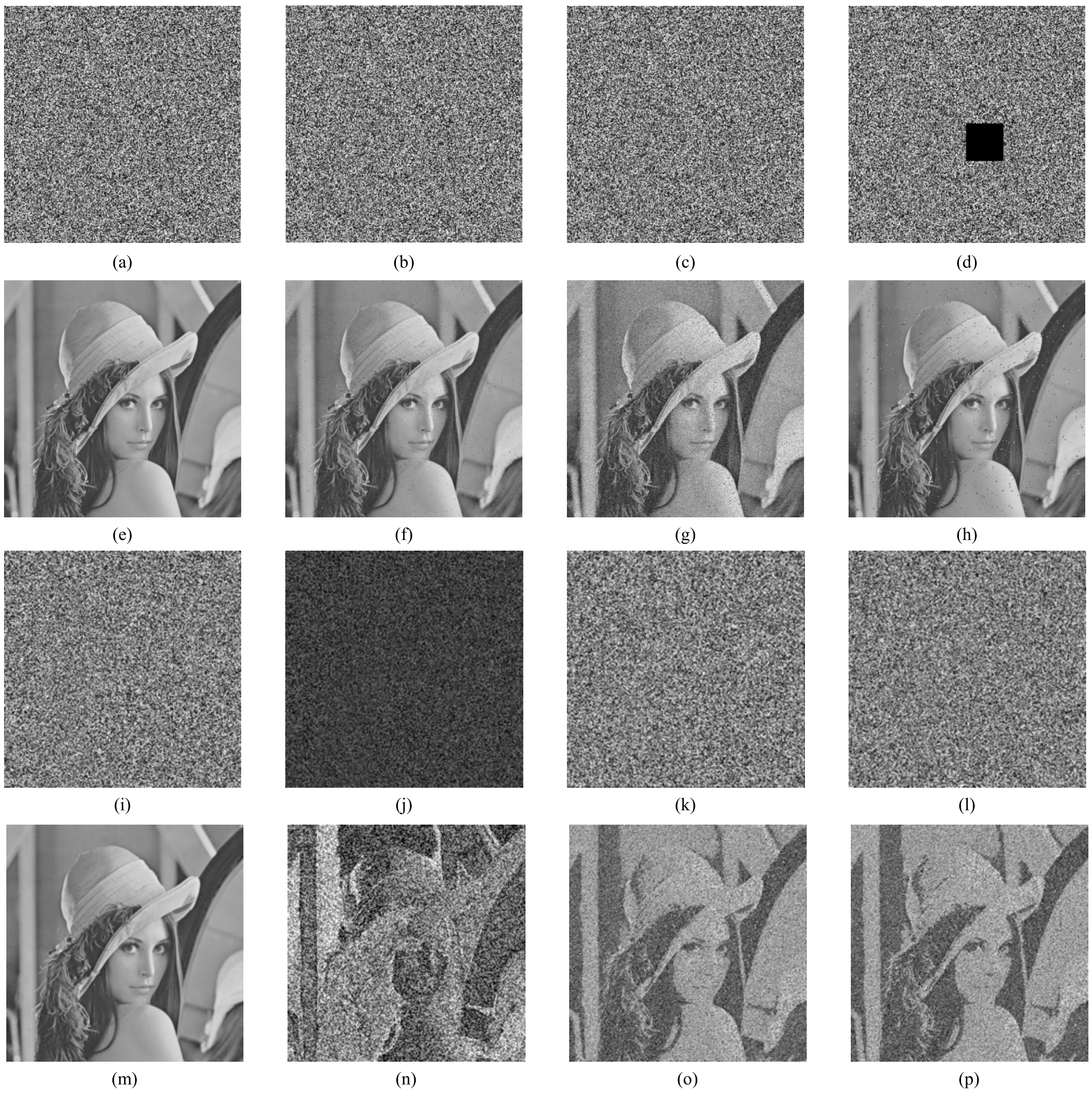

FIGURE 9. Attack analysis of the marked encrypted Lena. (a, e) Without attack and the recovered image; (b, f) with 1% Salt & Noise and the recovered image; (c, g) with 1% Gaussian noise and the recovered image; (d, h) with 80 × 80 image cut and the recovered image; (i, m) with the LSB plane removed and the recovered image; (j, n) with the MSB plane removed and the recovered image; (k, o) with sigma = 2.5, filer size = 2 × 2 Gaussian filter; (l, p) with filter size = 2 × 2 Mean filter.

