Improved Reversible Data Hiding in Encrypted Images using Histogram Modification

Shuang Yi, Yicong Zhou∗
Department of Computer and Information Science
University of Macau, Macau 999078, China
Email: ∗yicongzhou@umac.mo.

Abstract—Inspired by Zhang et al.’s method that applies the integer discrete wavelet transform (DWT) to the original image, and embeds the secret data into the middle (LH, HL) and high (HH) frequency sub-bands of integer DWT coefficients with histogram modification based method, we propose a reversible data hiding method that embeds the secret data into the encrypted prediction error values. Compared with the LH, HL and HH integer DWT coefficients, the prediction error values generated by our proposed method are more concentrated to 0, and thus a high visual quality of the marked decrypted image can be achieved. Experimental results show that our proposed method has a better performance than Zhang’s.

I. INTRODUCTION

Reversible data hiding (RDH) in images aims to hide the secret data into an original image by slightly modifying the pixel values, and to perfectly recover the original image after the secret data have been extracted. Many RDH methods have been proposed such as difference expansion (DE) [1], histogram shifting (HS) [2], prediction error expansion [3], [4]. In recent years, RDH in encrypted images (RDHEI) has caught many researchers’ attention. The original image is encrypted by content provider and the secret data is embedded by data hider, and the data hider has no knowledge about the encrypted image. The receiver can obtain different contents (secret data, original image or both) with different access rights. Due to the property of reversibility, RDHEI can be used for many applications such as cloud storage and medical image management system, where the original images are unwilling to disclose to the cloud provider or system administrator. For example, to prevent any unauthorized access, the content owner encrypts his personal images before store them to the Cloud. The Cloud provider who manage the resources may have a better performance than Zhang's.

Li et al. [7] proposed a RDHEI method by embedding one bit of the secret data in one selected pixel by flipping the 3 LSBs. The four neighboring pixels of each selected pixel will keep unchanged to ensure that the selected pixels can be accurately predicted. In Zhou et al.’s method [8], n(n ⩾ 1) bits of the secret data are embedded in one image block using the public key cryptography. And the support vector machine (SVM) is utilized to determine whether an image block is successfully decrypted. In [9] and [10], two separable RDHEI methods are proposed by compression the a number of the LSB planes and the 4th LSB plane for reserving spare space to embed secret data. Yin et al. [11] encrypt the original image using a coarse-grained encryption to permute the blocks in a global image and a fine-grained encryption to permute the pixels in each block. Then two pixel are randomly selected form each block to indicate the peak points, the secret data is then embedded into the pixels that equals to the either of the peak points using the histogram modification method.

For the VRAE methods, the data extraction and image recovering procedures are mainly accomplished by measuring the smoothness of the recovered image, thus they may suffer from incorrectly extraction of secret data and/or original image, and the embedding rate is relatively low. In order to solve these problems, Ma et al. [12] first proposed a RRBE method, which suggests reserving spare space from the original image before encryption, and embedding the secret data into the reserved spare space. Mathew et al. [13] proposed an active block exchange strategy so that the quality of the marked decrypted image can be improved. Zhang et al. [14] randomly select a number of the secret data and obtain their estimation-error and embed the secret data into the encrypted estimation-error values. In Zhang et al.’s method in [15], the integer discrete wavelet transform (DWT) is first applied to the original image, the secret data is then embedded into the middle (LH, HL) and high (HH) frequency sub-bands of integer DWT coefficients with histogram modification based method. Cao et al. [16] reserve the spare space by using the sparse representation method.

Inspired by the method in [15], in this paper, we propose a RDHEI method by using 1/4 of the pixels in original image to predict the remaining 3/4 pixels, and the secret data is then embedded into the encrypted prediction-error values. Compared with the LH, HL and HH of integer DWT...
coefficients, our proposed method can generate the prediction error values that are more concentrated to 0, and thus a high visual quality of the marked decrypted image can be achieved.

The rest of this paper is organized as follows: Section II will introduce our proposed algorithm. Section III will show some simulation results and the conclusion will be drawn in Section IV.

