Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

High-payload completely reversible data hiding in encrypted images by an interpolation technique*

Di XIAO†‡[1], Ying WANG[1], Tao XIANG[1], Sen BAI[2]

([1]*MOE Key Laboratory of Dependable Service Computing in Cyber Physical Society, College of Computer Science, Chongqing University, Chongqing 400044, China*)

([2]*Department of Information Engineering, Chongqing Communication Institute, Chongqing 400035, China*)

†E-mail: xiaodi_cqu@hotmail.com

Received Mar. 8, 2016; Revision accepted Apr. 2, 2016; Crosschecked Nov. 24, 2017

Abstract: We present a new high-payload joint reversible data-hiding scheme for encrypted images. Instead of embedding data in the encrypted image directly, the content owner first uses an interpolation technique to estimate whether the location can be used for embedding and generates a location map before encryption. Next, the data hider embeds the additional data through flipping the most significant bits (MSBs) of the encrypted image according to the location map. At the receiver side, before extracting the additional data and reconstructing the image, the receiver decrypts the image first. Experimental results demonstrate that the proposed method can achieve real reversibility, which means data extraction and image recovery are free of error. Moreover, our scheme can embed more payloads than most existing reversible data hiding schemes in encrypted images.

Key words: Encrypted image; Data hiding; Image recovery; Real reversibility; Interpolation
https://doi.org/10.1631/FITEE.1601067 **CLC number:** TP309.7

1 Introduction

Reversible data hiding (RDH) in images is a technique to ensure embedding additional data into the original image in a lossless manner. After the extraction of additional data, the original cover image can be exactly recovered. It has many applications such as in the medical field as well as in military and law forensics, where any distortion of the original image is unacceptable. Therefore, RDH has been a hot research topic in recent years.

Many RDH schemes have emerged so far. They can be roughly categorized into three fundamental strategies: difference expansion (DE), histogram modification (HM), and lossless-compression-based schemes. In the DE method, the difference between adjacent pixels is doubled to embed 0 or 1 (Tian, 2003; Thodi and Rodriguez, 2007). Instead of trying to keep the peak signal-to-noise ratio (PSNR) high, Wu *et al*. (2015) proposed an algorithm that enhances the contrast of a host image to improve its visual quality by repeating the histogram equalization. In the HM method (Ni *et al*., 2006), the bins between the peak and zero points in the histogram are shifted to vacate space for additional data. The main idea of the lossless-compression-based scheme is to use a lossless compression algorithm to empty space for embedding (Fridrich *et al*., 2001; Celik *et al*., 2005). For better performance, some other methods apply the above three strategies to reduce the prediction errors of the image (Hu *et al*., 2009; Luo *et al*., 2010; Lee

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (Nos. 61572089 and 61633005), the Natural Science Foundation of Chongqing Science and Technology Commission (No. cstc2017jcyjBX0008), the Chongqing Graduate Student Research Innovation Project (No. CYB17026), and the Fundamental Research Funds for the Central Universities (Nos. 106112017CDJQJ188830 and 106112017CDJXY180005)

ORCID: Di XIAO, http://orcid.org/0000-0002-6958-5807

© Zhejiang University and Springer-Verlag GmbH Germany 2017

and Chen, 2012; Peng *et al*., 2014; Dragoi and Coltuc, 2015; Li *et al.*, 2015). Besides, some robust RDH schemes have been proposed (Zeng *et al.*, 2010; An *et al*., 2012a; 2012b). They can provide robustness against unintentional attacks, which means that the recovery performance can be improved even when the images are attacked to some extent.

With the increasing demand for privacy protection, encryption is an effective and popular way by which the original image is concealed to an unintelligible one. It is of great use to embed data in encrypted images. For instance, in cloud computing applications, because the images that the users store in the cloud server need privacy protection, they will be encrypted by the users first. Then, the mission like embedding classified information, copyright information, or other information by the service provide, is based on the situation where the original content is kept unknown. When the images disallow any distortion, such as military images, they should be restored completely. In some other application scenarios, data hiding in encryption is also preferred. This research is urgently needed and valuable.

Several attempts have been made to embed additional data in encrypted images reversibly. These embedding methods can be further classified into two categories, namely, joint and separable. In joint methods, data extraction and image decryption are performed jointly. The encrypted image is divided into several blocks first (Zhang, 2011). Then the additional data is embedded by flipping three least significant bits (LSBs) of half of the pixels in each block, and the embedded data is extracted with the aid of spatial correlation in the natural image. Hong *et al.* (2012) improved Zhang (2011)'s method by further exploiting the spatial correlation with a different estimation equation and side-matching to obtain a lower error rate at the receiver side. Liao and Shu (2015) further increased the correctness of data extraction/image recovery when the block size was appropriate, by designing a new function to estimate the complexity of image blocks sufficiently and considering the data embedding ratio. In separable methods, the independence between data extraction and image decryption is guaranteed. A new way has been found to embed data in encrypted images (Ma *et al.*, 2013; Zhang W *et al.*, 2014). By this way, some prior operations are conducted in the original image before encryption. In Ma *et al*. (2013)'s method, some pixels are estimated before encryption, so that additional data can be embedded in the estimated errors rather than in encrypted images directly. In Zhang W *et al*. (2014)'s method, the room is reserved before the original image is encrypted with a traditional RDH algorithm. Some other common methods are based on compression using distributed source coding (DSC) or LDPC code (Zhang, 2012; Zhang X *et al.*, 2014; Qian and Zhang, 2016). Cao *et al.* (2016) proposed to consider patch-level sparse representation when hiding the secret data. Wu and Sun (2014) presented both a joint method and a separable method by adopting the prediction error. Besides, some secure RDH schemes applied specifically in JPEG have been proposed (Qian *et al.*, 2013).

The schemes proposed by Zhang (2011), Hong *et al.* (2012), and Liao and Shu (2015) confine errors in data extraction and image distortion during image recovery. Data extraction and image recovery are based on estimation of the original LSB planes with the fluctuation function. Because the functions are not accurate enough, the recovery is free of error only when a small number of additional bits have been embedded. Although error-correcting codes can be used to eliminate the error, the pure payloads will be further reduced. In all the methods above, the data hider can embed only a small payload of data. As Wu and Sun (2014) showed, a joint method has been proposed to provide high-payload embedding. However, the problem of incorrect extraction and recovery still exists.