filter size = 2 × 2, and Mean filter with filter size = 2 × 2. Figs. 9(f-h, m-p) show the corresponding recovered images. From the results of the recovered images, one can know that, when suffering from MSB plane removed, the final recovered image has a worse visual quality, but as for LSB plane removed, the visual quality of the final recovered image

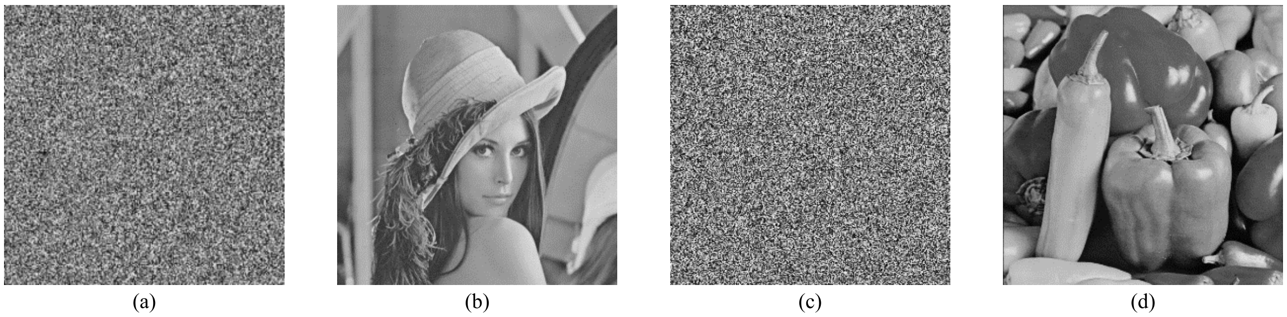

FIGURE 10. **Marked and encrypted images and final recovery images for Lena and Peppers. (a) The marked and encrypted Lena (embedding capacity 79690 bits, and $psnr = 9.4144$); (b) the final recovery Lena ($psnr = \infty$); (c) the marked and encrypted Peppers (embedding capacity 53766 bits, and $psnr = 8.6475$); (d) the final recovery Peppers ($psnr = \infty$).**

is satisfactory. For other two situations where noise pollution and filters are added to the marked encrypted image Lena separately, which can be found in Figs. 9(b-c, k-l), most contents of the original image still have been recovered, as shown in Figs. 9(f-g, o-p). Thus, the proposed method can resist against several typical image processing attacks.

B. THE PERFORMANCE OF RDH

1) EMBEDDING CAPACITY

Assume 79690 bits and 53766 bits secret messages are separately embedded into the encrypted Lena and Peppers images shown in Figs. 10(a, c), and the embedding rates are 0.3039 and 0.2051. We can see that the PSNRs of the two marked encrypted images are less than that of the merely encrypted images, mainly due to that the POB-based embedding can re-encrypt the previous encrypted images. Using POB number to compress and vacate room is extremely essential for block-based RDHEI schemes, mainly because the re-encryption can cope with the potential loopholes caused by intra-block correlations, thus making a higher encryption security. Besides, a joint data extraction and image recovery method is adopted in this paper, and the final recovery image is shown in Figs. 10(b, d).