II. PROPOSED ALGORITHM

The framework of the proposed algorithm is shown in Fig. 1. It consists of three phases: image encryption, data embedding, and data extraction/image recovering. The content owner first uses 1/4 of the pixels in the original image to predict its remaining 3/4 pixels and obtains the prediction-error values, and then uses 1/4 of the pixels in the original image to predict its unchanged pixels and the prediction-error values separately by using the encryption key \( K_E \). Then the data hider embeds secret data into the prediction-error values using the data hiding key \( K_H \). At the receiver side, one can obtain the secret data or marked decrypted image which is similar to the original image using \((K_E, K_H)\) or \((K_S, K_E)\), respectively. And if three keys \((K_S, K_H, K_E)\) are held, the receiver can extract the secret data and perfectly recover the original image.

A. Image Encryption

Assume that an original image \( I \) is with a size of \( M \times N \), and the pixel values are within a range of \([0, 255]\). Firstly, decompose the image \( I \) to form four sub-images, namely \( I_1, I_2, I_3 \) and \( I_4 \), by

\[
\begin{align*}
I_1^{(i,j)} &= I(2i-1, 2j-1) \\
I_2^{(i,j)} &= I(2i-1, 2j) \\
I_3^{(i,j)} &= I(2i, 2j-1) \\
I_4^{(i,j)} &= I(2i, 2j)
\end{align*}
\]

where \( 1 \leq i \leq M/2 \) and \( 1 \leq j \leq N/2 \). The illustration of image decomposition is shown in Fig. 2. Then use the pixels in \( I_1 \) to predict the pixels in the rest three sub-images by

\[
I_2^{(i,j)} = \begin{cases} 
\frac{(I_1^{(i,j)} + I_4^{(i-1,j+1)})}{2} & \text{if } i < N/2 \\
I_4^{(i,j)} & \text{if } i = N/2
\end{cases}
\]

\[
I_3^{(i,j)} = \begin{cases} 
\frac{(I_1^{(i,j)} + I_3^{(i,j+1)})}{2} & \text{if } i < N/2 \\
I_3^{(i,j)} & \text{if } i = N/2
\end{cases}
\]

\[
I_4^{(i,j)} = \begin{cases} 
\frac{(I_1^{(i,j)} + I_4^{(i-1,j+1)})}{2} & \text{if } i < N/2 \\
I_4^{(i,j)} & \text{if } i = N/2
\end{cases}
\]

(1)

where

\[
I_2^{(i,j)} = \begin{cases} 
\frac{(I_1^{(i,j)} + I_4^{(i+1,j-1)})}{2} & \text{if } i < M/2, j < N/2 \\
I_2^{(i,j)} & \text{if } i = M/2, j < N/2 \\
I_2^{(i,j)} & \text{if } i < M/2, j = N/2 \\
I_2^{(i,j)} & \text{if } i = M/2, j = N/2
\end{cases}
\]

(2)

\[
H_1^{(i,j)} = \begin{cases} 
I_1^{(i,j)} - I_1^{(i+1,j+1)} & \text{if } i < M/2, j < N/2 \\
H_1^{(i,j)} & \text{if } i = M/2, j < N/2 \\
H_1^{(i,j)} & \text{if } i < M/2, j = N/2 \\
H_1^{(i,j)} & \text{if } i = M/2, j = N/2
\end{cases}
\]

(3)

and

\[
H_2^{(i,j)} = \begin{cases} 
I_2^{(i,j)} - I_2^{(i+1,j+1)} & \text{if } i < M/2, j < N/2 \\
H_2^{(i,j)} & \text{if } i = M/2, j < N/2 \\
H_2^{(i,j)} & \text{if } i < M/2, j = N/2 \\
H_2^{(i,j)} & \text{if } i = M/2, j = N/2
\end{cases}
\]

(4)

Thus we can obtain the prediction-error values of these three sub-images by

\[
\begin{align*}
E_1^{(i,j)} &= I_1^{(i,j)} - H_1^{(i,j)} \\
E_2^{(i,j)} &= I_2^{(i,j)} - H_2^{(i,j)} \\
E_3^{(i,j)} &= I_3^{(i,j)} - H_3^{(i,j)}
\end{align*}
\]

(5)

Obviously, the prediction-error can be positive or negative values. We use the most significant bit (MSB) to represent the sign bit (e.g., 0 for positive value and 1 for negative value)
and the rest 7 bits to store its absolute value. Thus, only the prediction-error values fall into the range of \([-127, 127]\) can be successfully stored with 8 bits. For the prediction error values out of this data range, we record them into a overflow map \(O_2\) and embed it into the sub-image \(I_1\) using the RDH method [4].

