Received December 13, 2019, accepted January 16, 2020, date of publication January 22, 2020, date of current version February 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2968577

Fidelity Preserved Data Hiding in Encrypted Images Based on Homomorphism and Matrix Embedding

SISHENG CHEN[1,2], CHIN-CHEN CHANG[2,4], (Fellow, IEEE), AND QUNYING LIAO[3]
[1]Engineering Research Center for ICH Digitalization and Multi-source Information Fusion (Fuqing Branch of Fujian Normal University), Fujian Province University, Fuzhou 350300, China
[2]Department of Information Engineering and Computer Science, Feng Chia University, Taichung 40724, Taiwan
[3]School of Mathematics Science, Sichuan Normal University, Chengdu 610066, China
[4]School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China

Corresponding author: Chin-Chen Chang (alan3c@gmail.com)

This work was supported in part by the Youth Teacher Research Program of Fujian Province under Grant JT180618, and in part by the Natural Science Foundation of Fujian Province of China under Grant 2018J01788.

ABSTRACT This paper proposes a data hiding scheme for high quality stego-images in the encrypted images based on a homomorphic encryption algorithm and matrix embedding method. To achieve the message embedding in the encryption domain, we first present an image encryption algorithm satisfying additive homomorphism based on the stream cipher. Then, we offer a matrix embedding method using Golay code, which has good embedding efficiency. In the proposed scheme, we pre-code the least significant bit (LSB) vectors of the cover image into the codewords before image encryption, so that we can implement the matrix embedding in the encrypted image. The experimental results show that the proposed scheme has excellent security performance. We can get a high quality stego-image after the image decryption and recover the cover image into a fidelity image while providing considerable payload.

INDEX TERMS Fidelity preserved, homomorphism, matrix embedding, Golay code, data hiding, encrypted image.

I. INTRODUCTION

Data hiding in images is a scheme which hides a secret message in a cover image. It can be used to send the secret message through the cover image or embed important information, such as copyright information, authentication information, or management information into the cover image. In the past decade, many data hiding schemes have been proposed. Schemes [1]–[3] embedded the data in the spatial domain. Schemes [4]–[6] implemented the data hiding in the compressed domain.

Recently, a new technology called data hiding in encrypted image (DHEI) has become an interesting notion in the data hiding arena. Considering such application, in a cloud storing and computing system, a user will send the encrypted images to the cloud server to protect the privacy data. For data management, this process still requires that the server can embed the messages, such as copyright information or authentication message, in uploaded encrypted image. The DHEI technology just meets the requirements.

Many data hiding schemes in encrypted images have been proposed. To summarize the methods of those schemes, there are two main methods to embed data in a cipher image: keeping the local redundancy in the encrypted image [7]–[15] and using the homomorphic evaluation [16]–[20]. The first method usually uses the stream cipher to encrypt the image. Schemes [7]–[11] encrypted the images with the stream cipher, which kept the redundancy of the inter-pixel on nature images. Scheme [12] divided an encrypted image into blocks and embedded the secret data by modifying three LSBs of portion pixels of the encrypted block. Schemes [13]–[15] used a specific encryption algorithm to keep spatial corrections of the image blocks in the encrypted image, then they embedded the secret data based on the corrections of the encrypted image blocks. It is well known that keeping the redundancy of the encrypted image may affect the security of the cipher image, such as being attacked by statistical analysis. The second method usually applies a public key to encrypt the image. Chen *et al.* [16] transformed a pixel into two

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

parts before image encryption and used the two pixels' LSBs to carry the secret data. Zhang *et al.* [17] shrank the image histogram before encryption and embedded the secret data by modifying the cipher text values using the homomorphism of encryption algorithm. Scheme [18] pre-processed the original image to make the adjacent pixels pairwise odd or even, then used homomorphism of the encryption algorithm to achieve the data embedding in the pixel pairwise. Zheng *et al.* [19] used the self-blinding property of homomorphic cryptography to embed the secret data in the original image through shifting the pixel values of the cipher image according to the secret data into different intervals. This method can embed the secret data in the intervals but does not affect the image decryption. The homomorphic encryption method can solve the security problem of the first method. It is very suitable for the schemes reversing room before encryption. But, as known, the public key encryption is practically inefficient and it will lead to the expansion of the image size. Chen *et al.* [20] pointed out that it is necessary to construct efficient encryption algorithm for image encryption suiting for DHEI.

In addition, for different applications, different DHEI schemes have focused on improving different metrics, such as high capacity, the reversibility, or the quality of a stego-image. The quality of a stego-image is the main object of the applications such as copyright protection and the image authentication. The method to improve the quality is to enhance the embedding efficiency. Crandall [21] presented the matrix embedding to improve the embedding efficiency for reducing the distortion of the cover image. Some works [22], [23] have studied this process extensively. In order to improve the embedding capacity and embedding efficiency, other matrix embedding methods have been proposed using linear block code [24]–[28]. Filler *et al.* [24] used the syndrome-trellis code to complete the data embedding. Scheme [26] proposed a fast algorithm for finding the coset leader in Hamming code to decrease the embedding cost. Yang and Chen [28] presented a low complexity data hiding scheme in binary images based on RM codes. The big challenge of the matrix embedding based on linear code is finding the coset leader. Therefore, it is important to choose a suitable linear code to construct the matrix embedding, to ensure that it can find the coset leader fast and have high embedding efficiency.

This paper focuses on improving the stego-image quality of the data hiding in encrypted image. Firstly, we present an image encryption algorithm based on the stream cipher, which satisfies the additive homomorphism. Then, we give a fast matrix embedding steganography based on the Golay code [23, 12, 7], which has high embedding efficiency and considerable payload. Next, we propose the data hiding scheme in the encrypted image. In order to use the coding embedding in the encrypted image, we pre-code to the LSB vectors of the cover image before encryption. Because of the homomorphism of the encryption algorithm, we can embed the secret message into the cover image. Finally, we analyze the security of the cipher image and the performance of the proposed scheme. The experiment and the analysis results show that proposed scheme has excellent security performance. We can get a high quality stego-image and preserve the fidelity of cover image.

The rest of paper is organized as follows. In Section 2, we present the homomorphism cryptosystem and the matrix embedding steganography using the Golay code [23, 12, 7]. Section 3 proposes our data hiding scheme. In Section 4, the experiment and analysis are given. Our scheme is concluded in Section 5.

II. PRELIMINARIES

In this section, we firstly introduce the image encryption algorithm, which satisfies the additive homomorphism. Then, we present the matrix embedding steganography using Golay code [23, 12, 7] and analyze its advantages in embedding efficiency.

A. HOMOMORPHIC CRYPTOSYSTEM

Homomorphic cryptography allows for operations on a cipher image, with a result equal to performing directly on the plain image. This homomorphism can be defined as follow:

$$E_k(p_1 \oplus p_2) = E_k(p_1) \otimes E_k(p_2), \quad \forall p_1, p_2 \in M, \tag{1}$$

where M is the plaintext space, $E_k()$ is the encryption function with encryption key k, $\oplus$ denotes the operation in a plaintext domain and $\otimes$ denotes the operation in the cipher domain. So, the sender can encrypt the privacy image by the homomorphic cryptography and the receiver completes the data management by the homomorphic evaluation. Thus, the homomorphic encryption algorithm is widely applied to encrypt the image in data hiding scheme, such as schemes [16]–[19] which have been introduced in section 1.

Stream ciphers are always applied to encrypted image for its simplicity. But, Khelifi [29] pointed out the weakness of the cipher image encrypted by the stream cipher. Because of the redundancy of the inter-pixel on natural images, it can reveal the visual content of the encrypted images by estimating the encrypted key. It also points out that one of the methods to overcome the weakness is to permute pixels before encryption.

