Improved Joint Reversible Data Hiding in Encrypted Images

Zhenxing Qian^{1,2}, Shu Dai¹, Fei Jiang¹, Xinpeng Zhang^{1,3}

1. School of Communication and Information Engineering, Shanghai University, Shanghai, China

2. Nanjing University of Information Science Technology (NUIST), Nanjing, China

3. Key Laboratory of Specialty Fiber Optics and Optical Access, Shanghai University, Shanghai, China.

E-mail: zxqian@shu.edu.cn

Abstract—This paper proposes an improved method of reversible data hiding in encrypted images (RDH-EI). Three parties constitute the proposed system: the image owner, the remote server and the recipient. To preserve privacy, an image owner encrypts the original image using a stream cipher algorithm and uploads the ciphertext to a remote server. On server side, a data-hider is allowed to embed additional message into the encrypted image using a swapping/shifting based algorithm. After downloading the marked encrypted image from the server and implementing the decryption, a recipient can extract the hidden messages and losslessly recover the original image. Experimental results show that the proposed method achieves a larger payload than the related works. Meanwhile, a limitation in the related works that few bits can be embedded into the encrypted medical images is also eliminated in the proposed method.

Index Terms—Reversible data hiding, image encryption, image recovery

1. Introduction

Reversible data hiding (RDH) is a technique to embed additional message into a cover media, such as military or medical images, using a reversible manner such that the original cover content can be perfectly restored after the extraction of hidden messages [1][2]. Reversible data hiding in encrypted images (RDH-EI) is a new topic of reversible data hiding [3]. This technique allows a service provider to embed additional messages into encrypted images without accessing the original contents, and guarantees that the original images can be losslessly recovered on the recipient side. RDH-EI technique can be used in many applications [4-12]. For example in medical systems, the medical images can be encrypted before uploading to a server if a patient does not allow his/her privacy to be revealed to outsiders. Meanwhile, for a better management, the database administrator can embed the medical records or the patient's information into the encrypted image. This way, the storage payload can be saved, and the profile management is more convenient. On the other hand, when a doctor downloads the encrypted images containing additional information from the medical server, he/she can extract the patient's information exactly and recover the original medical images for diagnosis without any error.

Some works have been done in the field of RDH-EI. Generally, there are two kinds of RDH-EI approaches: the *joint* RDH-EI and the *separable* RDH-EI. In the former technique, hidden messages are extracted from the marked encrypted image by the user who has the decryption key, which is realized together with image recovery. While in the latter technique, hidden messages can be extracted by the one who has not the decryption key, in which data extraction and image recovery are separated. Compared with *separable* RDH-EI methods, *joint* RDH-EI methods always estimate the spatial correlation of pixels inside the original image. As a result, the marked image should be decrypted before extracting the hidden message and recovering the original contents.

RDH-EI was first proposed in [3], in which a content owner encrypts the original image using a stream cipher algorithm, and a data hider embeds additional message into encrypted blocks by flipping

three least significant bits (LSB) of half the pixels in each block. When extracting the additional message, a recipient decrypts the enciphered image and generates two candidates for each block by flipping the LSB layers again. Since the original block is much smoother than the interfered block, the embedded bits can be extracted and the original image can be perfectly recovered. In this joint RDH-EI, errors of data extraction and image recovery can be avoided, as long as the block size is large enough. This method was improved in [4] by exploiting spatial correlation between neighboring blocks and using a side-match algorithm to achieve a better embedding rate. The flipping based idea was further improved in [5], making use of the multiple neighboring pixels in different locations, which can efficiently reduce the error rate of extraction and recovery.

Separable RDH-EI method was first proposed in [6]. The data hider divides the encrypted pixels into segments, and compresses some LSB layers of each segment into a less amount of bits using a pseudo-randomly generated matrix. As a result, spare room in each segment is vacated for data embedding. On the receiver side, the recipient estimates the LSB bits using the most significant bits (MSB) of neighboring pixels, and recovers the original content by comparing the estimated bits with the extracted vectors. Since the additional message can be extracted from the encrypted images, data extraction and image recovery are therefore separable. If higher bitplanes are appropriately used [7], better embedding rate can be achieved. In [8], the data-hider compresses MSB bits of an encrypted image using LDPC based Slepian-Wolf encoding, and the recipient recovers original content by DSC decoding. This method illustrates that RDH-EI is a distributed source coding (DSC) problem. This method is also improved in [16] by a progressive recovery based RDH-EI algorithm.