All the above analyses show that reversible data hiding in encrypted images is difficult and not perfect. The operation of encryption will maximize the information entropy of the image, so it is hard to embed additional information into the encrypted image. In addition, due to the huge amount of modifications in the encrypted image, the reconstruction process is more technically challenging. Compared with the simple RDH, RDH in the encrypted domain is more difficult. Also, improving the embedded capacity is more challenging.

In this paper, we propose a joint RDH in encrypted images based on the interpolation technique, which can embed more data than most existing methods in the encrypted images. First, the content owner uses the interpolation technique to estimate

whether the pixel location can be used for embedding, generates a location map by recording those unavailable pixels, and then encrypts the original image. Although the interpolation value of the original image is calculated, the original image does not need to be directly processed before encryption. Next, the data hider embeds the additional data through flipping the MSBs of the encrypted image according to the location map. At the receiver side, the receiver first decrypts the image, and then extracts the additional data and reconstructs the image. In our method, real reversibility is realized, which means both data extraction and image decryption are free of error. The method has the potential of being applied in the following scenario. For example, in cloud computing, as the user of the cloud, the content owner will upload the encrypted image to the cloud for privacy protection. For easy storage and management, the cloud server, as a third party, needs to embed some special data into the encrypted image. The legal receiver, who needs the data stored in the cloud, will download the encrypted image with additional data, and then implement extraction and recovery by himself/herself.

2 General framework

The general framework of the proposed scheme is made up of five phases, namely location map generation, image encryption, data embedding, location map extraction and image decryption, and data extraction and image recovery. As shown in Fig. 1, the content owner first implements location map generation and image encryption. Then, data embedding is done by the data hider. At the receiver side, the receiver first extracts the location map and decrypts the image, and then extracts the additional data and recovers the image. In the following, we will briefly illustrate the five phases of the entire workflow through three parties: content owner, data hider, and receiver.

The content owner generates a location map before image encryption using the interpolation technique, which indicates whether the location can be used for embedding or not, and then encrypts the original image to conceal the principal content. An encryption key is decided by the content owner. Both the encrypted image and the location map are sent to the data hider.

The data hider determines the available location to be embedded by leveraging the location map and the data hiding key, and then embeds the additional data into the encrypted image though flipping the MSB. It embeds the location map into the encrypted image by replacing the border pixels. The embedding capacity is the number of available embedding locations and is more than that of most existing schemes. The data hiding key is decided by the data hider.

With both the encryption key and data hiding key, the receiver extracts the location map, decrypts the encrypted image with embedded data, and then recovers the additional data and the image simultaneously using the location map. The extracted data is free of error and the recovered image is exactly the same as the original image.

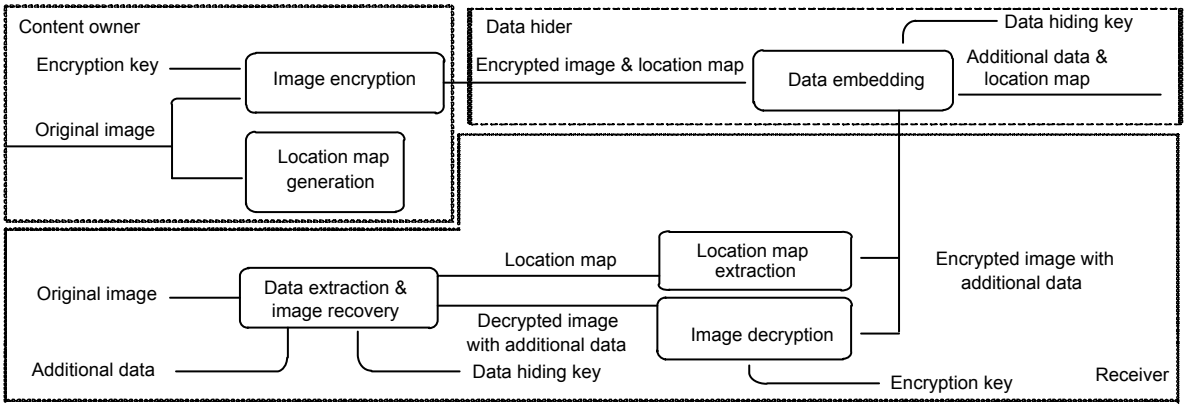

Fig.1 Sketch of the proposed method

3 Location map generation and encryption

3.1 Interpolation technique

First, we apply a feasible interpolation algorithm to obtain interpolation values of pixels. Image interpolation is a process during which the image is resized and a tradeoff among efficiency, smoothness, and sharpness is involved, which can achieve a high-resolution image from its low-resolution counterpart. Here, we call the different pixels between the high- and low-resolution images missing pixels. It has applications in medical imaging, remote sensing, and digital photographs.

Luo *et al*. (2010) proposed an effective image interpolation algorithm that infers the missing pixels (white pixels in Fig. 2a) in the original image from their neighboring pixels (the gray pixels in Fig. 2a, which form the low-resolution image directly downsampled from Fig. 2a) along diagonal and horizontal/vertical directions, respectively, in two steps. First, the pixels with black dots are interpolated along the 45° and 135° directions, where four neighboring pixels are all gray in both directions. Second, the pixels with white dots are interpolated along the 0° and 90° directions. We can observe that there are two gray pixels and two black dot pixels as neighbors, which means the second interpolation operation is based on the first step. A different result at the first step will lead to a different result at the second step. For example, if we change all the missing pixels for embedding using their interpolation values, then in the recovery process the interpolation values of white dot pixels are different from those in the embedding process because of the different values of the black dot pixels. There must be an error after restoring the data and image. So, we simplify Luo *et al*. (2010)'s method by interpolating along just one direction, which needs only one step to make the missing pixels independent. In this way, not only the correctness of recovery can be guaranteed but also the size of the location map can be decreased.