Also, in order to present the result of the final recovered image in detail, we utilize six test images to test PSNR, structural similarity measure index (SSIM) and normalized cross correlation (NCC) under the maximum capacity. The definitions of the SSIM and NCC are presented as follows.

$$SSIM\ (X, Y) = [l(X, Y)]^{\alpha} \cdot [c(X, Y)]^{\beta} \cdot [s(X, Y)]^{\gamma} \tag{13}$$

$$l(X, Y) = \frac{2\mu_X\mu_Y + C_1}{\mu_X^2 + \mu_Y^2 + C_1} \tag{14}$$

$$c(X, Y) = \frac{2\sigma_X\sigma_Y + C_2}{\sigma_X^2 + \sigma_Y^2 + C_2} \tag{15}$$

$$s(X, Y) = \frac{\sigma_{XY} + C_3}{\sigma_X\sigma_Y + C_3} \tag{16}$$

where $\mu_X, \mu_Y, \sigma_X, \sigma_Y$, and σ_{XY} are the local means, standard deviations, and cross-covariance for images X and Y. When using the default exponents and selections of C_3, the SSIM expression is given by:

$$SSIM\ (X, Y) = \frac{(2\mu_X\mu_Y + C_1)(2\sigma_{XY} + C_2)}{(\mu_X^2 + \mu_Y^2 + C_1)(\sigma_X^2 + \sigma_Y^2 + C_2)} \tag{17}$$

$$NCC = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} X(i,j) \times Y(i,j)}{\sum_{i=1}^{M}\sum_{j=1}^{N} X(i,j)^2} \tag{18}$$

where M and N are the weight and height of an image respectively. The evaluation metric NCC defined in (18) is mainly used to determine the extent that the cover image and the recovered image are close to each other. The evaluation metric SSIM mainly exploits the luminance, contrast and structure to analyze the similarity between the cover image and the recovered image. The values of both SSIM and NCC are in [0, 1].

For the maximum embedding capacity of the proposed method, due to only difference "0" and "1" to be embedded, the total capacity is thus exactly twice the sum of differences "0" and "1", and the embedding rate is shown in (19). All results for six test images are shown in Table 3. It is obvious that the maximum capacity is mainly relied on the characteristic of an image. The smoother the image, the higher the capacity. The PSNR, SSIM and NCC for all recovered images are perfect, indicating that the proposed method equips with the real reversibility.

$$rate = \frac{2^{*}(difference_0 + difference_1)}{total\ \ pixel\ \ so\ \ fanimage} \tag{19}$$

In addition to the above analysis of the final recovery image, we also compare the embedding rate with our method and the-state-of-the-art works [13]–[16], [19]–[24], as listed in Table 4. The embedding capacity of our method has the second highest capacity, this is mainly because the embedding capacity is equal to twice of two peak points (differences "0" and "1"). The capacity [19] is the highest, but the method is mainly applied in binary image. And other methods have a quite lower embedding capacity compared to our proposed method. Thus, the proposed method is acceptable in terms of maximum embedding capacity.

TABLE 3. The maximum capacity and the PSNR, NCC, and SSIM of recovered image for different images.

Image (512×512)	Embedding capacity			Recovered image		
	Difference D_0	Difference D_1	Total capacity $2\times(D_0+D_1)$	PSNR(dB)	NCC	SSIM
Lena	20918	18927	79690	∞	1	1
Plane	28991	21979	101940	∞	1	1
Peppers	13603	13280	53766	∞	1	1
Man	20348	15026	70748	∞	1	1
Boat	18384	16847	70462	∞	1	1
Baboon	6441	6503	25888	∞	1	1

TABLE 4. Comparison of embedding rate among the proposed method and related methods.

Method	Test images					
	Lena	Plane	Peppers	Man	Boat	Baboon
Zhang [13]	<=0.0039	<=0.0039	<=0.0015	<=0.0015	<=0.0039	<=0.0009
Hong [14]	<=0.0039	<=0.0039	<=0.0039	<=0.0015	<=0.0039	<=0.0009
Zhang [15]	<0.035	<0.03	<0.03	<0.02	<0.03	<=0.01
Ma et al. [16]	<=1	<=1	<=0.8	<=1	<=1	<=0.8
Yi et al. [19]	<=1.68	<=1.87	<=1.68	<=1.68	<=1.87	<=1.45
Zhou et al. [20]	<0.75	<0.75	<0.75	<0.75	<0.75	<0.75
Chen et al. [21]	<=0.002	<=0.002	<=0.002	<=0.002	<=0.002	<=0.002
Shiu et al. [22]	<=0.004	<=0.004	<=0.004	<=0.004	<=0.004	<=0.004
Zhang et al. [23]	<0.008	<0.008	<0.008	<0.008	<0.008	<0.008
Li et al. [24]	<0.008	<0.008	<0.008	<0.008	<0.008	<0.008
Ours	<=0.3039	<=0.3889	<=0.2051	<=0.2698	<0.2687	<=0.0988