With different ideas, methods in [9] [10] and [11] create embedding room in the plaintext image by embedding certain contents into other pixels using traditional RDH method. The pre-processed image is then encrypted by the owner to generate an encrypted image. Thus, vacated rooms in the encrypted image can be used by the data-hider to hide message. Besides, RDH-EI can also be realized using public key cryptosystem [12] and [13]. In [13], with a two-class SVM classifier, a recipient distinguishes the encrypted and the non-encrypted image patterns to jointly extract additional message and recover the original image. This method provides a larger embedding rate than previous ones. RDH-EI for the encrypted JPEG bitstream was also proposed in [14], which encrypts JPEG bitstream into a format compliant structure and embeds additional message into the encrypted copy by slightly modifying the stream. In addition, RDH is applied in encrypted video systems [15].

In this paper, we focus on joint reversible data hiding in encrypted images, and propose an improved algorithm. We first introduce the related works and analyze their limitations in Section 2. Section 3 and Section 4 present the procedures of image encryption and data embedding, respectively. Data extraction and image recovery are elaborated in Section 5. Section 6 provides the experimental results, and Section 7 concludes the paper.

2. Related Works

In this section, joint RDH-EI methods in [3]~[5] are briefly introduced and analyzed. Framework of these works is depicted in Fig. 1. The original image is encrypted by the content owner and uploaded to a remote server. On the server side, a data-hider embeds additional messages into the ciphertext to construct a marked encrypted image. After a user/recipient downloads the marked copy from the server, he/she can decrypt it to an approximate image with good quality, and can further extract the additional message and recover the original content using a combined extraction and recovery operation.

Fig. 1 General Framework of joint RDH-EI

In [3]~[5], the same encryption and embedding algorithms are used. An image is encrypted by the stream cipher algorithm to generate an encrypted image. When hiding messages, the encrypted image is first divided into non-overlapped blocks. The encrypted pixels in each block are separated into two groups. If a bit “1” is to be embedded into the block, three LSB layers of the first group are flipped. Otherwise, three LSB layers of the second group are flipped. On the recipient side, after decrypting the marked copy to an approximate image, he/she divides the image into blocks again and flips three LSB layers of each group respectively to construct two candidate blocks. Main difference among these methods is the definition of the *fluctuation function*, which is used to determine which candidate is the original content. A better defined fluctuation function can result in the decreased block size, and thus more bits can be accommodated in the encrypted image.

In [3], the fluctuation function is defined in (1), in which $s_1 \times s_2$ stands for the block size, p the pixel value in a candidate block, and f the fluctuation value of the block. A smaller value of f represents a smoother block.

$$f = \sum_{u=2}^{s_1-1} \sum_{v=2}^{s_2-1} \left| p_{u,v} - \frac{p_{u-1,v} + p_{u,v-1} + p_{u+1,v} + p_{u,v+1}}{4} \right| \quad (1)$$

This function is modified in [4] by (2).

$$f = \sum_{u=1}^{s_2} \sum_{v=1}^{s_1-1} |p_{u,v} - p_{u,v+1}| + \sum_{u=1}^{s_2-1} \sum_{v=1}^{s_1} |p_{u,v} - p_{u+1,v}| \quad (2)$$

As an improved algorithm, the method in [5] uses the same function as (2) along with an additional operation on the boundary pixels, as shown in (3).