Here, we present an image cryptograph algorithm based on the stream cipher and the matrix scrambling. It can verify that the encryption algorithm satisfies the additive homomorphism. The detailed encryption process contains the following four steps.

step 1 (Initialization): Divide the image into non-overlapping block with a $1 \times n$ size. Choose a matrix A sized $n \times n$ as a scrambling matrix, which is invertible in module 256, i.e. $gcd(|A|, 256) = 1$.

Step 2 (Key Generation): Use the key generator of the stream cipher with the seed key *key* to produce a key stream, and divide it into groups denoted as $K = (ek_1, ek_2, \cdots, ek_L)$, where $ek_i = (k_{i,1}, k_{i,2}, \cdots, k_{i,n})^T$, $k_{i,j} \in Z_{256}, i = 1, 2, \cdots, L, j = 1, 2, \cdots, n$ and L denotes cycle of the key stream.

Step 3 (Encryption): For each block, get the grayscale values vector $p_i = (p_{i,1}, p_{i,2}, \cdots, p_{i,n})^T$, and get the block encryption key $ek_i = (k_{i,1}, k_{i,2}, \cdots, k_{i,n})^T$ from K. The cipher block of the p_i is denoted as $c_i = (c_{i,1}, c_{i,2}, \cdots, c_{i,n})^T$. The encryption computing is defined as follows:

$$c_i \equiv A \cdot p_i + ek_i (\text{mod} 256). \tag{2}$$

The encryption process is denoted as $c = E_{ek}(p)$.

Step 4: Decryption: When receiving the cipher text $c_i = (c_{i,1}, c_{i,2}, \cdots, c_{i,n})^T$, it can obtain the plain text as follow.

$$p_i \equiv A^{-1} \cdot (c_i - ek_i)(\text{mod} 256). \tag{3}$$

The decryption process is denoted as $p = D_{ek}(c)$.

It can verify that there is additional homomorphic operation as follows:

$$\begin{aligned} E_{ek_i}(p_i) + E_{ek_j}(p_j) &\equiv A \cdot p_i + ek_i + (A \cdot p_j + ek_j)(\text{mod} 256) \\ &\equiv A \cdot (p_i + p_j) + (ek_i + ek_j)(\text{mod} 256) \\ &\equiv (A \cdot p_i + ek_i) + (A \cdot p_j + ek_j)(\text{mod} 256) \\ &= E_{ek_i + ek_j}(p_i + p_j). \end{aligned}$$

That is

$$E_{ek_i}(p_i) + E_{ek_j}(p_j) = E_{ek_i + ek_j}(p_i + p_j). \tag{4}$$

The security of the proposed image encryption algorithm is based on the security of the encryption key K. We use the key generator of the stander stream cipher to guarantee the security of the encryption key.

B. MATRIX EMBEDDING

Two important factors to evaluate an image steganography are the embedding rate (ER) and the embedding efficiency (EE). Assume that l_{cp} pixels of the cover image carry the l_m bits data by adjusting l_{cb} bits pixel values. Then, EE and ER are defined as

$$EE = l_m / l_{cb}, \tag{5}$$

$$ER = l_m / l_{cp}. \tag{6}$$

In the image steganography based on $[n, k]$ linear code, n pixels of cover image carry $(n-k)$ bits data. Thus, EE and ER of steganography based on $[n, k]$ linear code are

$$EE = (n-k)/aw, \tag{7}$$

$$ER = (n-k)/n. \tag{8}$$

Here aw denotes the average weight of all the coset leaders.

We will present a matrix embedding scheme using the Golay code [23, 12, 7]. For the Golay code [23, 12, 7], the length of the message is 12, the length of the codewords is 23, and the minimum hamming distance of codewords is 7. The error correcting capacity is 3. The generation matrix of the Golay code [23, 12, 7] is denoted by G,

$$G = (I_{12}, P), \tag{9}$$

and the parity check matrix H is

$$H = \left(P^T, I_{11}\right). \tag{10}$$

Algorithm 1 Coset leaders finding algorithm

Input: parity check matrix H, a codeword $x = (x_1, \cdots, x_{23})$

Output: the cosets table of map between syndromes and coset leaders: Tab.

```
1: for $i = 1$ to 23
2:   $cl \leftarrow (0, \cdots, 0, 1_{(i)}, 0, \cdots, 0)_{23}$
3:   $y \leftarrow x \oplus cl$
4:   $syn \leftarrow H \cdot y^T$
5:   $Tab(syn) \leftarrow cl$
6: end
7: for $i = 1$ to 22
8:   for $j = i + 1$ to 23
9:     $cl \leftarrow (0, \cdots 1_{(i)}, 0, \cdots, 0, 1_{(j)}, 0, \cdots, 0)_{23}$
10:    $y \leftarrow x \oplus cl$
11:    $syn \leftarrow H \cdot y^T$
12:    $Tab(syn) \leftarrow cl$
13:  end
14: end
15: for $i = 1$ to 21
16:   for $j = i + 1$ to 22
17:     for $k = j + 1$ to 23
18:       $cl \leftarrow (0, \cdots 0, 1_{(i)}, 0, \cdots 0, , 1_{(j)}, 0, \cdots 0, 1_{(k)}, 0, \cdots, 0)_{23}$
19:       $y \leftarrow x \oplus cl$
20:       $syn \leftarrow H \cdot y^T$
21:       $Tab(syn) \leftarrow cl$
22:     end
23:   end
24: end
```

where I_t denotes the t-dimensional eye matrix and

$$P = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

The first work is to find the coset leaders of all the syndromes. Because Golay code [23, 12, 7] is a perfect code, the elements of the coset whose weights are no more than 3 are all the coset leaders. We can obtain the cosets of Golay code [23, 12, 7] by Algorithm 1. In Algorithm 1, the vector $(0, \cdots, 1_{(i)}, \cdots, 0)$ denotes that the i-th element is 1 and all the remaining elements are 0. The length of the syndrome table is 2048, we only show partial contents of the coset table in Table 1.

TABLE 1. Partial coset leader of the Golay code [23, 12, 7].

Syndrome	Coset leader	Syndrome	Coset leader
00000000000	00000000000000000000000	10000000010	00000000000010000000010
00000000001	00000000000000000000001	10000000011	00000000000010000000011
00000000010	00000000000000000000010	⋮	⋮
00000000011	00000000000000000000011	11111111000	00000000000101001000000
00000000100	00000000000000000000100	11111111001	01000000000000010001000
00000000101	00000000000000000000101	11111111010	00000101100000000000000
00000000110	00000000000000000000110	11111111011	10000000000010000000100
00000000111	00000000000000000000111	11111111100	00000000001000100100000
⋮	⋮	11111111101	10000000000010000000010
10000000000	00000000000010000000000	11111111110	10000000000010000000001
10000000001	00000000000010000000001	11111111111	10000000000010000000000

Then, the embedding process is shown as follows, where F_2^n denotes the finite field of order 2 and length n.

Step 1: Take 23 pixels in the cover image as a block and get their LSBs. Denote them as the vector x,where $x^T \in F_2^{23}$.

Step 2: Get 11 bits secret data m and compute the syndrome, $syn = H \cdot x + m$, where $m^T, syn^T \in F_2^{11}$.

Step 3: Get the coset leader whose syndrome is syn from the coset leader table Tab, and denote it as, $cl = Tab(syn)$, where $cl^T \in F_2^{23}$.

Step 4: Compute the stego LSB vector, $y = x + cl$, and modify the LSBs of the corresponding pixels according to y.

Step 5: Repeat the above four steps until embedding all the secret messages into the cover image.

The receiver can extract the secret message by computing: $m = Hy$, since,

$$\begin{aligned} H \cdot y + m &= H \cdot (x + cl) + m \\ &= (H \cdot x + m) + H \cdot (cl) \\ &= (0, \cdots, 0)^{11}. \end{aligned}$$

That is $Hy = m$.