Let $I(i, j)$ be the pixel value of the original grayscale image, and $I'(i, j)$ the interpolation value at position (i, j). Fig. 2b illustrates our method. Pixels in the image are categorized into two sets: missing pixels of Fig. 2b (white pixels) and neighboring pixels (gray pixels in Fig. 2b). Each missing pixel is estimated through the interpolation value obtained by the four neighboring pixels. The neighboring pixels are partitioned into two orthogonal directions, 0° (horizontal) and 90° (vertical), and the two directional interpolation results are denoted by I_0' and I_{90}', respectively. The interpolation value of the missing pixels can be estimated as

$$I'(i,j) = w_0 \cdot I_0' + w_{90} \cdot I_{90}', \quad w_0 + w_{90} = 1, \tag{1}$$

where w_0 and w_{90} are an optimal pair of weights to fuse the two values that can be determined by the same method as proposed by Luo *et al*. (2010). I_0' and I_{90}' are computed as

$$\begin{cases} I_0' = [I(i, j-1) + I(i, j+1)] / 2, \\ I_{90}' = [I(i-1, j) + I(i+1, j)] / 2. \end{cases} \tag{2}$$

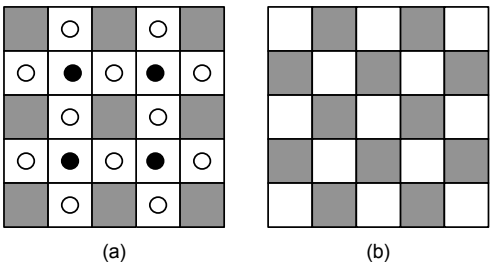

Fig. 2 Missing pixels and four adjacent pixels: (a) Luo *et al*. (2010)'s scheme; (b) our scheme

In this way, we can estimate the missing pixels by four adjacent pixels.

3.2 Location map generation

An example is given in Fig. 3. Let i and j be the indices of the original image I with the size of $M \times N$. It is evident that the missing pixels satisfy the following equation:

$$(i + j) \bmod 2 = 0,\ 3 \le i \le M - 2,\ 3 \le j \le N - 2. \tag{3}$$

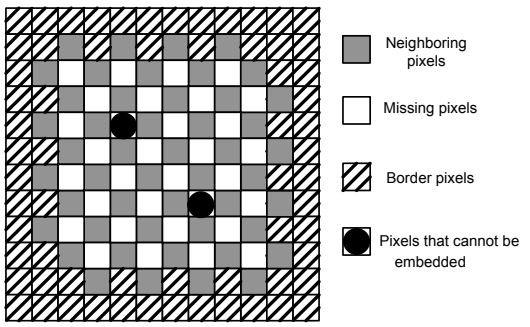

Fig. 3 Example of the distribution of different types of pixels in the image

In the best case, the data hider can embed data in every pixel of the white location. However, some pixels in the white location cannot be used for embedding. The reason is that an error will occur in the extraction process if the data hider embeds the data there. To eliminate the error, the content owner has to generate a location map to indicate the locations where data can be embedded.

Let L, which is a one-dimensional array, be the location map, and $2\times b$ be the size of the location map (b is the number of bad correlation pixels). If the pixel in the white location has correlation with four adjacent pixels (the four neighboring gray pixels as in Fig. 2b), the data hider can embed data there. If the pixel in the white location does not have correlation with four adjacent pixels, the data hider cannot embed data, and we record this pixel by putting its location (i, j) to L (for one pixel, we can first put its row number i into L, and then put its column number j into L). As mentioned earlier, $I(i, j)$ is the original pixel value in the white location. $I'(i, j)$ is the estimation value of $I(i, j)$ using interpolation. We define $I(i, j)$, which has correlation with adjacent pixels, by

$$\text{abs}(I(i,j)-I'(i,j)) < \text{abs}(I'(i,j)-f(I(i,j))), \quad (4)$$

where $f(I(i, j))$ is to flip the MSB of $I(i, j)$. It can be described as

$$f(I(i,j)) = I(i,j) \oplus 128, \quad (5)$$

where $\oplus$ means the bitwise XOR operation. Eq. (4) indicates that if the estimation value of a pixel is closer to $I(i, j)$ rather than $f(I(i, j))$, it has correlation with four adjacent pixels. On the contrary, $I(i, j)$ has bad correlation with adjacent pixels if

$$\text{abs}(I(i,j)-I'(i,j)) \geq \text{abs}(I'(i,j)-f(I(i,j))). \quad (6)$$

The location map is a piece of auxiliary information. It tells the data hider whether the location can be used for embedding or not. In addition, the receiver can decide whether to extract data with the location map.

3.3 Image encryption

The plain image is $[I(1,1), \ldots, I(i, j), \ldots, I(M, N)]$. The key stream $[K(1,1), \ldots, K(i,j), \ldots, K(M, N)]$ is obtained from an encryption key using a standard stream cipher algorithm. A number of secure stream cipher methods can be used to ensure that anyone without the encryption key does not know any information about the original cover. To encrypt every pixel of the image, bitwise XOR is performed on the binary digits of $I(i, j)$ and $K(i, j)$:

$$E(i,j) = I(i,j) \oplus K(i,j), \quad (7)$$

where $E(i, j)$ is the pixel of the encrypted image. The content owner encrypts all the pixels of the original image using the encryption key and obtains the encrypted image. The content owner sends it to the data hider along with the location map.

4 Data embedding

After data encryption, when the data hider receives the encrypted image and location map, he/she can embed additional data $A=\{a_0, a_1, \ldots, a_{n-1}\}$, where n is the size of A in the white location. According to the location map, he/she knows exactly which pixel can be used for embedding among these white pixels. The data hider pseudo-randomly chooses a collection of locations of these pixels for embedding (white pixels in Fig. 3) using a data hiding key until the number of collections is equal to the size of the additional data n. $E(i, j)$ is a pixel in the white location of the encrypted image. The embedding process works in order as follows:

$$E'(i,j) = \begin{cases} E(i,j), & a_k = 0, \\ f(E(i,j)), & a_k = 1, \end{cases} \quad (8)$$

where E' is the encrypted image with additional data, and k is the index of additional data A. If the embedding data is 0, we do not apply any operation to the encrypted pixel; if the embedding data is 1, we flip the MSB of the encrypted pixel. Then the encrypted image containing additional data is generated. Here we choose the operation of flipping the MSB instead of the LSB because we use Eq. (4) to generate the location map in Section 3.2; by doing so, the reversibility can be realized, which will be described later.