$$\begin{aligned} f = & \left| p_{1,1} - \frac{p_{1,2} + p_{2,1}}{2} \right| + \left| p_{1,s_2} - \frac{p_{1,s_2-1} + p_{2,s_2}}{2} \right| + \left| p_{s_1,1} - \frac{p_{s_1,2} + p_{s_1-1,1}}{2} \right| + \left| p_{s_1,s_2} - \frac{p_{s_1,s_2-1} + p_{s_1-1,s_2}}{2} \right| \\ & + \sum_{v=2}^{s_2-1} \left| \left(p_{1,v} - \frac{p_{1,v-1} + p_{1,v+1} + p_{2,v}}{3} \right) + \left(p_{s_1,v} - \frac{p_{s_1,v-1} + p_{s_1,v+1} + p_{s_1-1,v}}{3} \right) \right| \\ & + \sum_{u=2}^{s_1-1} \left| \left(p_{u,1} - \frac{p_{u-1,1} + p_{u+1,1} + p_{u,2}}{3} \right) + \left(p_{u,s_2} - \frac{p_{u-1,s_2} + p_{u+1,s_2} + p_{u,s_2-1}}{3} \right) \right| \\ & + \sum_{u=2}^{s_1-1} \sum_{v=2}^{s_2-1} \left| p_{u,v} - \frac{p_{u-1,v} + p_{u,v-1} + p_{u+1,v} + p_{u,v+1}}{4} \right| \end{aligned} \quad (3)$$

Although these methods are efficacy for hiding messages into encrypted images, we found that there is a limitation inside them. When a large area of pixels share the unified values in an image, *e.g.*, a medical image, it would be difficult to losslessly recover the original content unless the embedding rate is close to zero. An example in Fig. 2 shows the reason, in which (a) is a medical image, in which a large area of pixels has the same values. One block in (a) is enlarged in Fig. (b). After encrypting the image and uploading the encrypted image onto a server, a data hider divides the encrypted copy into blocks and embeds one additional bit into each by flipping three LSB bits of half pixels. During recovery, the marked encrypted image is decrypted and then divided into blocks. The block containing an additional bit corresponding to (b) is shown in (c). By flipping the LSB bits of half pixels, two candidate blocks are generated in (d) and (e). With the methods in [3]~[5], the recipient is difficult to identify which candidate is the original block, since the calculation results of both candidates using aforementioned fluctuation functions are the same.

Fig. 2 An example of RDH-EI for a medical image, (a) is the original medical image, (b) a block of the image, (c) the decrypted block containing an additional bit, (d) and (e) two candidate blocks for recovery

To eliminate this limitation, this paper proposes a new joint RDH-EI method. The proposed system shares the same framework as the general framework depicted in Fig. 1. In the module of data hiding, the algorithms of data swapping and cyclic shifting are used to embed the additional messages. Accordingly, the reverse operations are used to extract the messages and recover the original image in the extraction and recovery module.

3. Image Encryption and Decryption

Without loss of generality, we assume that the original image $\mathbf{O}$ is a grayscale image with all pixel values ranging from 0 to 255, and the image size is $M \times N$. The image owner first decomposes each pixel value into 8 bits by

$$b_{i,j,u} = \lfloor O_{i,j} / 2^u \rfloor \bmod 2, \quad u = 0, 1, 2, \dots, 7 \quad (4)$$

where $O_{i,j}$ stands for the pixel values of the original image, $\lfloor \cdot \rfloor$ the rounding down operator, $1 \leq i \leq M$, and $1 \leq j \leq N$.

After selecting an encryption key $\mathbf{K}$, the owner generates pseudo-random bits using a stream cipher function, such as the algorithm of RC4 or SEAL. The original image is then encrypted by

$$e_{i,j,u} = b_{i,j,u} \oplus k_{i,j,u}, \quad u = 0, 1, 2, \dots, 7 \quad (5)$$

where $k_{i,j,u}$ is the key stream, $e_{i,j,u}$ the generated ciphertext, and $\oplus$ the exclusive-OR operation. Thus, the encrypted image $\mathbf{E}$ is constructed by

$$E_{i,j} = \sum_{u=0}^7 e_{i,j,u} \cdot 2^u \quad (6)$$

where $E_{i,j}$ represents the encrypted pixel value, $1 \leq i \leq M$, and $1 \leq j \leq N$. The image owner uploads the encrypted image $\mathbf{E}$ onto the remote server.

Procedure of image decryption is the same as encryption. Given an encrypted image $\mathbf{E}$ and the decryption key $\mathbf{K}$, the original image can be constructed using the same way as (4)~(6).