Next, we encrypt the four sub-images separately. For \(I_1\), we encrypt it by

\[
E_1 = I_1 \oplus R
\]

where ‘\(\oplus\)’ is a bit-level XOR operation and \(R\) is a \(256 \times 256\) random matrix generated by the encryption key \(K_E\), and \(R^{(1,3)} \in [0, 255]\). Here, \(R\) can be generated by secure chaotic systems such as [17], [18]. For \(e_3, e_3\) and \(e_4\), we permutate them to form the encrypted sub-images \(E_2, E_3\), and \(E_4\) using \(K_E\). Finally, we concatenate \(E_1, E_2, E_3\), and \(E_4\) to generate the final encrypted image \(E\).

B. Data Embedding

After obtained the encrypted image, the data hider uses the data hiding key \(K_H\) to encrypt the secret data and embeds the encrypted data into the sub-images \(E_2, E_3\), and \(E_4\) using histogram modification method. The detailed histogram modification based data embedding procedures are list as follows:

Step 1: According to the data hiding key \(K_H\), permute \(s\) elements in sub-images \(E_2, E_3\), and \(E_4\) and form a pixel sequence. Keep the first 500 pixels in the sequence and denote the remaining \(K = (3M N / 4) - 500\) pixels as a sub-sequence \((E'_2)^{E_1}_{C \times MN, 500}\). Let \(b(k)\) be the histogram when prediction error values in the sequence \((E'_2)^{E_1}_{C \times MN, 500}\) equals to \(k\).

Step 2: Given a \(C\)-bits encrypted secret data, concatenate it with LSBs of preselected 500 pixels to form the payload \(P\), thus the length of \(P\) is \((C + 500)\) bits. Then we use the center \(p\) pairs of histograms of \((E'_2)^{E_1}_{C \times MN, 500}\) to embed the payload, where \(p\) is calculated by

\[
p = \arg \min_{k=1, 2, \ldots} \sum_{k=1}^{500} b(e) \geq C + 500, k = 1, 2, \ldots\]

Next, divide \((E'_2)^{E_1}_{C \times MN, 500}\) into two sub-sequences: \((E'_2)^{E_1}_{C \times MN, 500, 250}\) and \((E'_2)^{E_1}_{C \times MN, 250}\) that denote the first \(U\) elements and the remaining part of \((E'_2)^{E_1}_{C \times MN, 500}\), respectively. Here \((E'_2)^{E_1}_{C \times MN, 500, 250}\) satisfies that the number of elements falling into the range of \([-2p, 2p]\) equals to \((C + 500)\), which means that by only shifting or modifying the elements in \((E'_2)^{E_1}_{C \times MN, 250}\), we can embed the whole payload bits, and pixels in \((E'_2)^{E_1}_{C \times MN, 250}\) will keep unchanged. Thus we embed the payload into \((E'_2)^{E_1}_{C \times MN, 250}\) by

\[
E'_2 = \begin{cases} \frac{E'_2 + p}{2} & \text{if } E'_2 \geq p \\ \frac{E'_2 - p}{2} & \text{if } E'_2 < -p \\ m & \text{otherwise} \end{cases}
\]

where \(m \in \{0, 1\}\) is one bit of the secret data.

Step 3: After embedded the payload bits, \(E'_2\) may fall out of the range of \([-127, 127]\). Thus, we store these overflow information into an overflow map \(O_2\) and embed it into the preselected 500 pixels by simply LSB substitution. The parameters \(p\) and \(U\) also need to store with \(O_2\) since they will be utilized for secret data extraction at the receiver side.