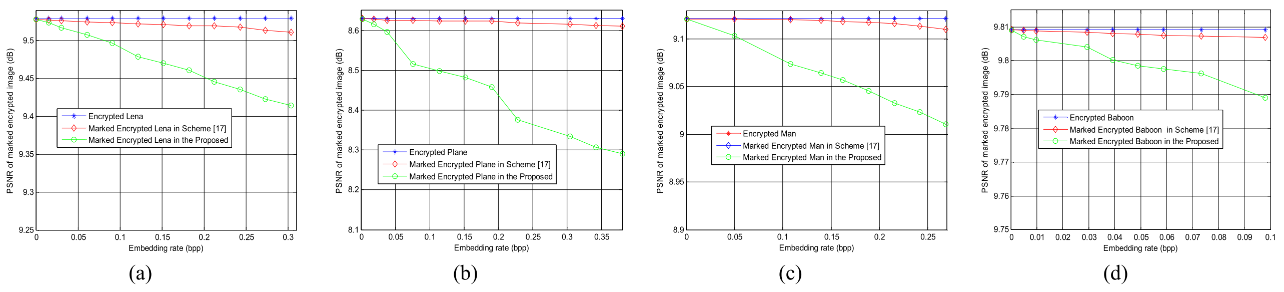

FIGURE 11. Comparisons of PSNR values of the marked encrypted images with respect to the encrypted images under different embedding rates. (a) Lena; (b) Plane; (c) Man; (d) Baboon.

2) COMPARISONS

In this section, PSNR values and SSIM values under different embedding rates are further tested to compare with our method and other methods. In addition, we also analyze the performance of the proposed method.

Firstly, in order to test the effectiveness of the proposed embedding synchronized with re-encryption, PSNRs for the marked encrypted images with respect to the encrypted images between our method and [17] are tested, and the results are plotted in Fig. 11. As seen, the PSNR values are obviously decreased with the increase of embedding rates, mainly because more pixels needs to be modified to implement embedding; once modified, pixels to be embedded will be re-encrypted. But for the method [17], the embedding only adds 1 or subtracts 1 on the specific pixels in the encrypted images, and the correlation of intra-block in encrypted images still highly correlated, thus causing a little change for PSNR values of the marked encrypted image. Like many other methods that are similar to the method [17], if they adopt a block-size strategy to fully encrypt an original image, and PEE-based RDHEI techniques to hide secret messages, then the intra-block correlations in encrypted images must be preserved in order to data embedding, thus these methods are likely to incur several possible security loopholes. Based on this, the proposed embedding synchronized with re-encryption can improve encryption security to some degree.

Secondly, we also test the recovery accuracy compared our method with methods [13], [15], [18]. Figs. 12-13 show the values of PSNR and SSIM for recovery images under different embedding rates. It can be observed that, because the POB number system that carries out a lossless compression and embedding is utilized in our method, the recovery accuracy is perfect. As for the method [18], obviously, it cannot ensure the real reversibility when the embedding rates are enlarged. However, due to the utilization of an accuracy prediction method, it can generally improve the recovery accuracy compared with methods [15] and [18], especially