4. Data Embedding

On the remote server, the data hider embeds additional messages into the encrypted image. He/she first divides the encrypted image $\mathbf{E}$ into four down-sampled images. We denote these sub-images as $\mathbf{E}_1$, $\mathbf{E}_2$, $\mathbf{E}_3$ and $\mathbf{E}_4$. As shown in Fig. 3, the sub-image $\mathbf{E}_1$ is labeled by “ $\diamond$ ”, $\mathbf{E}_2$ by “ $\circ$ ”, $\mathbf{E}_3$ by “ $\square$ ”, and $\mathbf{E}_4$ by “ Δ ”.

Fig. 3 Image Down-sampling

4.1 First Round Embedding

The data hider uses a cyclic-shifting based data hiding algorithm to embed messages into $\mathbf{E}_2$, $\mathbf{E}_3$ and $\mathbf{E}_4$. Three LSB layers are extracted from $\mathbf{E}_2$, $\mathbf{E}_3$ and $\mathbf{E}_4$ by

$$J_k(i, j) = E_k(i, j) \bmod 8, \quad k = 2, 3, 4 \quad (7)$$

where $1 \leq i \leq M$ and $1 \leq j \leq N$. Divide all elements in $\mathbf{J}_k$ ($k=2,3,4$) into L segments $\{\mathbf{S}_1, \mathbf{S}_2, \dots, \mathbf{S}_L\}$. Let each segment be

$$\mathbf{S}_m = (s_{m,1}, s_{m,2}, \dots, s_{m,a})$$

where $m=1,2,\dots,L$, $a = \lfloor 3/4 \cdot (MN/L) \rfloor$.

By shifting the elements in each segment $\mathbf{S}_m$, $\lfloor \log_2 a \rfloor$ bits are embedded into the segment. For example, when embedding the number t into $\mathbf{S}_m$, the data hider modifies the segment into

$$\mathbf{S}_m' = (s_{m,t+1}, \dots, s_{m,a}, s_{m,1}, \dots, s_{m,t})$$

where $0 \leq t \leq 2^{\lfloor \log_2 a \rfloor} - 1$.

After embedding, the data hider replaces the original data in three LSB layers in $\mathbf{E}_2$, $\mathbf{E}_3$ and $\mathbf{E}_4$ with the modified data $\{\mathbf{S}_1', \mathbf{S}_2', \dots, \mathbf{S}_L'\}$, and generates the marked encrypted sub-images $\mathbf{E}_2'$, $\mathbf{E}_3'$ and $\mathbf{E}_4'$. Therefore, payload (*bits*) of the first round embedding is equal to

$$C_1 = L \cdot \lfloor \log_2 a \rfloor$$

4.2 Second Round Embedding

The data hider further hides additional bits into the sub-image $\mathbf{E}_1$ using a data-swapping based embedding algorithm. Three LSB layers of the least significant bits from $\mathbf{E}_1$ by

$$J_1(i, j) = E_1(i, j) \bmod 8$$

where $1 \leq i \leq M$ and $1 \leq j \leq N$. Divide $\mathbf{J}_1$ into a set of blocks $\{\mathbf{W}_1, \mathbf{W}_2, \dots, \mathbf{W}_R\}$, where $R = \lfloor MN/4r^2 \rfloor$ and $r \times r$ is the size of each block. Several swapping steps are then used to do the embedding.

In the first step, if the data hider wants to embed a number r , $0 \leq r \leq 2^{\lfloor \log_2 R-1 \rfloor} - 1$, he/she swaps the order of $\mathbf{W}_1$ and $\mathbf{W}_{r+2}$ and put the swapped blocks back to the corresponding LSB layers in $\mathbf{E}_1$. Remove the blocks $\mathbf{W}_1$ and $\mathbf{W}_{r+2}$ from the set $\{\mathbf{W}_1, \mathbf{W}_2, \dots, \mathbf{W}_R\}$, and rank the remaining blocks as $\{\mathbf{U}_1, \mathbf{U}_2, \dots, \mathbf{U}_{R-2}\}$.