5 Reversibility

When the receiver has both the encryption key and data hiding key, he/she should decrypt the image first, and can finally recover the image completely and extract the encrypted data correctly.

5.1 Image decryption

At the beginning, the receiver decrypts E' to achieve the directly decrypted image D using the same encryption key as that in image encryption described in Section 3.2:

$$D(i,j)=E'(i,j)\oplus K(i,j). \tag{9}$$

In Section 4, we can see that the additional bits are hidden in the MSBs, so the directly decrypted image D is different from the original image. However, after implementing the extraction with the help of the data hiding key and the location map as described in the following, the receiver can recover the image completely and restore the additional data free of error.

When the receiver does not have a data hiding key, he/she cannot achieve the original image, which is not a desirable result. The receiver can do some operations further. As shown in Section 3.1, he/she can leverage the interpolation technique again to estimate the white pixels in the directly encrypted image:

$$\begin{cases} D'(i,j)=w_0\cdot D'_0+w_{90}\cdot D'_{90}, \\ D'_0=[D(i,j-1)+D(i,j+1)]/2, \\ D'_{90}=[D(i-1,j)+D(i+1,j)]/2, \end{cases} \tag{10}$$

where $D'(i,j)$ means the interpolation value of $D(i,j)$, and the values of w_0 and w_{90} are the same as described in Section 3.1. After doing so, the quality of the image will be improved.

5.2 Data extraction and image recovery

As described above, the directly decrypted image D is not the original image. For the pixels in the gray location and the pixels in the white location that are embedded with 0, they are completely recovered. For the pixels embedded with 1, the decryption results are the original pixels with the MSBs flipped. We can illustrate it using the following equations:

$$\begin{cases} E'(i,j)=f(E(i,j))=E(i,j)\oplus 128 \\ \qquad\quad =I(i,j)\oplus K(i,j)\oplus 128, \\ D(i,j)=E'(i,j)\oplus K(i,j) \\ \qquad\quad =I(i,j)\oplus 128=f(I(i,j)). \end{cases} \tag{11}$$

Therefore, the receiver needs to make sure whether the pixel embedded with data is $I(i,j)$ or $f(I(i,j))$ after decryption to completely recover the original image. The pixel embedded with data satisfies Eq. (4). So, the receiver can use the following method to determine whether the pixel embedded with data is $I(i,j)$ or $f(I(i,j))$:

If $\mathrm{abs}(D(i,j)-I'(i,j))<\mathrm{abs}(I'(i,j)-f(D(i,j)))$, it means $D(i,j)$, which is the same as $I(i,j)$, is the correct result, and the embedded encrypted data is 0.

If $\mathrm{abs}(D(i,j)-I'(i,j))\geq\mathrm{abs}(I'(i,j)-f(D(i,j)))$, it means $D(i,j)$, which is the same as $f(I(i,j))$, is not the correct result, and $f(D(i,j))$ is the final result; conversely, the embedded encrypted data is 1.

In this way, the receiver obtains every original pixel of the image, and the embedded data has been embedded in the white position. Then the receiver recovers these original pixels of the image in order and reorders the embedded data in correct order according to the data hiding key and the location map. When the receiver does not have the data hiding key, he/she cannot acquire the right data or the original image. Finally, the restored image is exactly the same as the original one, and the additional data is extracted correctly.

5.3 About the side information

To realize reversibility, the side information includes the size of the additional bits n, the number of bad correlation pixels b, and the location map L, which should be sent to the receiver as auxiliary information, as mentioned previously. It can be embedded in the border pixels by the data hider. As shown in Fig. 3, the border pixels are used to embed the side information, and the pixels with a circle mean the location cannot be used for embedding. As the size of the original image is $M\times N$, the total number of pixels on the border is $2\times M+2\times N-4+((M-2)/2+1)\times 2+(N-4)/2\times 2=3(M+N)-8$. As shown in Section 3, since the value of L is the row number or column number of one pixel, it can be represented by $\log_2 M$ or $\log_2 N$ bits. The number of bad correlation pixels is

b, so we need $b(\log_2 M+\log_2 N)$ bits to store the location map. In addition, we reserve 16 bits for the variables n and b, respectively. So, the number of bits of the side information is $b(\log_2 M+\log_2 N)+32$. Then, we replace the original border pixels to store the side information according to the clockwise direction. Actually, for a better performance, entropy coding can be exploited to reduce the size of side information.

The image owner sends the encrypted image and location map to the data hider. For the data hider, as some of the border pixels are occupied by the location map, the original data of border pixels will be embedded in the encrypted image with the additional data, as mentioned previously. At the receiver side, after location map extraction and image decryption, the receiver extracts the additional data, including the border pixels. After recovering the image, the receiver also needs to recover these border pixels.

6 Experimental results

The proposed method is implemented using MATLAB 2010a on a computer with an Intel core I3-3220 CPU at 3.30 GHz. The standard test image Lena (512×512×8) shown in Fig. 4a is tested to demonstrate the feasibility of the proposed algorithm. The encrypted image is displayed in Fig. 4b, while the one after embedding is shown in Fig. 4c; the embedding rate is 0.4922 bits/pixel. The receiver decrypts the encrypted image by the encryption key, and obtains the directly decrypted image in Fig. 4d, which has bad visual quality. After using the interpolation technique, the secret image is approximately reconstructed in Fig. 4e, whose PSNR is 37.7542 dB. When the receiver also has the data hiding key, the right original image illustrated in Fig. 4f can be obtained, and complete reversibility can be achieved between the extracted data and the original data.