Step 4: Concatenate \((E'_2)^{E_1}_{C \times MN, 250}\) with \((E'_3)^{E_1}_{C \times MN, 250}\) to form the marked encrypted sequence \((E'_2)^{E_1}_{C \times MN, 500}\) which contains payload bits. According to the data hiding key \(K_H\), inversely permutate \((E'_2)^{E_1}_{C \times MN, 500}\) and \((E'_3)^{E_1}_{C \times MN, 500}\) and form 3 sub-images and put them to their original positions. Finally, scramble all pixels in the image using the sharing key \(K_S\) to generate the final marked encrypted image \(D\).

C. Data Extraction and Image Recovering

At the receiver side, with different combinations of security keys, the receiver can obtain different contents separately. With the keys \((K_S, K_H)\), one can extract the secret data from the marked encrypted image, and if \(K_E\) is also held, the original image can be further obtained. If the receiver holds only the keys \((K_S, K_H)\), he/she can obtain a marked encrypted image which is similar to the original one. Next, we present the data extraction and image recovering procedures separately.

1) Data extraction: Using the sharing key \(K_S\), firstly, inversely scramble the marked encrypted image, and denote the obtained image as \(D_S\). Then extract the encrypted sub-image \(E_1\) from \(D_S\), permute the remaining \((3M N / 4)\) pixels and form a pixel sequence based on the data hiding key \(K_H\). Extract the overflow map \(O_2\), if any, parameters \(p\) and \(U\) from the LSBs of the first 500 pixels of the sequence, and denote the remaining pixel sub-sequence as \((E'_2)^{E_1}_{C \times MN, 500}\). Separate \((E'_2)^{E_1}_{C \times MN, 500}\) into \((E'_2)^{E_1}_{C \times MN, 500, 250}\) and \((E'_2)^{E_1}_{C \times MN, 250}\), respectively. According to the parameter \(p\), extract the payload bits from \((E'_2)^{E_1}_{C \times MN, 250}\) by

\[
m = \frac{E'_2 + p}{2} - \frac{E'_2 - p}{2}, \quad \text{for} \quad -2p \leq E'_2 < 2p
\]

Finally, the receiver obtains the encrypted secret data from payload and decrypts it using the data hiding key \(K_H\) and obtains the plain secret data.

2) Image recovering: After obtained the secret data, if the receiver also holds \(K_E\), he/she can further recover the original image. Firstly, recover the pixel values in \((E'_2)^{E_1}_{C \times MN, 500}\) by

\[
E'_2 = \begin{cases} \frac{E'_2 + p}{2} & \text{if } E'_2 < -2p \\ \frac{E'_2 - p}{2} & \text{if } E'_2 \geq 2p \\ E'_2 / 2, & \text{if } -2p \leq E'_2 < 2p \end{cases}
\]

Next, replace the LSBs of the preselected 500 pixels by extracted payload bits. According to \(K_H\) and \(K_E\), inversely permute these 500 pixels. \((E'_2)^{E_1}_{C \times MN, 500}\) to \((E'_2)^{E_1}_{C \times MN, 500, 250}\) and \((E'_2)^{E_1}_{C \times MN, 250}\) form three sub-images: \(e_3, e_3\), and \(e_4\). Then decrypt \(E_1\) using Eq. (13) to obtain the decrypted sub-image \(I_1\).

\[
I_1 = E_1 \oplus R
\]

where \(R\) is generated by the same way as in image encryption process. Next, extract overflow map \(O_1\), if any, from \(I_1\), and recover the overflow values in \(e_2, e_3\), and \(e_4\). According to Eqs. (2)-(4), obtain the prediction-error sub-images \(I_2, I_3\) and \(I_4\).
Fig. 3. (a) The original image; (b) the marked encrypted image with embedding rate = 0.1777 bpp, p = 1; (c) the marked decrypted image with PSNR=49.929 dB and (d) the recovered image.

Fig. 4. PSNR comparisons of the marked decrypted images generated by the proposed and other RDHEI methods. (a) Lena; (b) Peppers; (c) Boat.

$\hat{I}_4$, recover the original sub-images $I_2$, $I_3$ and $I_4$ by

$$\begin{align*}
I_2^{(1)} & = I_2^{(0)} + \hat{e}_2^{(1)} \\
I_3^{(1)} & = I_3^{(0)} + \hat{e}_3^{(1)} \\
I_4^{(1)} & = I_4^{(0)} + \hat{e}_4^{(1)}
\end{align*}$$

Finally, put the pixels in four sub-images to their original image pixel locations, thus, the recovered image is generated.