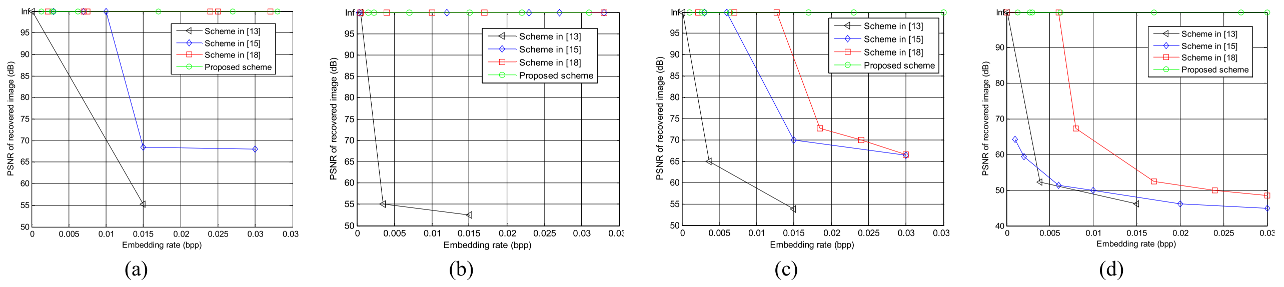

FIGURE 12. Comparisons of PSNR values for recovered image under different embedding rate. (a) Lena; (b) Plane; (c) Man; (d) Baboon.

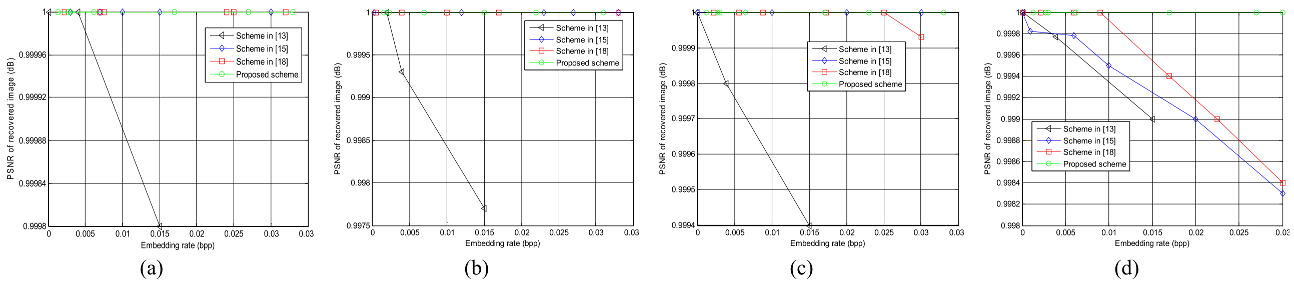

FIGURE 13. Comparisons of SSIM values for recovered image under different embedding rate. (a) Lena; (b) Plane; (c) Man; (d) Baboon.

TABLE 5. Comparisons of feature among the proposed method and related methods.

Method	Encryption Method	Error in image recovery	Reserving room after embedding	Embedding Synchronized with re-encryption	Computational complexity
Zhang [13]	Stream cipher	No	Yes	No	Low (M×N)
Hong [14]	Stream cipher	No	Yes	No	Low (M×N)
Zhang [15]	Stream cipher	No	Yes	No	Low (2^S×M×N)
Ma et al. [16]	Stream cipher	Yes	No	No	High (M×N)
Yi et al. [19]	Stream cipher	Yes	Yes	No	High (M×N)
Zhou et al. [20]	Stream cipher	No	Yes	No	Moderate (2^n+ M×N)
Chen et al. [21]	Paillier encryption	Yes	Yes	No	High (M×N)
Shiu et al. [22]	Paillier encryption	Yes	Yes	No	High (M×N)
Zhang et al. [23]	Paillier encryption	Yes	Yes	No	High (M×N)
Li et al. [24]	Paillier encryption	Yes	Yes	No	High (M×N)
Ours	Stream cipher	Yes	Yes	Yes	Low (M×N)

for the images with relatively smoother distribution. To sum up, the proposed method is acceptable.

Finally, we analyze the performance by comparing the proposed method and the related methods [13]–[16], [19]–[24], Table 5 presents some feature comparisons on the aspects of encryption method, error image recovery, reserving room after embedding, embedding synchronized with re-encryption, and computational complexity. For those methods [21]–[24] using paillier encryption as a cryptosystem, the resource consumptions are greatly higher, and relatively have the high computational complexity. The reserving room after embedding adopted by Ma *et al.* [16] is poorly practical, because it needs a content owner to carry out an extra preprocessing before image encryption. The superiority of embedding synchronized with re-encryption embraces merely our method, mainly due to that the POB number system serves as a compression for embedding and re-encryption. Moreover,

TABLE 6. Running times of encryption embedding extraction and decryption for different images.