6.1 Feasibility analysis

The size of the original image is $M\times N$, where M and N are even numbers. The number of white pixels (denoted as t), which is also the total number of embedding bits in the ideal case, is

$$t=(N-4)(M-4)/2. \tag{12}$$

If b is the number of pixels of bad correlation, the number of embedding bits is $t-b$. Obviously, a smaller b will result in a larger embedding capacity $t-b$. The embedding rate is

$$\begin{aligned}&\text{Embedding rate}\\&=\frac{\text{Number of total embedded bits}}{\text{Number of total pixels of the image}}\\&=\frac{t-b}{MN}=0.5-\frac{2(M+N-4)+b}{MN}.\end{aligned} \tag{13}$$

If M and N are large enough, the upper limit of the maximum embedding rate is approximately 0.5. Actually, the side information should be embedded in the encrypted image as described in Section 5.3. The number of bits of the side information is $s=b(\log_2 M+\log_2 N)+32$. In our experiment, the original image is the standard test image of size 512×512×8, so the number of bits of the side information is $s=18b+32$. Then the pure embedding rate is

$$\begin{aligned}\text{Pure embedding rate}&=\frac{t-b-s}{MN}\\&=0.5-\frac{2(M+N-12)+19b}{MN}.\end{aligned} \tag{14}$$

In practice, for a better performance, entropy coding can be exploited to reduce the size of side information.

6.1.1 About the side information

Before comparing the proposed method with existing RDHs for encrypted images, let us take a look at the location map, which plays an important role in the side information. We first test on eight standard images: Lena, Airplane, Barbara, Baboon, Boat, Hill, Peppers, and Lake (Fig. 5). Then we keep a record of the number of pixels of bad correlation, which is denoted as b.

From Table 1, we can see that in most occasions, the number of pixels of bad location is very small. However, in Barbara and Baboon, this number is a little larger than those in other images. In these two images, there are many spots that have a large difference with its neighboring pixels, which leads to a long location map.

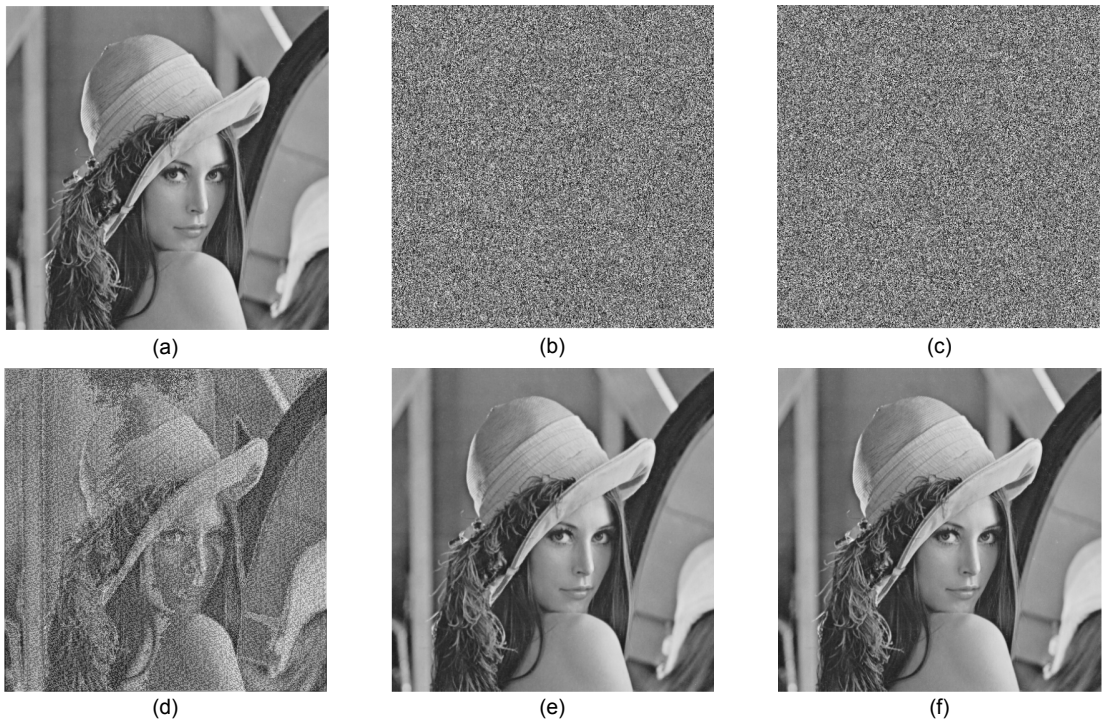

Fig. 4 Original Lena (a), encrypted image (b), encrypted image with additional data (c), directly decrypted image (d), interpolated decrypted image (e), and completely restored image (f)