Here, we give an example to show the embedding process. Given the cover LSB vector,

$$x = (1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0)^T,$$

and the secret data, $m = (1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0)^T$, then,

$$syn = H \cdot x + m = (1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 11)^T.$$

The coset leader of syndrome $(1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1)^T$ is:

$$cl = (0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1)^T.$$

Then, the stego-vector is

$$y = x + cl = (1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ 1)^T.$$

The receiver extracts the secret data through computing

$$m = H \cdot y = (1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0)^T.$$

Here, we compare the embedding ratio and embedding efficiency of the matrix embedding schemes based on different codes. According to (7) and (8), the *ER* and *EE* can be evaluated with the parameter of the code. Scheme [26] based on [7,4] Hamming code embed 3 bits secret data in each 7 pixels by modifying 1 bit LSB, so $ER = 0.43$. The RM(2,5) code in scheme [28] is equal to [32, 16, 8] linear code, so $ER = 0.5$. The proposed scheme based on the Golay code carries 11bits secret data in each 23 pixels, so the embedding ratio is 0.48. The detail comparing results shown in Table 2. From the comparing results, we know that the matrix embedding using the Golay code [23, 12, 7] can achieve a higher embedding efficiency with proper embedding capacity comparing with other schemes.

TABLE 2. Comparing results of *ER* and *EE* using different codes.

	ER	*EE*	Code length
Filler [24]	0.33	3.02	9
[7,4] Hamming code [26]	0.43	3.43	7
RM(2,5) [28]	0.5	3.16	32
Golay code [23,12,7]	0.48	3.86	23

III. PROPOSED SCHEME

In this section, we will describe our data hiding scheme based on the image encryption algorithm and matrix embedding presented above. The frame of our scheme is shown in Figure 1. In the pre-encoding process, the image owner O modifies the LSBs of pixels, which corrects the LSBs vectors of each block to be codewords of a linear code. Then, O encrypts the fidelity image obtained after the pre-encoding is complete. The data hider H embeds the secret message by the matrix embedding method into the encrypted image. After obtaining the marked encrypted image, the receiver R decrypts it. Then, R extracts the secret data and recovers the fidelity image by decoding. The details of each phase are described as follows.

A. IMAGE PRE-CODING

In the data embedding process based on matrix embedding, the syndrome of cover vectors must be computed, but it is impossible in the encrypted image. To solve the problem,

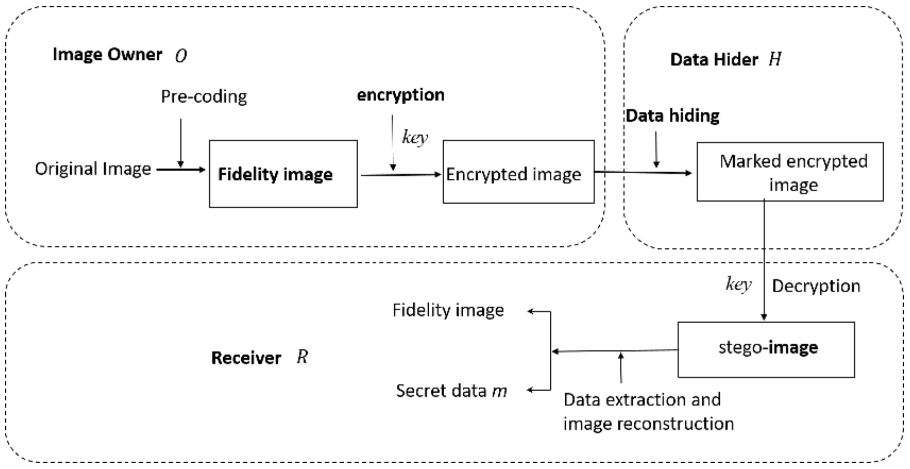
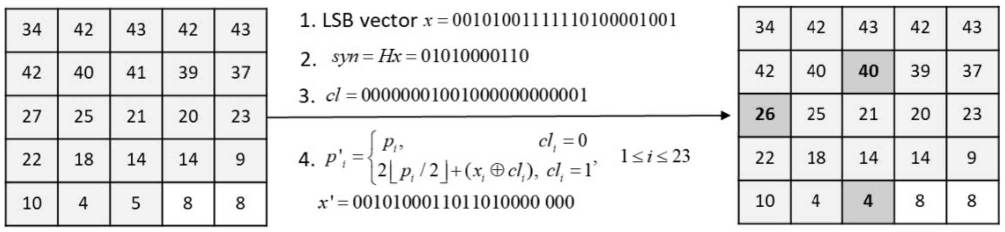


FIGURE 1. **The framework of the proposed scheme.**



34	42	43	42	43
42	40	41	39	37
27	25	21	20	23
22	18	14	14	9
10	4	5	8	8

1. LSB vector $x = 00101001111110100001001$
2. $syn = Hx = 01010000110$
3. $cl = 00000001001000000000001$
4. $p'_i = \begin{cases} p_i, & cl_i = 0 \\ 2\lfloor p_i/2 \rfloor + (x_i \oplus cl_i), & cl_i = 1 \end{cases}, \quad 1 \le i \le 23$

$x' = 00101000110110100000 000$

34	42	43	42	43
42	40	40	39	37
26	25	21	20	23
22	18	14	14	9
10	4	4	8	8

FIGURE 2. **Example of the pre-coding.**

we first correct the cover LSB vectors into codewords of Golay code before encryption, which is called pre-coding.

In the pre-coding process, the image owner O divides the $M \times N$ size original image I into blocks with the size of 1×23. For each block, we apply the Golay code [23, 12, 7] to pre-process their LSBs. The same with Section 2.1, H denotes the parity matrix, and Tab is the coset table of the Golay code [23, 12, 7]. The pre-coding process of a block $p = (p_1, p_2, \cdots, p_{23})$ is described as follows.

Step 1: Get the LSBs of their grayscale values and denote them as the vector $x = (x_1, x_2, \cdots, x_{23})^T$, where $x \in F_2^{23}$.

Step 2: Compute the syndrome of vector x by: $syn = H \cdot x$,where $syn \in F_2^{11}$.

Step 3: Get the coset leader cl, whose syndrome is syn from the coset table, that is $cl = Tab(syn)$, $cl \in F_2^{23}$.

Step 4: Compute the new pixel values

$$p'_i = \begin{cases} p_i, & cl(i) = 0 \\ 2\lfloor p_i/2 \rfloor + (x_i \oplus cl_i), & cl(i) = 1, \ 1 \le i \le 23. \end{cases}$$

Here, the new LSB vector is denoted as x', then

$$x' = x + cl^T, \quad (x')^T \in F_2^{23}.$$

We get the new image I_F when all the blocks have been processed. Because the correct capacity of the Golay code [23, 12, 7] is 3, at most 3 pixels' LSBs are modified in each block. Thus, the pre-coding is a process with low distortion to the original image. That is, we can obtain a fidelity image when all the blocks have been processed. After completing the pre-coding, the LSB vectors of each block in the fidelity image are codewords of the Golay code, that is $H \cdot x' = (0 \cdots 0)_{11}^T$. Figure 2 shows an example of the pre-coding with a sized 5×5 image.

B. I MAGE ENCRYPTION

After the pre-coding, the image owner encrypts the fidelity image using the encryption algorithm presented in Section 2.1. The image owner chooses the scrambling matrix A (the size is 23×23 and $gcd(|A|, 256) = 1$), and generates the encryption key $K = (ek_1, ek_2, \cdots, ek_L)$ using the standard steam key generator with the seed key, which is shared with the receiver. The encryption process is shown as follows.