In the second step, when embedding a number q , $0 \leq q \leq 2^{\lfloor \log_2 R-3 \rfloor} - 1$, the data-hider swaps the orders of $\mathbf{U}_1$ and $\mathbf{U}_{q+2}$, and put the swapped blocks back to the LSB layers. Using the same way, the data hider embeds a number not larger than $2^{\lfloor \log_2 R-2i-1 \rfloor} - 1$ in the $(i+1)$ -th step. Repeat the embedding until $R/2$ blocks have been swapped to accommodate additional message. The data hider replaces the original data of three LSB layers in $\mathbf{E}_1$ with the swapped blocks to construct the marked encrypted images $\mathbf{E}_1'$. As a result, payload (*bits*) of the first round embedding is equal to

$$C_2 = \sum_{i=0}^{\lfloor R/4 \rfloor} \lfloor \log_2 (R-1-2i) \rfloor$$

After cyclic shifting and data swapping, the data hider merges $\mathbf{E}_1'$, $\mathbf{E}_2'$, $\mathbf{E}_3'$ and $\mathbf{E}_4'$ together to construct a marked encrypted image $\mathbf{E}'$. The embedding payload is therefore equal to

$$C = C_1 + C_2 = L \cdot \lfloor \log_2 a \rfloor + \sum_{i=0}^{\lfloor R/4 \rfloor} \lfloor \log_2 (R-1-2i) \rfloor. \quad (8)$$

5. Data Extraction and Image Recovery

On the recipient side, a receiver can generate an approximate image by decrypting the marked encrypted image $\mathbf{E}'$. Image decryption is a reverse procedure of the encryption. With the decryption key, pseudo-random bits are generated using the stream cipher function. Each bit of the marked encrypted image is then decrypted using the same algorithm depicted in Section 3. Since only three LSB layers are different from the original image, the decrypted image preserves good quality.

When extracting the additional message and recovering the original image, the receiver first down-samples the marked encrypted image $\mathbf{E}'$ into four sub-images, $\mathbf{E}_1'$, $\mathbf{E}_2'$, $\mathbf{E}_3'$ and $\mathbf{E}_4'$, using the same algorithm depicted in Fig. 3. The receiver uses following to rounds to recover the original image.

5.1 First Round Recovery

The receiver first implements the reverse operations of data swapping, during which one part of additional messages can be extracted and the original content corresponding to $\mathbf{E}_1'$ can be recovered. The receiver extracts three layers of LSB from $\mathbf{E}_1'$ by

$$J_1'(i, j) = E_1'(i, j) \bmod 8$$

where $1 \leq i \leq M$ and $1 \leq j \leq N$. Divide $\mathbf{J}_1'$ into R blocks $\{\mathbf{W}_1', \mathbf{W}_2', \dots, \mathbf{W}_R'\}$ with each sized $r \times r$, where $R = \lfloor MN/4r^2 \rfloor$. Meanwhile, the receiver removes three LSB layers from $\mathbf{E}_1'$ using

$$A_1(i, j) = 8 \cdot \lfloor E_1'(i, j) / 8 \rfloor$$

where $1 \leq i \leq M$ and $1 \leq j \leq N$. In image $\mathbf{A}_1$, five MSB layers are the same as those in $\mathbf{E}_1'$, and three LSB layers are zero. The receiver also divides $\mathbf{A}_1$ into R blocks $\{\mathbf{T}_1, \mathbf{T}_2, \dots, \mathbf{T}_R\}$, each of which is sized $r \times r$, where $R = \lfloor MN/4r^2 \rfloor$. Next, several steps are used to do the recovery.