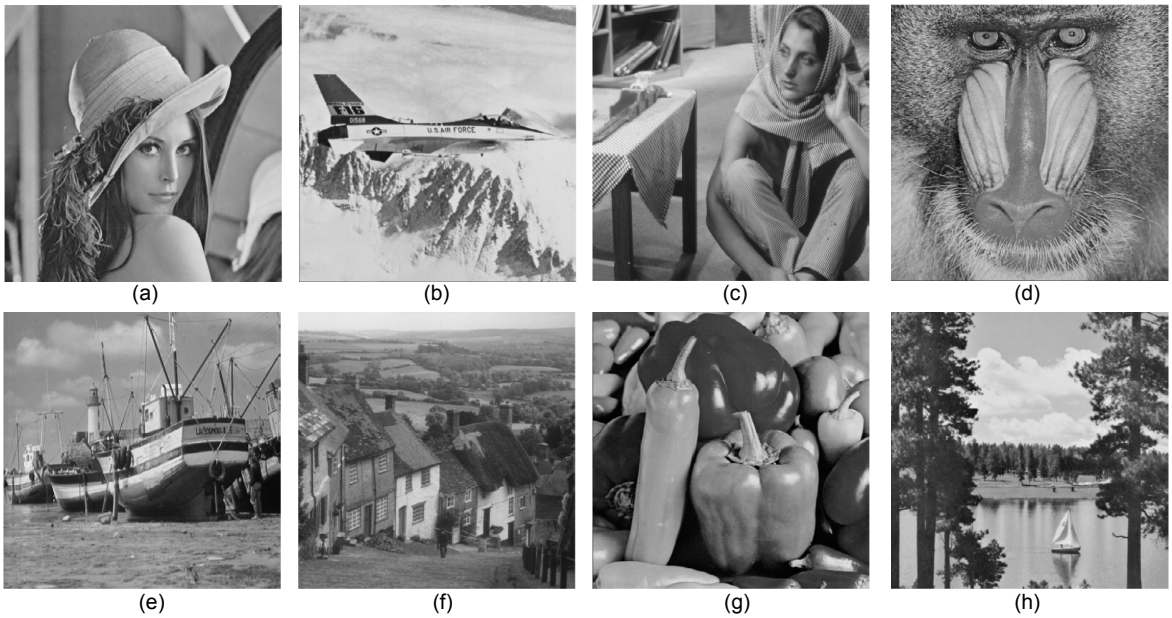

Fig. 5 Eight test images: (a) Lena; (b) Airplane; (c) Barbara; (d) Baboon; (e) Boat; (f) Hill; (g) Peppers; (h) Lake

Here, the proportion p of the side information to the embedding capacity is

$$p=\frac{s}{t-b}\times 100\%. \tag{15}$$

If $p<100\%$ is true (which means $s<t-b$), there is space for embedding, and our proposed method is feasible in practice. Actually, based on the spatial correlation in natural images and the intrinsic feature of the interpolation technique, the value of the

interpolated I' is close to the value of the original I, which can ensure that b is smaller than t and s is also smaller than t (Table 1). In addition, for a smoother image or a better interpolation technique, the performance will be better. Table 2 shows our experimental results, showing that the side information is very small.

Table 1 Number of pixels of bad correlation with adjacent pixels in the eight test images

Image	Number of pixels	Image	Number of pixels
Lena	0	Boat	28
Airplane	2	Hill	0
Barbara	546	Peppers	3
Baboon	294	Lake	1

Table 2 Proportion of side information to the available embedding capacity

Image	Proportion (%)	Image	Proportion (%)
Lena	0.0248	Boat	0.4155
Airplane	0.0527	Hill	0.0248
Barbara	7.6740	Peppers	0.0667
Baboon	4.1355	Lake	0.0388

6.1.2 About border pixels

As described above, the side information is $s=b(\log_2 M+\log_2 N)+32$ bits and the number of border pixels is $x=3(M+N)-8$. In our experiment, the original image is the standard test image of size 512×512×8. Only when $s\le 8x$ is true, are the border pixels enough to contain the side information. The equation can be rewritten as

$$\begin{cases} s \le 8x, \\ 18b+32 \le 8\times(3\times 1024-8) \\ \quad \Rightarrow\ b \le 1360. \end{cases} \tag{16}$$

It means that if the number of pixels of bad correlation b satisfies Eq. (15), there is enough space to embed the side information, and our proposed method is feasible. Actually, as we have discussed in Section 6.1.1, the value of b is small in most natural images. In Table 1 we can see that the maximum value is 546, which is much smaller than 1360.

6.1.3 Performance analysis

Without loss of generality, we test on eight representative images and obtain their maximum embedding rate. Table 3 shows the maximum embedding rate of the eight test images. It is evident that the maximum embedding rate is near 0.5, which is larger than that obtained using the methods presented by Zhang (2011), Hong *et al.* (2012), and Liao and Shu (2015), which is 0.0625 when the block size is 4×4 (a larger block size results in a smaller embedding rate), consistent with our results. In Wu and Sun (2014)'s scheme, the embedding rate can be controlled using different parameters. However, the reversibility will be affected at the same time. From their experimental results, we can see that the maximum embedding rate is 0.0625 while the number of incorrectly extracted bits is 0. We can find that the embedding rates are very close to each other even though the numbers of bad locations are quite different. This is because the size of the image is large for the amount. In addition, the pure embedding rate depends on the size of side information. As shown in Table 3, the pure embedding rates of Barbara and Baboon are smaller than those of other images. The reason is that there are long location maps in these two images. All the pure embedding rates are near the embedding rate, which embodies the fact that the side information does not influence the embedding capacity significantly (Table 2). The side information occupies a small space in the embedding capacity.

Table 3 Maximum embedding rate and maximum pure embedding rate of the eight test images

Image	Maximum embedding rate	Maximum pure embedding rate
Lena	0.4922	0.4921
Airplane	0.4922	0.4920
Barbara	0.4901	0.4525
Baboon	0.4911	0.4708
Boat	0.4921	0.4901
Hill	0.4922	0.4921
Peppers	0.4922	0.4919
Lake	0.4922	0.4920

6.2 Comparison with existing methods

In addition, to analyze the performance of our proposed scheme, two commonly used parameters PSNR and error rate are used:

$$\mathrm{PSNR}=10\times\lg\frac{255^2}{\mathrm{MSE}}, \tag{17}$$

$$\text{Error rate}=\frac{\text{Number of wrongly extracted bits}}{\text{Number of total embedded bits}}\times 100\%, \tag{18}$$

where MSE denotes the mean-squared error between the original and the current images.

Now, we compare the proposed scheme with the joint methods proposed by Zhang (2011). Hong *et al.* (2012), Wu and Sun (2014), and Liao and Shu (2015). Here, because of the inseparability of our method, we just compare it with the joint method proposed by Wu and Sun (2014). Since the embedding rate has been analyzed in Section 6.1.3, we just compare other performances here. The methods presented by Zhang (2011), Hong *et al.* (2012), and Liao and Shu (2015) extract data and recover the image based on the fluctuation function, and the method presented by Wu and Sun (2014) is based on the prediction error. Neither the functions nor the prediction error is accurate enough, so there are errors in all the above schemes.

Let us use experimental results to demonstrate the above analysis visually. Here, the different parameters in Wu and Sun (2014) will result in different results, and we just choose the best condition. Tables 4–7 show the error rate, PSNR of the decrypted images, and PSNR of the recovered images for Lena, Airplane, Baboon, and Lake under different embedding rates.