If only the keys $\{K_S, K_E\}$ are held, the receiver can obtain the marked decrypted image which is similar to the original one. Firstly, inversely permute the marked encrypted image using the sharing key $K_S$, the obtained image is denoted as $D_r$. Then divide $D_r$ into four parts, the encrypted sub-image $E_1$ and three encrypted prediction error sub-images $E_2$, $E_3$ and $E_4$. Decrypt these four sub-images separately. For $E_1$, decrypt it using Eq. (13), the decrypted sub-image is denoted as $I_1$. For other three sub-images, inversely permute them using the image encryption key $K_E$ and generate three sub-images $e_2, e_3$ and $e_4$. Next, according to Eqs. (2)-(4), obtain the prediction-error sub-images $\hat{I}_2, \hat{I}_3$ and $\hat{I}_4$, and recover the original sub-images $I_2, I_3$ and $I_4$ by Eq. (14). Finally, put the pixels in four sub-images to their original locations, and the generated marked encrypted image is denoted by $I_m$. Note that the pixel values in the marked encrypted image $I_m$ may exceed the data range of $[0, 255]$, we simply set the pixel value to 0 (255) if it is less (larger) than 0 (255).

III. SIMULATION RESULTS

All images used in our simulation results are with the size of $512 \times 512$, and their pixel values fall into $[0, 255]$.

Fig. 3 shows the experimental results of the proposed algorithm using the standard test image Lena which selected from the Miscellaneous database, the parameter $p = 1$ and the embedding rate $r = 0.1777$ bpp. From the results we can observe that the marked decrypted image has high visual quality (peak signal to noise ratio (PSNR)=49.929 dB) when ER=0.1777 bpp, the recovered image is exactly the same with the original image due to the reversibility of the proposed algorithm.

We select three commonly used standard test images, Lena, Peppers and Boat from the Miscellaneous database to show the quality of the marked decrypted images generated by proposed method and Zhang et al.’s [15] method. As can been seen from Fig. 4, the PSNR results of our proposed method outperform Zhang et al.’s [15] at different embedding rates.

We randomly selected 200 images in BOWSBase to show the PNSR results of marked decrypted images generated by proposed and Zhang et al.’s [15] methods at different

2 http://bows2.cs-lille.fr/.
embedding rates. The results are plotted in Fig. 5. From the results we can observe that our proposed method has higher PSNR results than Zhang et al.'s at different embedding rates. The average PSNR results of these 200 images at different embedding rates are listed in Table I. One can observe that, compared with Zhang et al.'s method, the average PSNRs of our proposed algorithm improved about 2.5 dB on average.

<table>
<thead>
<tr>
<th>TABLE I</th>
<th>AVERAGE PSNRs OF 200 MARKED ENCRYPTED IMAGE GENERATED BY THE PROPOSED AND ZHANG ET AL.'S METHODS AT DIFFERENT EMBEDDING RATES (ER).</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR (dB)</td>
<td>Proposed method</td>
</tr>
<tr>
<td>0.1 bpp</td>
<td>35.0874</td>
</tr>
<tr>
<td>0.2 bpp</td>
<td>32.7828</td>
</tr>
<tr>
<td>0.3 bpp</td>
<td>30.5866</td>
</tr>
</tbody>
</table>

IV. CONCLUSION

This paper proposed a new RDHEI method by using 1/4 of the pixels in original image to predict the remaining 3/4 of the pixels, and embedding the secret data bits into the encrypted prediction error values. It can achieve full reversibility, data extraction and image recovery procedures can be accomplished separately by using different combinations of security keys. Experimental results have shown that our proposed method has better performance than Zhang et al.'s [15].

ACKNOWLEDGMENT

This work was supported in part by the Macau Science and Technology Development Fund under Grant FDCT/2016/2015/A1 and by the Research Committee at University of Macau under Grants MYRG2014-00003-FST and MYRG2016-00123-FST.

REFERENCES