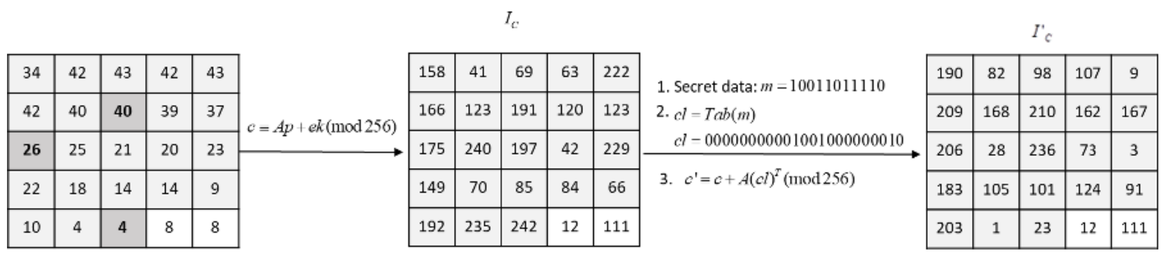
Step 1: Divide the fidelity image I_F into blocks with a size of 1×23.

Step 2: Encrypt the image block by block. For the i-th block p_i $(1 \le i \le \lfloor M \cdot N/23 \rfloor)$, get the grayscale values vector $p_i = (p_{i,1}, p_{i,2}, \cdots, p_{i,23})^T$, obtain the group encryption key $ek_i = (k_{23(i-1)+1}, k_{23(i-1)+2}, \cdots, k_{23i})^T$ from K, and compute:

$$c_i \equiv A \cdot p_i + ek_i (\text{mod} 256). \tag{11}$$

Then, $c_i = (c_{i,1}, c_{i,2}, \cdots, c_{i,23})^T$ is the encryption result.

If the size of the last block is less than 23, we cut the sizes of matrix A and encryption key ek to suit it. When all the blocks have been encrypted, we obtain the encrypted image I_C and send it to H. The purpose of using the matrix operation is to destroy the correlation in the inter-pixel in the encrypted image. So, the matrix A is a public parameter.


I_c
I'_c
$c = Ap + ek \pmod{256}$
1. Secret data: $m = 10011011110$
2. $cl = Tab(m)$
$cl = 00000000001001000000010$
3. $c' = c + A(cl)^T \pmod{256}$

34	42	43	42	43
42	40	40	39	37
26	25	21	20	23
22	18	14	14	9
10	4	4	8	8

158	41	69	63	222
166	123	191	120	123
175	240	197	42	229
149	70	85	84	66
192	235	242	12	111

190	82	98	107	9
209	168	210	162	167
206	28	236	73	3
183	105	101	124	91
203	1	23	12	111

FIGURE 3. **Example of the encryption and embedding process.**

C. DATA EMBEDDING

When receiving the encrypted image I_C, the data hider embeds the secret data into the encrypted image using the matrix embedding based on the Golay code [23, 12, 7]. Assume the secret data $m = (m_1, m_2, \cdots, m_t)$, where $m_i \in \{0, 1\}^{11}$, $1 \le i \le t$, $t \le \lfloor M \cdot N/23 \rfloor$. The data embedding processing contains the four steps as follows.

Step 1: Divide the encrypted image I_C into blocks with a size of 1×23. The secret message m_i will be embedded into the i-th cipher block, $1 \le i \le t$.

Step 2: For the i-th cipher block, get their grayscale values and denote them as the vector $c_i = (c_{i,1}, c_{i,2}, \cdots, c_{i,23})^T$. Take the secret data m_i as the syndrome and get the corresponding coset lead cl_i from the coset table. That is, $cl_i = Tab(m_i)$.

Step 3: Compute the new i-th cipher group:

$$c'_i \equiv c_i + A \cdot (cl_i)^T (\text{mod} 256). \tag{12}$$

Step 4: Repeat Steps 2-3 until all the secret messages are embedded into the cipher image. Lastly, H obtains the marked encrypted image I'_C.

We operate the data hiding processing in the cipher in Step 3, according to the additive homomorphism (4). The corresponding result of (10) in the plain image is:

$$p'_i \equiv p_i + (cl_i)^T (\text{mod} 256). \tag{13}$$

Figure 3 shows an example of the image encryption, and the data embedding continues from Figure 2.

D. IMAGE DECRYPTION

When receiving I'_C, R generates the encryption key stream $K = (ek_1, ek_2, \cdots, ek_L)$, and decrypts the image as follows.

Step 1: Divide the fidelity image I'_C into blocks with length of 23 pixels.

Step 2: Decrypt the image block by block. For the i-th block, get the grayscale values vector $c_i = (c_{i,1}, c_{i,2} \cdots, c_{i,23})^T$, obtain the block decryption key ek_i from K, and compute

$$p'_i \equiv A^{-1} \cdot (c'_i - ek_i)(\text{mod} 256). \tag{14}$$

Then, $p'_i = (p'_{i,1}, p'_{i,2}, \cdots, p'_{i,23})^T$ is the decryption result of $c_i = (c_{i,1}, c_{i,2}, \cdots, c_{i,23})^T$. Here,

$$\begin{aligned} p'_i &\equiv A^{-1} \cdot (c'_i - ek_i) \\ &\equiv A^{-1} \cdot (A \cdot p_i + ek_i - ek_i) + (cl_i)^T \\ &\equiv p_i + (cl_i)^T (\text{mod} 256). \end{aligned}$$

This means that we can obtain a stego-image I_S after direct decryption.

E. DATA EXTRACTION AND IMAGE RECOVERY

After the decryption, the receiver can extract the secret data, and then recover the stego-image to the fidelity image. The process of extraction and recovery are operated as follows.

Step 1: Divide the stego-image I_S into blocks with a size of 1×23. The secret data m_i will be extracted from the i-th block, $1 \le i \le t$.

Step 2: For the i-th block, get their grayscale values and denote them as the vector $p'_i = (p'_{i,1}, p'_{i,2}, \cdots, p'_{i,23})^T$. Then, get the LSBs vector $y_i = (y_{i,1}, y_{i,2}, \cdots, y_{i,23})^T$.Compute, $m_i = H \cdot y_i$, then m_i is the secret data, since

$$\begin{aligned} H \cdot y_i &= H \cdot (x'_i + (cl_i)^T) \\ &= H \cdot x_i + H \cdot (cl_i)^T \\ &= m_i. \end{aligned}$$

Step 3: Get the coset leader cl_i whose syndrome is m_i, and recover the fidelity pixels of the i-th group by computing:

$$p_i \equiv p'_i - (cl_i)^T (\text{mod} 256). \tag{15}$$

Repeat Steps 2-3 until all the secret data are extracted from the stego-image. Finally, we can extract all the secret data $m = (m_1, m_2, \cdots, m_t)$ and recover the stego-image to the fidelity image I_F. Figure 4 shows an example of the decryption, data extraction and image recovery continued from Figures 2 and 3.

IV. EXPERIMENT AND ANALYSIS

In this section, we first analyze the security of the cipher image from the information entropy and the histogram. Then, we give a performance analysis of the proposed scheme. Lastly, we compare the proposed scheme with relevant works on quality of the image.

A. SECURITY ANALYSIS OF CIPHER IMAGE

It is usual to analyze an image encryption algorithm from two aspects: the key sensitivity and the statistical characteristics of the cipher image. The encrypted key in the proposed scheme is generated based on the key generator of the standard stream cipher. Thus, we can ensure the security of the key by using a security key generator. So, here we will only analyze the statistical characteristics of the cipher image by the information entropy and the histogram.