In the first step, the receiver exchanges the order of $\mathbf{W}_1'$ with $\mathbf{W}_i'$ ($1 < i \leq R$), and calculates $\mathbf{T}_1^{(i)} = \mathbf{T}_1 + \mathbf{W}_i'$ and $\mathbf{T}_i^{(1)} = \mathbf{T}_i + \mathbf{W}_1'$. After decrypting $\mathbf{T}_1^{(i)}$ to $\mathbf{V}_1^{(i)}$ and $\mathbf{T}_i^{(1)}$ to $\mathbf{V}_i^{(1)}$, the receiver finds the best one from $R-1$ exchanges using

$$k = \arg \min_i \left\{ \sum_{a=1}^r \sum_{b=1}^{r-1} |V_1^{(i)}(a, b) - V_1^{(i)}(a, b+1)| + \sum_{a=1}^{r-1} \sum_{b=1}^r |V_1^{(i)}(a, b) - V_1^{(i)}(a+1, b)| \right. \\ \left. + \sum_{a=1}^r \sum_{b=1}^{r-1} |V_i^{(1)}(a, b) - V_i^{(1)}(a, b+1)| + \sum_{a=1}^{r-1} \sum_{b=1}^r |V_i^{(1)}(a, b) - V_i^{(1)}(a+1, b)| \right\} \quad (9)$$

where $i=2, \dots, R$. This way, the original content of the first block and the k -th block are recovered as $\mathbf{V}_1^{(k)}$ and $\mathbf{V}_k^{(1)}$, and the message hidden in this swapping can be identified as $(k-2)$.

In the second step, the receiver removes $\mathbf{W}_1'$ and $\mathbf{W}_k'$ from the set $\{\mathbf{W}_1', \mathbf{W}_2', \dots, \mathbf{W}_R'\}$, and removes $\mathbf{T}_1$ and $\mathbf{T}_k$ from $\{\mathbf{T}_1, \mathbf{T}_2, \dots, \mathbf{T}_R\}$. With the same algorithm used in first step, the receiver recovers another pair of the blocks and accordingly extracts the hidden data. Repeat the procedure until all swapped blocks have been recovered and hidden data extracted. This way, the original content of the first sub-image is recovered. Denote the recovered sub-image as $\mathbf{D}_1$.

5.2 Second Round Recovery

The receiver further implements the reverse operations of cyclic shifting, during which hidden messages can be extracted and the original contents corresponding to $\mathbf{E}_2'$, $\mathbf{E}_3'$ and $\mathbf{E}_4'$ can be recovered. The receiver constructs a new image $\mathbf{X}$ sized $M \times N$, and identifies a quarter of pixels in $\mathbf{X}$ by

$$X(2i-1, 2j-1) = D_1(i, j), \quad i = 1, 2, \dots, M/2; j = 1, 2, \dots, N/2,$$

and estimates the other pixels by linear interpolation. Divides $\mathbf{X}$ into four sub-images $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_3$ and $\mathbf{X}_4$ using the down-sampling depicted in Fig. 3.

The receiver extracts three least significant bits from $\mathbf{E}_2'$, $\mathbf{E}_3'$ and $\mathbf{E}_4'$ by

$$J_k'(i, j) = E_k'(i, j) \bmod 8, \quad k = 2, 3, 4$$

where $1 \leq i \leq M/2$ and $1 \leq j \leq N/2$, and construct image $\mathbf{A}_2$, $\mathbf{A}_3$ and $\mathbf{A}_4$ by removing three LSB layers in $\mathbf{E}_2'$, $\mathbf{E}_3'$ and $\mathbf{E}_4'$,

$$A_k(i, j) = E_k'(i, j) - J_k'(i, j), \quad k = 2, 3, 4$$

In image $\mathbf{A}_2$, $\mathbf{A}_3$ and $\mathbf{A}_4$, five MSB layers are the same as those in $\mathbf{E}_2'$, $\mathbf{E}_3'$ and $\mathbf{E}_4'$, and three LSB layers are zero. Divide all $3MN/4$ data in $\mathbf{J}_k'$, $k=2, 3, 4$, into L segments $\{\mathbf{Z}_1, \mathbf{Z}_2, \dots, \mathbf{Z}_L\}$, where $\mathbf{Z}_m = (z_{m,1}, z_{m,2}, \dots, z_{m,a})$ and $m=1, 2, \dots, L$. The receiver also divides all pixels in $\mathbf{A}_2$, $\mathbf{A}_3$ and $\mathbf{A}_4$ into L segments $\{\mathbf{C}_1, \mathbf{C}_2, \dots, \mathbf{C}_L\}$, and all pixels in $\mathbf{X}_2$, $\mathbf{X}_3$ and $\mathbf{X}_4$ into L segments $\{\mathbf{G}_1, \mathbf{G}_2, \dots, \mathbf{G}_L\}$.