Regarding the error rate, as proved above, the proposed method extracts the additional data free of error, while in Zhang (2011), Hong *et al.* (2012), and Liao and Shu (2015)'s algorithms, an error occurs more often when the block size is smaller (which means that the embedding rate is higher). In Wu and Sun (2014)'s scheme, an error occurs according to the parameters.

Regarding image quality, the decrypted image's PSNR of our scheme is low as discussed in Section 5.1 because of the operation of flipping the MSB. However, after adopting the interpolation technique as a filter, the PSNR is improved, better than those obtained using the methods proposed by Zhang (2011), Hong *et al.* (2012), Wu and Sun (2014), and Liao and Shu (2015). By flipping the three LSBs, the PSNR of the decrypted image obtained using methods proposed by Zhang (2011), Hong *et al.* (2012), and Liao and Shu (2015) is about 37.9 dB. The result of Wu and Sun (2014)'s scheme also varies with the

Table 4 Performance comparison among the proposed method and related methods for Lena at two embedding rates of 0.0156 and 0.0625

Method	Error rate (%)		PSNR (dB)			
			Decrypted image		Recovered image	
	0.0156	0.0625	0.0156	0.0625	0.0156	0.0625
Zhang (2011)	1.440	13.538	37.90	37.90	55.69	44.80
Hong *et al.* (2012)	0.073	2.502	37.89	37.89	68.19	52.37
Liao and Shu (2015)	0.195	4.360	37.93	37.92	62.09	49.15
Wu and Sun (2014)	0	0	30.53	28.29	∞	∞
Proposed method	0	0	27.07/58.37*	21.04/50.51*	∞	∞

* The first value is the PSNR of the directly decrypted image, and the second value is the PSNR of the decrypted image after interpolation

Table 5 Performance comparison among the proposed method and related methods for Airplane at two embedding rates of 0.0156 and 0.0625

Method	Error rate (%)		PSNR (dB)			
			Decrypted image		Recovered image	
	0.0156	0.0625	0.0156	0.0625	0.0156	0.0625
Zhang (2011)	4.419	18.640	38.00	38.00	51.07	43.18
Hong *et al.* (2012)	0.146	4.657	37.99	37.99	66.26	49.97
Liao and Shu (2015)	0.732	5.273	37.98	37.99	59.97	49.38
Wu and Sun (2014)	0	0.003	30.66	27.07	∞	65.78
Proposed method	0	0	27.07/57.02*	21.04/52.23*	∞	∞

* The first value is the PSNR of the directly decrypted image, and the second value is the PSNR of the decrypted image after interpolation

Table 6 Performance comparison among the proposed method and related methods for Baboon at an embedding rate of 0.0625

Method	Error rate (%)	PSNR (dB)	
		Decrypted image	Recovered image
Zhang (2011)	34.002	37.93	39.65
Hong *et al.* (2012)	21.252	37.93	41.65
Liao and Shu (2015)	23.969	37.91	41.18
Wu and Sun (2014)	2.873	27.06	39.47
Proposed method	0	21.04/33.31*	∞

* The first value is the PSNR of the directly decrypted image, and the second value is the PSNR of the decrypted image after interpolation

Table 7 Performance comparison among the proposed method and related methods for Lake at two embedding rates of 0.0156 and 0.0625

Method	Error rate (%)		PSNR (dB)			
			Decrypted image		Recovered image	
	0.0156	0.0625	0.0156	0.0625	0.0156	0.0625
Zhang (2011)	5.4440	26.843	37.92	37.90	47.73	40.80
Hong *et al.* (2012)	1.3670	10.535	37.90	37.90	53.67	44.97
Liao and Shu (2015)	2.3440	13.930	37.92	37.89	51.45	43.77
Wu and Sun (2014)	0.0002	0.032	30.64	27.06	63.76	59.57
Proposed method	0	0	27.07/48.35*	21.04/42.57*	∞	∞

* The first value is the PSNR of the directly decrypted image, and the second value is the PSNR of the decrypted image after interpolation

parameters. Finally, the PSNR of the recovered image obtained using the proposed method is infinite, better than those obtained using the other methods.

We can see that, the proposed method can extract the data free of error and completely recover the image. In Table 6, we compare our proposed method only with methods proposed by Zhang (2011), Hong *et al.* (2012), Wu and Sun (2014), and Liao and Shu (2015), and only in the case of the embedding rate equal to 0.0625. This is because Baboon has a long location map. The number of pixels of bad correlation is 294 (Table 1). The number of bits of the side information is s=18b+32=5324. If we set the embedding rate to 0.0156, the embedding capacity is 4096 bits, which will be smaller than s. Therefore, we just compare our method in the case of the embedding rate equal to 0.0625.

6.3 Computational complexity analysis

Finally, we measure the computational complexity of the proposed scheme and how it allocates the complexity to the three major steps of encryption, data hiding, and recovery. The three steps correspond to the three parties: content owner, data hider, and receiver (Fig. 1). The encryption step includes location map generation and image encryption, done by the content owner. All the available white pixels are used for embedding, which means that the embedding capacity is the largest in this measurement. The recovery step includes data extraction and image recovery after image decryption, operated by the receiver. Table 8 shows the average computation time for the three steps on the eight test images. It is obvious that the data hiding step costs much less time relative to the encryption and recovery steps, implying that the data hider has fewer loads during the whole process. Therefore, our proposed scheme is ideal for service providers to deal with encrypted images where the primary manipulation is data hiding.

Table 8 Average computation time for the three steps on the eight test images

Step	Average computation time (s)
Encryption	1.3647
Data hiding	0.0180
Recovery	1.3530

7 Conclusions

It is an important issue to have a data hiding scheme for encrypted images that can extract the

secret data and restore the original image without any loss of information. In this paper, we proposed a reversible data hiding method for encrypted images. We introduced the interpolation technique to generate a location map to record whether the pixels could be embedded or not, and then embedded the additional data through modifying the MSB. Both the embedding rate and embedding capacity were improved significantly. In addition, as the pixels of bad correlation were eliminated, real reversibility is realized using our method, which means both data extraction and image decryption are free of error.