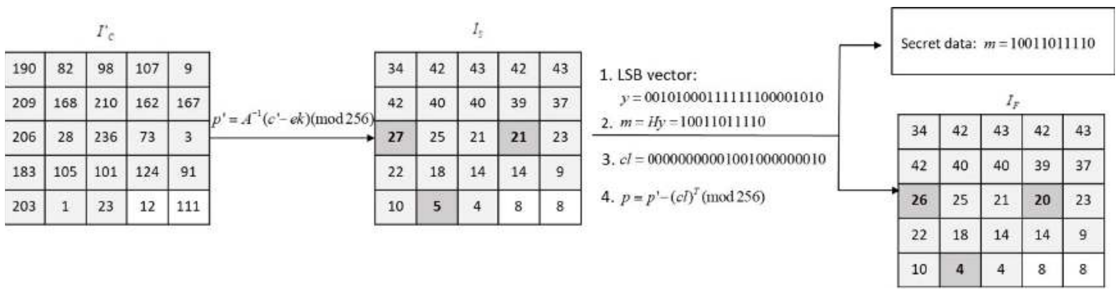

I'_c
I_S
$p' = A^{-1}(c' - ek)(\bmod 256)$
1. LSB vector: $y = 001010001111111100001010$
2. $m = Hy = 10011011110$
3. $cl = 00000000001001000000010$
4. $p = p' - (cl)^T (\bmod 256)$
Secret data: $m = 10011011110$
I_F

FIGURE 4. Example of the image decryption, data extraction and image recovery.

TABLE 3. Information entropy analysis.

Test images	Lena	pepper	baboon	wine	couple	sailboat	lake
Plain images	7.4455	7.5944	7.3579	7.4648	7.0581	7.6321	7.4570
Encrypted images	7.9993	7.9994	7.9994	7.9993	7.9993	7.9993	7.9992
Marked encrypted images	7.9915	7.9913	7.9916	7.9912	7.9915	7.9917	7.9916

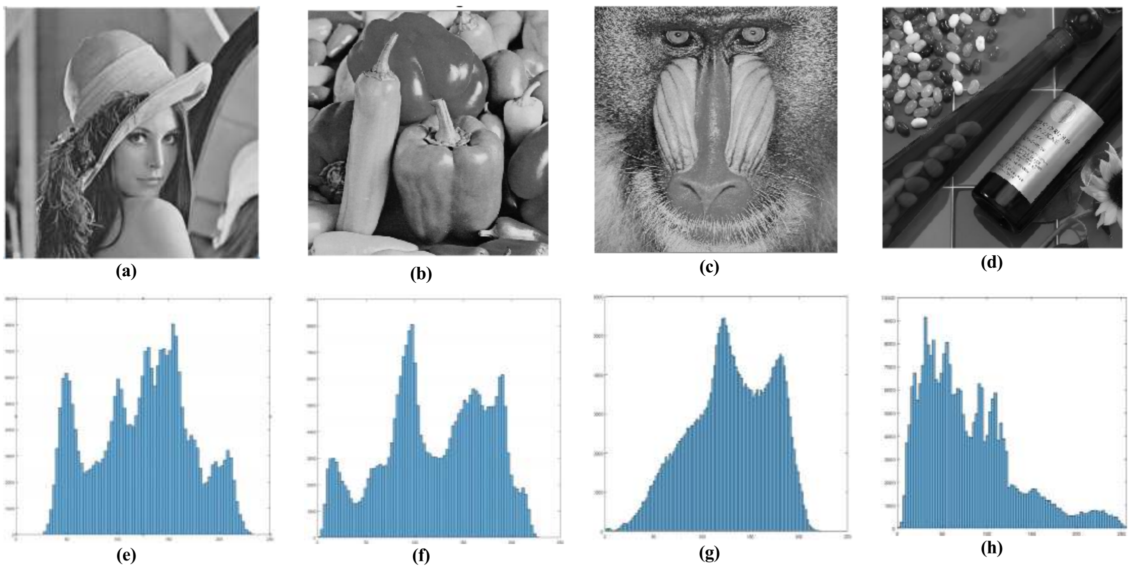


FIGURE 5. Original images and histogram. (a) Lena; (b) Pepper; (c) Baboon. (d) Wine; (e) histogram of (a); (f) histogram of (b); (g) histogram of(c). (h) histogram of (d).

The information entropy of the cipher image is a measurement of randomness, and it can be computed by the follow equation:

$$IE(I) = -\sum_{i=1}^{T} p(I_i)\log_2(p(I_i)), \qquad (16)$$

where $p(I_i)$ is the probability of pixel value I_i, and T is the amount of all pixel values in one image. If I is a random image, then the theoretical value of $IE(I)$ is 8. The experiment results of the test images are shown in Table 3. It can be found that the information entropies of all the encrypted images and the marked encrypted image are close to 8. This means that an attacker is hard to carry out the information entropy attacks to the cipher image.

The histogram reflects the frequency distribution of the pixel values. We take Lena, Pepper, Baboon and Wine as the test images. Figure 5 shows the histograms of the original images, and Figure 6 shows the histograms of the encrypted images. The results show that the corrections of the inter-pixels are destroyed by the encryption operation. The histograms of the cipher images are almost uniform. Thus, the proposed encryption algorithm can resist statistical attacks.

B. PERFORMANCE ANALYSIS

To evaluate the performance of the proposed scheme, we take five grayscale images, Lena, Baboon, Pepper, Airplane and Man, sized 512 × 512, as test images. According to (7), we know the maximum embedding capacity of proposed scheme is 0.4783, and it is the same for all images. So, we focus on the analysis of the visual quality of the stego-image and recovery image.

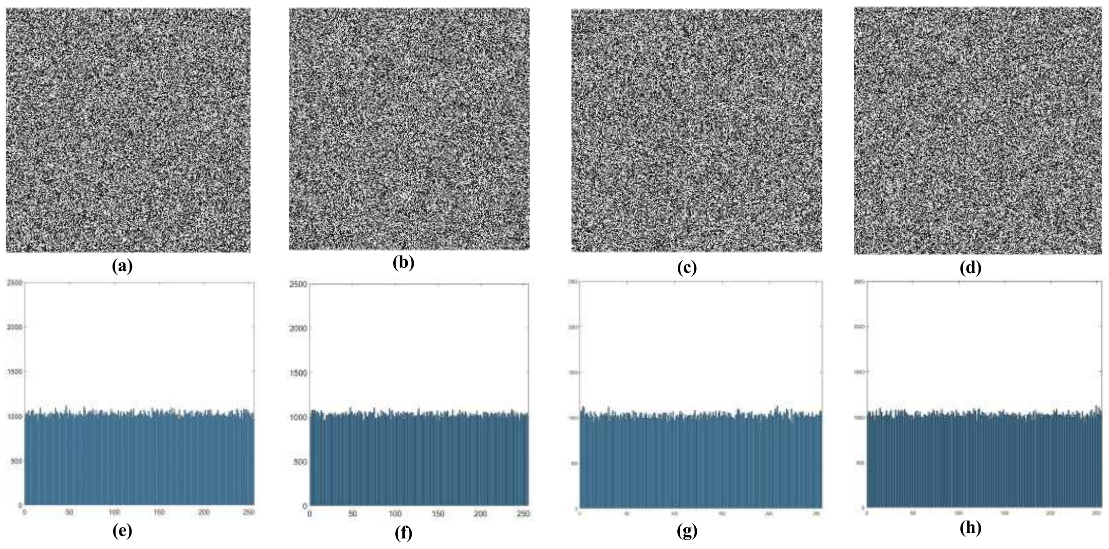


FIGURE 6. Encrypted images and histogram. (a) encrypted Lena; (b) encrypted Pepper; (c) encrypted Baboon; (d) encrypted Wine; (e) histogram of(a); (f) histogram of(b); (g) histogram of(c); (h) histogram of (d).