For each segment $\mathbf{Z}_m$, the receiver cyclically shifts the data within the segment to achieve all possible vectors $\mathbf{Z}_m^{(n)}$, where $\mathbf{Z}_m^{(n)} = (z_{m,n+1}, \dots, z_{m,a}, z_{m,1}, \dots, z_{m,n})$ and $n=1,2,\dots,a$. After calculating $\mathbf{C}_m^{(n)} = \mathbf{C}_m + \mathbf{Z}_m^{(n)}$, the receiver decrypts $\mathbf{C}_m^{(n)}$ to $\mathbf{F}_m^{(n)}$, and identifies the best one from all possible shifted vectors using

$$l = \arg \min_n \sum_{k=1}^a |G_m(k) - F_m^{(n)}(k)| \quad (10)$$

where $m=1,2,\dots,L$, and $n=1,2,\dots,a$. This way, the original content of the m -th segment is recovered as $\mathbf{F}_m^{(l)}$, and the hidden value is identified as $l-1$. After processing all segments, the original pixel values are recovered, and the hidden messages are extracted.

6. Experimental Results

To verify the proposed approach, we have implemented the algorithms in many images. A group results are shown in Fig. 4, in which (a) and (e) are the original images “Lena” and “Pepper”, (b) and (f) the encrypted images. Fig. 4(c) and (g) are the marked encrypted images containing 3897 and 4132 additional bits, respectively. Fig. 4(d) and (h) show the directly decrypted image. Since only three LSB layers of the encrypted images are modified, the directly decrypted images preserve good quality. PSNR values of (d) and (h) are 38.7db and 38.6db, respectively. The marked encrypted images in (c) and (g) can be losslessly recovered after image decryption. The recovered images are the same as (a) and (e).

Fig. 4 (a) and (e) are the original images, (b) and (f) the encrypted image, (c) and (g) the marked encrypted images, and (d) and (h) the decrypted image

We also compare the proposed method with the joint RDH-EI methods in [3]~[5]. All experiments are conducted with the condition that the marked encrypted images can be recovered losslessly. A general way of choosing parameters in [3]~[5] is to construct a training set containing plaintext images and find suitable parameters by implementing the recovery algorithms in plaintext domain. Achievable embedding capacities are listed in Table I, indicating that the proposed method has a much larger capacity than the other methods.

Table I. Comparisons of embedding capacity (bits) using different methods

Images	[3]	[4]	[5]	Proposed
Lena	1156	1296	1296	3897
Baboon	196	225	225	629
Man	625	900	784	3630
Lake	1024	1296	1024	2094
Goldhill	784	1764	784	2434
Boats	1024	1764	1296	2434
Couple	1024	1396	1396	3723
Tiffany	900	1153	784	1497

Errors may happen when the payload is higher than the achievable embedding capacity. We have also analyzed the error rates, comparing with the methods in [3]~[5]. Fig. 5 shows the relationships between the error rates and different payloads using the image “Lena”. The results show that the proposed method has a much smaller error rate than the other methods.

Fig.5 The relation between embedding payload and error rate

The proposed method can also be used in medical images, which avoids the limitation in [3]~[5]. We randomly choose six medical images, as shown in Fig. 6, and implement the experiments under the condition of lossless recovery. Achievable capacities of [3]~[5] and the proposed method is listed in Table 2. The results show that the proposed method preserves good embedding efficacy, which the few bits can be embedded using the other methods. Fig. 7 shows the relationships between the error rates and different payloads using these medical images.

Figure 6 Medical images

Table 2 Comparisons of the embedding capacity in medical images using different methods

	[3]	[4]	[5]	Proposed
(a)	9	9	9	11608
(b)	9	9	9	11628
(c)	9	9	9	8439
(d)	9	9	9	8439
(e)	4	4	4	11428
(f)	4	4	4	11428

Fig.7 The relation between embedding payload and error rate

7. Conclusion

This paper proposes an improved method of joint reversible data hiding in encrypted images. After analyzing the related joint RDH-EI works, we found a limitation inside them. Accordingly, we propose a new approach capable of avoiding the limitation and improving the achievable embedding capacity. The combination of cyclic-shifting and data-swapping is used to embed additional messages into the ciphered image. The process is reversible that the original image can be losslessly recovered and the hidden message exactly extracted. Experimental results show that this joint RDH-EI approach outperforms the related works on the aspect of embedding capacity, and can be used for the encrypted medical images.