References

An, L., Gao, X., Yuan, Y., 2012a. Robust lossless data hiding using clustering and statistical quantity histogram. *Neurocomputing*, **77**(1):1-11. https://doi.org/10.1016/j.neucom.2011.06.012

An, L., Gao, X., Li, X., 2012b. Encryption. Robust reversible watermarking via clustering and enhanced pixel-wise masking. *IEEE Trans. Image Process.*, **21**(8):3598-3611. https://doi.org/TIP.2012.2191564

Cao, X., Du, L., Wei, X., *et al*., 2016. High capacity reversible data hiding in encrypted images by patch-level sparse representation. *IEEE Trans. Cybern.*, **46**(5):1132-1143. https://doi.org/10.1109/TCYB.2015.2423678

Celik, M.U., Sharma, G., Tekalp, A.M., 2005. Lossless generalized-LSB data embedding. *IEEE Trans. Image Process.*, **14**(2):253-266. https://doi.org/10.1109/TIP.2004.840686

Dragoi, I.C., Coltuc, D., 2015. On local prediction based reversible watermarking. *IEEE Trans. Image Process.*, **24**(4):1244-1246. https://doi.org/10.1109/TIP.2015.2395724

Fridrich, J., Goljan, M., Du, R., 2001. Invertible authentication. *SPIE*, **4314**:197-208. https://doi.org/10.1117/12.435400

Hong, W., Chen, T.S., Wu, H.Y., 2012. An improved reversible data hiding in encrypted images using side match. *IEEE Signal Process. Lett.*, **19**(4):199-202. https://doi.org/10.1109/LSP.2012.2187334

Hu, Y., Lee, H.K., Li, J., 2009. DE-based reversible data hiding with improved overflow location map. *IEEE Trans. Circ. Syst. Video Technol.*, **19**(2):250-260. https://doi.org/10.1109/TCSVT.2008.2009252

Lee, C.F., Chen, H.L., 2012. Adjustable prediction-based reversible data hiding. *Dig. Signal Process.*, **22**(6):941-953. https://doi.org/10.1016/j.dsp.2012.05.015

Li, X., Zhang, W., Gui, X., *et al*., 2015. Efficient reversible data hiding based on multiple histograms modification. *IEEE Trans. Inform. Forens. Secur.*, **10**(9):2016-2027. https://doi.org/10.1109/TIFS.2015.2444354

Liao, X., Shu, C., 2015. Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels. *J. Vis. Commun. Image Represent.*, **28**(2):21-27. https://doi.org/10.1016/j.jvcir.2014.12.007

Luo, L., Chen, Z., Chen, M., 2010. Reversible image watermarking using interpolation technique. *IEEE Trans. Inform. Forens. Secur.*, **5**(1):187-193. https://doi.org/10.1109/TIFS.2009.2035975

Ma, K., Zhang, W., Zhao, X., *et al*., 2013. Reversible data hiding in encrypted images by reserving room before encryption. *IEEE Trans. Inform. Forens. Secur.*, **8**(3):553-562. https://doi.org/10.1109/TIFS.2013.2248725

Ni, Z., Shi, Y.Q., Ansari, N., 2006. Reversible data hiding. *IEEE Trans. Circ. Syst. Video Technol.*, **16**(3):354-362. https://doi.org/10.1109/TCSVT.2006.869964

Peng, F., Li, X., Yang, B., 2014. Improved PVO-based reversible data hiding. *Dig. Signal Process.*, **25**(1):255-265. https://doi.org/10.1016/j.dsp.2013.11.002

Qian, Z., Zhang, X., 2016. Reversible data hiding in encrypted image with distributed source encoding. *IEEE Trans. Circ. Syst. Video Technol.*, **26**(4):636-646. https://doi.org/10.1109/TCSVT.2015.2418611

Qian, Z., Zhang, X., Wang, S., 2013. Reversible data hiding in encrypted JPEG bitstream. *IEEE Trans. Multim.*, **15**(2):316-325. https://doi.org/10.1109/TMM.2014.2316154

Thodi, D.M., Rodriguez, J.J., 2007. Expansion embedding techniques for reversible watermarking. *IEEE Trans. Image Process.*, **16**(3):721-730. https://doi.org/10.1109/TIP.2006.891046

Tian, J., 2003. Reversible data embedding using a difference expansion. *IEEE Trans. Circ. Syst. Video Technol.*, **13**(8):890-896. https://doi.org/10.1109/TCSVT.2003.815962

Wu, H.T., Dugelay, J.L., Shi, Y.Q., 2015. Reversible image data hiding with contrast enhancement. *IEEE Signal Process. Lett.*, **22**(1):81-85. https://doi.org/10.1109/LSP.2014.2346989

Wu, X., Sun, W., 2014. High-capacity reversible data hiding in encrypted images by prediction error. *Signal Process.*, **104**:387-400. https://doi.org/10.1016/j.sigpro.2014.04.032

Zeng, X., Pan, X., Ping, L., *et al*., 2010. Robust lossless data hiding scheme. *J. Zhejiang Univ.-Sci. C (Comput. & Electron.)*, **11**(2):101-110. https://doi.org/10.1631/jzus.C0910177

Zhang, W., Ma, K., Yu, N., 2014. Reversibility improved data hiding in encrypted images. *Signal Process.*, **94**(1):118-127. https://doi.org/10.1016/j.sigpro.2013.06.023

Zhang, X., 2011. Reversible data hiding in encrypted image. *IEEE Signal Process. Lett.*, **18**(4):255-258. https://doi.org/10.1109/LSP.2011.2114651

Zhang, X., 2012. Separable reversible data hiding in encrypted image. *IEEE Trans. Inform. Forens. Secur.*, **7**(2):526-532. https://doi.org/10.1109/TIFS.2011.2176120

Zhang, X., Qian, Z., Feng, G., *et al*., 2014. Efficient reversible data hiding in encrypted images. *J. Vis. Commun. Image Represent.*, **25**(2):322-328. https://doi.org/10.1016/j.jvcir.2013.11.001