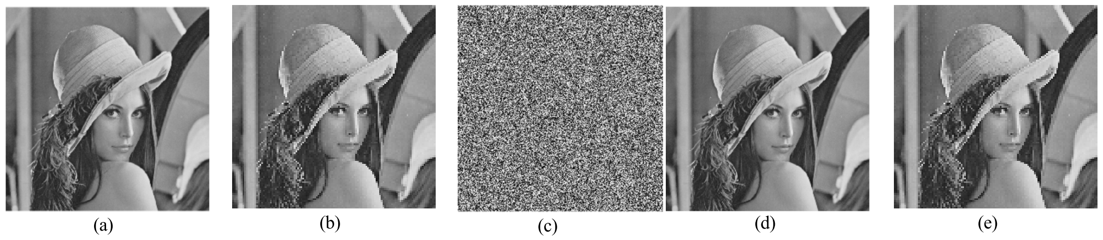


FIGURE 7. Lena image in five stages. (a) original image;(b) fidelity image (PSNR=57.1940 dB); (c) marked encrypted image (payload=0.2bpp); (d) stego-image (PSNR=55.6736 dB); (e) recovery image (PSNR=57.1940 dB).

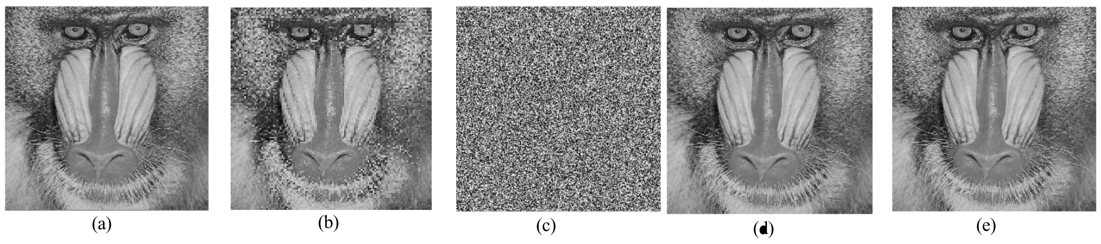


FIGURE 8. Baboon image in five stages. (a) original image;(b) fidelity image (PSNR=57.1955 dB); (c) marked encrypted image (payload=0.2bpp); (d) stego-image (PSNR=55.6622 dB); (e) recovery image (PSNR=57.1955 dB).

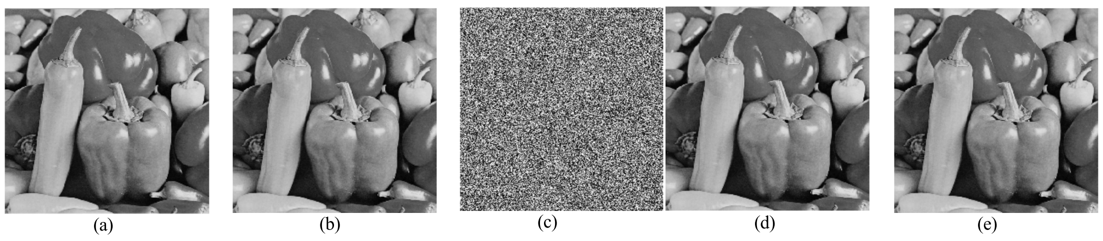


FIGURE 9. Baboon image in five stages. (a) original image;(b) fidelity image (PSNR=57.2015 dB); (c) marked encrypted image (payload=0.2bpp). (d) stego-image (PSNR=55.6800 dB). (e) recovery image (PSNR=57.2015 dB).

Figures 7-10 show the test images in five stages in the proposed scheme. The average PSNR of the pre-coded images is 57.19 dB, which is the fidelity image of the original images. In decryption stage, it obtains the stego-image by direct decryption. The results show that the average PSNR value of the stego-images is about 55.67 dB with a payload of 0.2bpp. Finally, we can recover the fidelity image.

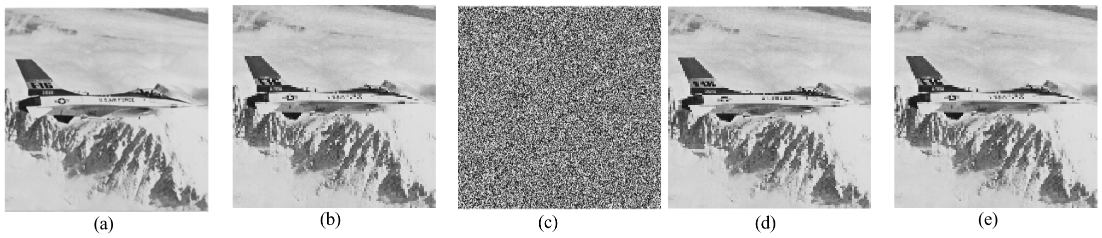


FIGURE 10. Baboon image in five stages. (a) original image;(b) fidelity image(PSNR=57.1983); (c) marked encrypted image (payload=0.2bpp); (d) stego-image (PSNR=55.6832 dB); (e) recovery image (PSNR=57.1983 dB).

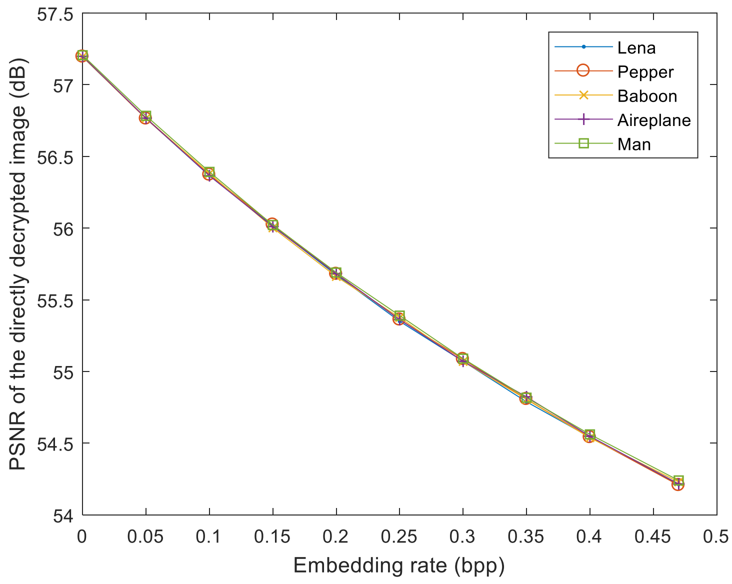


FIGURE 11. The PSNR of the directly decrypted image (stego-image).

Figure 11 shows the PSNR values with the increasing embedding rate. We can observe from the results that the PSNR values of the stego-images can maintain a high level with different embedding rates for all the test images. When the embedding rate reaches the maximum values, the PSNR of the stego-image is about 54.2dB. To further verify the PSNR of the proposed scheme, we take 500 images in BOWS_2 as test images, and the experiment results are shown in Figure 12. The average PSNRs of the recovery images and stego-images are 57.21dB and 54.37dB. We can conclude that the image qualities of stego-images and recovery images of the propose scheme are quite high.

C. COMPARISON ANALYSIS

The advantage of matrix embedding steganography is high embedding efficiency, which means that we can obtain a high quality stego-image. We have compared the embedding efficiency with other relevant matrix embedding steganography methods in Section 2.1. Now, we compare it with other data hiding schemes in encrypted image [15]–[17] on the PSNR of a directly decrypted image. Li *et al.* [15] embedded the secret by compressing the LSBs of the encrypted image, which is also a fidelity preserved scheme. Here, we choose the parameters $T = 30, u = 3$ for scheme [15]. Chen *et al.* [16] and Zhang *et al.* [17] implemented the data hiding in the encrypted images based on homomorphic encryption, and the embedding capacity is closer to the proposed scheme.

Figure 13 shows the comparison results in four test images sized 512×512. The experimental results show that stego-image quality of is higher than other schemes in all the four test images. In the case of less payload, all Chen *et al.* [16], Zhang *et al.* [17] and the proposed scheme have higher stego-image quality. With the increasing of the payload, the advantage of the proposed scheme become more and more obvious. Meanwhile, the decline of stego-image quality with the increasing payload is not significant in the proposed scheme, but it is yes in the others schemes especially in Chen *et al.* [16] and Zhang *et al.* [17]. When the payload reaching the maximum, the PSNR of the proposed scheme is more than 54dB, while the PSNR of other schemes is less than 42dB. From the experimental results, we can conclude that the proposed scheme has better embedding efficiency than other schemes because of using the matrix embedding method.