Acknowledgements

This work was supported by Natural Science Foundation of China (Grant 61572308, Grant 61232016, Grant U1405254, Grant U1536108, Grant 61525203, and Grant 61472235), Shanghai Key Laboratory of Intelligent Information Processing under Grant IIPL-2014-006, 2015 Shanghai University Filmology Summit Research Grant, the National Science Foundation China under grant 61402279, the Shanghai Nature Science Foundation under grant 14ZR1415900, the Open Project Program of the State Key Lab of CAD&CG under grant A1502, and the PAPD fund.

References

- [1] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su. Reversible data hiding, *IEEE Transactions on Circuits and Systems for Video Technology*, 2006, 16 (3): 354–362
- [2] W. L. Tai, C. M. Yeh, and C. C. Chang, Reversible data hiding based on histogram modification of pixel

differences, *IEEE Transactions on Circuits and Systems for Video Technology*, 2009, 19 (6): 906–910

- [3] X. Zhang, Reversible Data Hiding in Encrypted Image, *IEEE Signal Processing Letters*, 2011, 18(4): 255-258
- [4] W. Hong, T. S. Chen, H. and Y. Wu, An Improved Reversible Data Hiding in Encrypted Images Using Side Match, *IEEE Signal Processing Letters*, 2012, 19(4): 199-202
- [5] X. Liao, and C. W. Shu, Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels, *Journal of Visual Communication and Image Representation*, 2015, 28(2): 21-27
- [6] X. Zhang, Separable Reversible Data Hiding in Encrypted Image, *IEEE Transactions on Information Forensics and Security*, 2012, 7(2): 826-832
- [7] X. Wu, and W. Sun, High-capacity reversible data hiding in encrypted images by prediction error, *Signal processing*, 2014, 104: 387-400
- [8] Z. Qian, and X. Zhang, Reversible Data Hiding in Encrypted Image with Distributed Source Encoding, *IEEE Trans. on Circuits and Systems for Video Technology*, 2016, 26(4): 636-646
- [9] K. Ma, W. Zhang, X. Zhao, et al. Reversible Data Hiding in Encrypted Images by Reserving Room Before Encryption, *IEEE Transactions on Information Forensics and Security*, 2013, 8(3): 553-562
- [10] X. Cao, L. Du, X. Wei, et al. High capacity reversible data hiding in encrypted images by patch-level sparse representation, *IEEE Trans. on Cybernetics*, doi:10.1109/TCYB.2015. 2423678
- [11] W. Zhang, K. Ma and N. Yu, Reversibility improved data hiding in encrypted images, *Signal Processing*, 2014, 94: 118-127
- [12] Y. Chen, C. W. Shiu, and G. Horng, Encrypted signal-based reversible data hiding with public key cryptosystem, *Journal of Visual Communication and Image Representation*, 2014, 25(5): 1164-1170
- [13] J. Zhou, W. Sun, L. Dong, et al. Secure reversible image data hiding over encrypted domain via key modulation, *IEEE Trans. on Circuits and Systems for Video Technology*, 2016, 26(3): 441-452
- [14] Z. Qian, X. Zhang, and S. Wang, Reversible data hiding in encrypted JPEG bit stream, *IEEE Transactions on Multimedia*, 2014, 15(2): 316-325
- [15] D. Xu, R. Wang, and Y. Q. Shi, Data hiding in encrypted H. 264/AVC video streams by codeword substitution, *IEEE transactions on information forensics and security*, 2014, 9(3-4): 596-606
- [16] Z. Qian, X. Zhang, and G. Feng, Reversible Data Hiding in Encrypted Images Based on Progressive Recovery, *IEEE Signal Processing*, doi:10.1109/LSP.2016.2585580