Images	Running times (seconds)		
	Encryption	Embedding	Extraction and recovery
Lena	0.7173s	15.7819s	16.5317s
Plane	0.6041s	20.6041s	21.3367s
Peppers	0.70383	13.7315s	14.5245s
Man	0.6469s	14.7974s	15.4782s
Boat	0.6150s	14.7324s	15.4723s
Baboon	0.5395s	7.1327s	7.7289s

in terms of other features, the proposed method is also satisfactory. In addition, running times of encryption, embedding and extraction and decryption are also considered in our method. The results have been listed in Table 6, where the speed is measured with seconds based on MATLAB 2012b implements on a computer with Intel® core^{TM}i3 CPU. It can be observed that, the running times for encryption process are all small, mainly because a block-size XOR encryption is utilized in our method. But the embedding occupies a relatively longer time. On one hand, a transformation from POB number to POB value needs to be performed in order to achieve embedding synchronized with re-encryption; on the other hand, a higher capacity will cause a larger amount of time consumption (compared with a smoother image Plane and a texture image Baboon). Similar to the embedding, the extraction needs to perform a reverse transformation from POB value to POB number and the image decryption needs to perform an inverse of the image encryption, thus has a similarly time consumption. To sum up, based on the above analysis, the proposed method is acceptable.

V. CONCLUSION

In this paper, the properties of POB number system are adopted for reversible data hiding in encrypted images (RDHEI). By using the POB number system, the security loopholes incurred by the preservation of intra-block correlations for data embedding can be resolved through the embedding synchronized with re-encryption to some degree. On the other hand, although we utilize the two peak points to carry secret messages, the shifting for vacate room need not exist, which may server as a version of lossless compression embedding or an extension of prediction error expansion. The reversibility can be well ensured by the POB number system, and the embedding capacity is acceptable. As every coin has two sides, the embedding synchronized with re-encryption also brings a non-negligible drawback, causing the poor visual quality of the directly decrypted images. This is why we adopt joint data extraction and image recovery. In the future, we mainly concentrate on how to improve the visual quality of the directly decrypted images.

REFERENCES

[1] K.-L. Chung, Y.-H. Huang, P.-C. Chang, and H.-Y. M. Liao, "Reversible data hiding-based approach for intra-frame error concealment in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 11, pp. 1643–1647, Nov. 2010.

[2] X. T. Wang, C. Y. Shao, X. G. Xu, and X. M. Niu, "Reversible data-hiding scheme for 2-D vector maps based on difference expansion," *IEEE Trans. Inf. Forensics Security*, vol. 2, no. 3, pp. 311–320, Sep. 2007.

[3] J. Wang, J. Ni, X. Zhang, and Y.-Q. Shi, "Rate and distortion optimization for reversible data hiding using multiple histogram shifting," *IEEE Trans. Cybern.*, vol. 47, no. 2, pp. 315–326, Feb. 2017.

[4] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding—New paradigm in digital watermarking," *EURASIP J. Adv. Signal Process.*, vol. 2002, no. 2, pp. 185–196, 2002.

[5] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896, Aug. 2003.

[6] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, Mar. 2006.

[7] C. F. Lee and H. L. Chen, "Reversible data hiding based on histogram modification of prediction-error," *J. Photographic Sci.*, vol. 59, no. 5, pp. 278–292, 2011.

[8] X. Li, W. Zhang, X. Gui, and B. Yang, "Efficient reversible data hiding based on multiple histograms modification," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 9, pp. 2016–2027, Sep. 2015.

[9] N. Zhu, Y. Wang, J. Liu, J. Xie, and H. Zhang, "Optical image encryption based on interference of polarized light," *Opt. Express*, vol. 17, no. 16, pp. 13418–13424, 2009.

[10] C. Fu, J.-J. Chen, H. Zou, W.-H. Meng, Y.-F. Zhan, and Y.-W. Yu, "A chaos-based digital image encryption scheme with an improved diffusion strategy," *Opt. Express*, vol. 20, no. 3, pp. 2363–2378, 2012.