Then, we further compare our scheme with Li *et al.* [15], in which the recovery image is also a fidelity image, on the quality of the image and embedding capacity. Table 4 shows

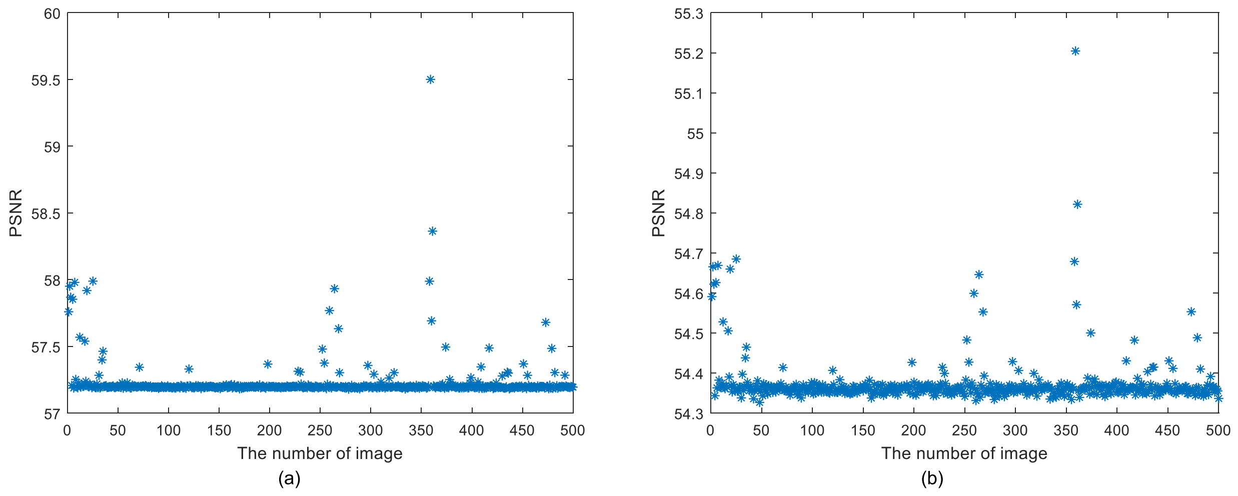


FIGURE 12. Experiment results in 500 images. (a) PSNR (dB) of recovery image; (b). PSNR (dB) of the directly decrypted images with max payload.

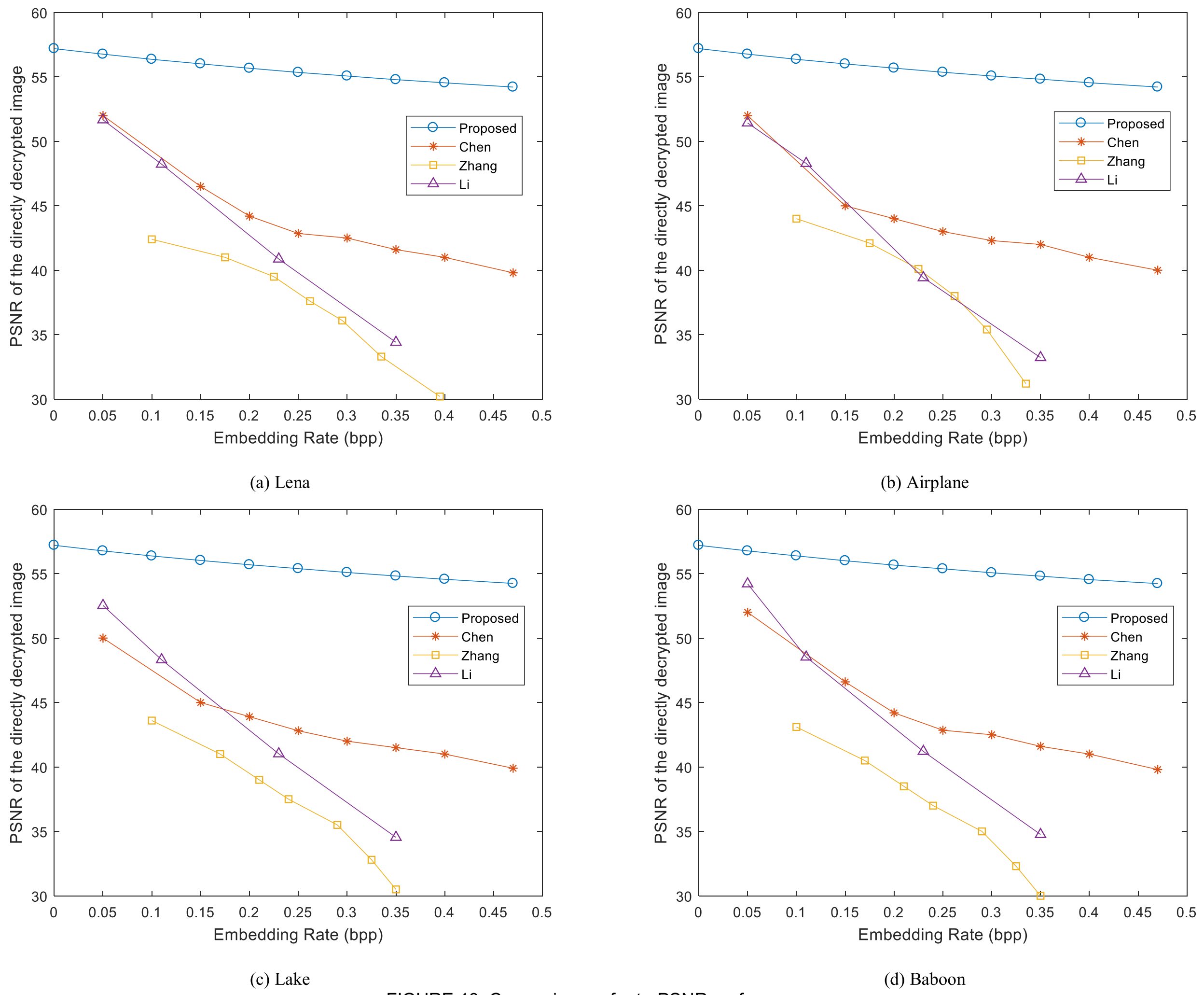


FIGURE 13. Comparisons of rate-PSNR performance.

FIGURE 13. Comparisons of rate-PSNR performance.

the comparison results of our scheme with scheme [15]. From the experimental results, we can observe that our scheme has a higher embedding capacity and better visual quality for the directly decrypted image and recovery image.

TABLE 4. Comparison results in the embedding capacity, PSNRs of stego-images and recovery images with max payload.

	Li[15] ($u=1, T=30, \beta=0.01$)			Proposed scheme		
	EC	PSNR	PSNR	EC	PSNR	PSNR
Lena	0.1195	48.23	51.33	0.4783	54.21	57.19
Baboon	0.1024	48.53	51.98	0.4783	54.23	57.19
Airplane	0.1158	48.29	51.43	0.4783	54.22	57.20
Man	0.1154	48.30	51.45	0.4783	54.24	57.21
Lake	0.1121	48.32	51.57	0.4783	54.22	57.20

V. CONCLUSION

This paper proposed a fidelity preserving data hiding scheme in encrypted images. We apply the matrix embedding steganography using the Golay code [23, 12, 7], which has a good embedding efficiency and high embedding capacity, to improve quality of the stego-image. We first corrected the LSB vectors group of the original image into codewords of the Golay code [23, 12, 7] before encryption. Then, we encrypted the image using a homomorphic encryption algorithm based on the stream cipher. Lastly, we embedded the secret message into the cipher image using the homomorphic evaluation. The results of the experiment and analysis shows that our scheme has a good security performance in cipher image. The receiver can obtain a high quality stego-image by directly decrypting the cipher image, and recover the stego-image into a fidelity image after extracting the secret message.

REFERENCES

[1] C.-K. Chan and L. Cheng, "Hiding data in images by simple LSB substitution," *Pattern Recognit.*, vol. 37, no. 3, pp. 469–474, Mar. 2004.

[2] M. Wu and B. Liu, "Data hiding in binary image for authentication and annotation," *IEEE Trans. Multimedia*, vol. 6, no. 4, pp. 528–538, Aug. 2004.

[3] Q. Ying, Z. Qian, X. Zhang and D. Ye, "Reversible data hiding with image enhancement using histogram shifting," *IEEE Access*, vol. 7, pp. 46506–46521, 2019.

[4] Y.-H. Yu, C.-C. Chang, and Y.-C. Hu, "A steganographic method for hiding data in VQ encoded images," in *Proc. Int. Symp. Intell. Multimedia, Video Speech Process.*, Jun. 2005, pp. 358–361.

[5] W. Hong, T.-S. Chen, and C.-W. Shiu, "Lossless steganography for AMBTC-compressed images," in *Proc. Congr. Image Signal Process.*, 2008, pp. 13–17.

[6] C. Qin, C.-C. Chang, and Y.-P. Chiu, "A novel joint data-hiding and compression scheme based on SMVQ and image inpainting," *IEEE Trans. Image Process.*, vol. 23, no. 3, pp. 969–978, Mar. 2014.

[7] 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.

[8] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted images," *Signal Process.*, vol. 94, pp. 118–127, Jan. 2014.

[9] T. Mathew and M. Wilscy, "Reversible data hiding in encrypted images by active block exchange and room reservation," in *Proc. Int. Conf. Contemp. Comput. Inform. (IC3I)*, Nov. 2014, pp. 839–844.

[10] S. Yi and Y. Zhou, "An improved reversible data hiding in encrypted images," in *Proc. 2015 IEEE China Summit Int. Conf. Signal Inf. Process. (ChinaSIP)*, Jul. 2015, pp. 225–229.

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

[12] X. Liao, K. Li, and J. Yin, "Separable data hiding in encrypted image based on compressive sensing and discrete fourier transform," *Multimed Tools Appl.*, vol. 76, no. 20, pp. 20739–20753, Oct. 2017.

[13] H. Ren, S. Niu, and X. Wang, "Reversible data hiding in encrypted images using POB number system," *IEEE Access*, vol. 7, pp. 149527–149541, 2019.

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

[15] M. Li, L. Wang, J. Fan, Y. Zhang, K. Zhou, and H. Fan, "Fidelity preserved data hiding in encrypted highly autocorrelated data based on homomorphism and compressive sensing," *IEEE Access*, vol. 7, pp. 69808–69825, 2019.

[16] 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.

[17] 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.

[18] 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.

[19] S. Zheng, Y. Wang, and D. Hu, "Lossless data hiding based on homomorphic cryptosystem," *IEEE Trans. Dependable Secure Comput.*, to be published, doi: 10.1109/tdsc.2019.2913422.

[20] Y. C. Chen, T. H. Hung, S. H. Hsieh, and C. W. Shiu, "A new reversible data hiding in encrypted image based on multi-secret and lightweight cryptographic algorithms," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 12, pp. 3332–3343, Dec., 2019.

[21] R. Crandall. (1998). *Some Notes on Steganography' Posted on Steganography Mailing List*. [Online]. [Online]. Available: http://os.inf.tu-dresden.de/ westfeld/crandall.pdf

[22] J. Fridrich, D. Soukal, "Matrix embedding for large payloads," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 3, pp. 390–395, Sep. 2006.

[23] A. Sarkar, U. Madhow, and B. S. Manjunath, "Matrix embedding with pseudorandom coefficient selection and error correction for robust and secure steganography," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 2, pp. 225–239, Jun. 2010.

[24] T. Filler, J. Judas, and J. Fridrich, "Minimizing additive distortion in steganography using syndrome-trellis codes," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 920–935, Sep. 2011.

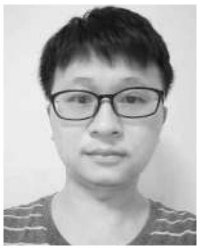
[25] C.-C. Chang, T. D. Kieu, and Y.-C. Chou, "A high payload steganographic scheme based on (7, 4) Hamming code for digital images," in *Proc. Int. Symp. Electron. Commerce Secur.*, Guangzhou, China, Aug. 2008, pp. 16–21.

[26] Q. Mao, "A fast algorithm for matrix embedding steganography," *Digit. Signal Process.*, vol. 25, pp. 248–254, Feb. 2014.

[27] X. Li, S. Cai, W. Zhang, and B. Yang, "A further study of large payloads matrix embedding," *Inf. Sci.*, vol. 324, pp. 257–269, Dec. 2015.

[28] T. Yang and H. Chen, "Matrix embedding in steganography with binary Reed-Muller codes," *IET Image Processing*, vol. 11, no. 7, pp. 522–529, Jul., 2017.

[29] F. Khelifi, "On the security of a stream cipher in reversible data hiding schemes operating in the encrypted domain," *Signal Process.*, vol. 143, pp. 336–345, Feb. 2018.



SISHENG CHEN received the B.S. degree from Shandong University, in 2005, and the M.S. degree in applied mathematics from Fujian Normal University, in 2008. He is currently pursuing the Ph.D. degree in information engineering and computer science with Feng Chia University, Taichung, Taiwan. He is also a Lecturer with the Fuqing Branch of Fujian Normal University. His research interests include multimedia security and network security.

CHIN-CHEN CHANG (Fellow, IEEE) received the B.Sc. degree in applied mathematics and the M.Sc. degree in computer and decision sciences from National Tsing Hua University, and the Ph.D. degree in computer engineering from National Chiao Tung University. He was with National Chung Cheng University, from 1989 to 2005. He has been the Chair Professor with the Department of Information Engineering and Computer Science, Feng Chia University, since February 2005. Prior to joining Feng Chia University, he was an Associate Professor with Chiao Tung University, a Professor with National Chung Hsing University, and the Chair Professor with National Chung Cheng University. He was a Visiting Researcher with Tokyo University and a Visiting Scientist with Kyoto University, Japan. He has served as the Chairman of the Institute of Computer Science and Information Engineering, the Dean of the College of Engineering, a Provost and then the Acting President of Chung Cheng University, and the Director of Advisory Office, Ministry of Education, Taiwan. His current research interests include database design, computer cryptography, image compression, and data structures. He is also a Fellow of IEE, U.K. He was a recipient of many research awards and honorary positions by and in prestigious organizations, including nationally and internationally. Since his early years of career development, he consecutively won the Outstanding Talent in Information Sciences of the R. O. C., the AceR Dragon Award of the Ten Most Outstanding Talents, the Outstanding Scholar Award of the R. O. C., the Outstanding Engineering Professor Award of the R. O. C., Distinguished Research Awards of National Science Council of the R. O. C., and the Top Fifteen Scholars in Systems and Software Engineering of the *Journal of Systems and Software*. On numerous occasions, he was invited to serve as a Visiting Professor, the Chair Professor, an Honorary Professor, the Honorary Director, the Honorary Chairman, a Distinguished Alumnus, a Distinguished Researcher, and a Research Fellow by universities and research institutes.

QUNYING LIAO received the Ph.D. degree from the College of Mathematics, Sichuan University. She is currently a Professor with the School of Mathematics Science, Sichuan Normal University, Chengdu, China. Her research interests include finite fields, cryptography, coding theory, and their applications.