[11] Y. Wu, G. Yang, H. Jin, and J. P. Noonan, "Image encryption using the two-dimensional logistic chaotic map," *J. Electron. Imag.*, vol. 21, no. 1, pp. 013014-1–013014-15, 2012.

[12] Y. Wang, M. Miao, J. Shen, and J. Wang, "Towards efficient privacy-preserving encrypted image search in cloud computing," *Soft Comput.*, vol. 23, no. 6, pp. 2101–2112, 2019.

[13] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Process. Lett.*, vol. 18, no. 4, pp. 255–258, Apr. 2011.

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

[15] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 826–832, Apr. 2012.

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

[17] M. Li, D. Xiao, Y. Zhang, and H. Nan, "Reversible data hiding in encrypted images using cross division and additive homomorphism," *Signal Process. Image Commun.*, vol. 39, pp. 234–248, Nov. 2015.

[18] C. Qin, W. Zhang, F. Cao, X. Zhang, and C.-C. Chang, "Separable reversible data hiding in encrypted images via adaptive embedding strategy with block selection," *Signal Process.*, vol. 153, pp. 109–122, Dec. 2018.

[19] S. Yi and Y. Zhou, "Binary-block embedding for reversible data hiding in encrypted images," *Signal Process.*, vol. 133, pp. 40–51, Apr. 2017.

[20] 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 Trans. Circuits Syst. Video Technol.*, vol. 26, no. 3, pp. 441–452, Mar. 2016.

[21] Y.-C. Chen, C.-W. Shiu, and G. Horng, "Encrypted signal-based reversible data hiding with public key cryptosystem," *J. Vis. Commun. Image Represent.*, vol. 25, no. 5, pp. 1164–1170, Jul. 2014.

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

[23] X. Zhang, J. Long, Z. Wang, and H. Cheng, "Lossless and reversible data hiding in encrypted images with public-key cryptography," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 9, pp. 1622–1631, Sep. 2016.

[24] M. Li and Y. Li, "Histogram shifting in encrypted images with public key cryptosystem for reversible data hiding," *Signal Process.*, vol. 130, pp. 190–196, Jan. 2017.

[25] A. Sreekumar and S. B. Sundar, "An Efficient Secret Sharing Scheme for n out of n scheme using POB-number system," *Int. J. Inf. Process.*, 2009.

[26] P. Singh, B. Raman, N. Agarwal, and P. K. Atrey, "Secure cloud-based image tampering detection and localization using POB number system," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 13, no. 3, p. 23, 2017.

[27] P. Singh, B. Raman, and N. Agarwal, "Toward encrypted video tampering detection and localization based on POB number system over cloud," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 9, pp. 2116–2130, Sep. 2018.

[28] P. Singh, N. Agarwal, and B. Raman, "Secure data deduplication using secret sharing schemes over cloud," *Future Gener. Comput. Syst.*, vol. 88, pp. 156–167, Nov. 2018.

[29] *Miscelaneous Gray Level Images*. Accessed: Mar. 13, 2014. [Online]. Available: http://decsai.ugr.es/cvg/dbimagenes/g512.php

HUA REN received the B.S. and M.S. degrees from Henan Normal University, Henan, China, in 2016 and 2019, respectively. She is currently pursuing the Ph.D. degree with the Beijing University of Posts and Telecommunications. Her current research interests include data hiding and digital watermarking.

SHAOZHANG NIU received the B.S. and M.S. degrees from Beijing Normal University, Beijing, China, in 1985 and 1988, respectively, and the Ph.D. degree from the Beijing University of Posts and Telecommunications, Beijing, in 2004, where he is currently a Professor with the School of Computer Science. His research interests include steganography, steganalysis, and digital forensics.

XINYI WANG received the B.Sc. degree in mathematics and applied mathematics from the Beijing University of Posts and Telecommunications, in 2012, and the M.S. degree in applied mathematics from China Agricultural University, in 2017. She is currently pursuing the Ph.D. degree with the Beijing University of Posts and Telecommunications. Her current research interests include information security and digital image forensics.