Reversible Data Hiding in Permutation-based Encrypted Images with Strong Privacy

Chih-Wei Shiu [1], Yu-Chi Chen [2] and Wien Hong [3]
[1] Department of Education Industry and Digital Media, National Taitung University
Taitung, 95092-TW
[e-mail: chihwei.shiu@gmail.com]
[2] Computer Science and Engineering, Yuan Ze University
Taoyuan, 11529-TW
[e-mail: wycchen@saturn.yzu.edu.tw]
[3] School of Electrical and Computer Engineering, Nanfang College of Sun Yat- Sen University
Guangzhou, 510970 -CN
[e-mail: wienhong@gmail.com]
*Corresponding author: Chih-Wei Shiu

Received November 15, 2017; revised March 27, 2018; accepted October 17, 2018; published February 28, 2019

Abstract

Reversible data hiding in encrypted images (RDHEI) provides some real-time cloud applications; i.e. the cloud, acting as a data-hider, automatically embeds timestamp in the encrypted image uploaded by a content owner. Many existing methods of RDHEI only satisfy user privacy in which the data-hider does not know the original image, but leaks owner privacy in which the receiver can obtains the original image by decryption and extraction. In the literature, the method of Zhang et al. is the one providing weak content-owner privacy in which the content-owner and data-hider have to share a data-hiding key. In this paper, we take care of the stronger notion, called strong content-owner privacy, and achieve it by presenting a new reversible data hiding in encrypted images. In the proposed method, image decryption and message extraction are separately controlled by different types of keys, and thus such functionalities are decoupled to solve the privacy problem. At the technique level, the original image is segmented along a Hilbert filling curve. To keep image privacy, segments are transformed into an encrypted image by using random permutation. The encrypted image does not reveal significant information about the original one. Data embedment can be realized by using pixel histogram-style hiding, since this property, can be preserved before or after encryption. The proposed method is a modular method to compile some specific reversible data hiding to those in encrypted image with content owner privacy. Finally, our experimental results show that the image quality is 50.85dB when the averaged payload is 0.12bpp.

Keywords: Reversible Data Hiding, Encrypted Image, Histogram, Content Owner Privacy

1. Introduction

Reversible data hiding (RDH) [1]-[7] is currently a popular research topic in information security. In general, RDH involves the following phases. (i) Embedding procedure embeds a message into a cover image to generate the stego image. (ii) Extracting procedure extracts the message from the stego image. (iii) Image recovery procedure can finally recover the original cover image from the stego image. Considering some real-life purposes, the property, reversibility, is very critical if the cover image is meaningful, such as medical imagery, military imagery and law forensics.

In order to enhance protection of the cover image, some prior works [8]-[9] proposed encryption for the cover image, and then relied on a non-reversible data hiding algorithm to embed a message. However, such techniques of [8]-[9] are irreversible. Therefore, an issue, RDH in the encrypted images, is naturally considered to capture more applications. If an encrypted image can be used by RDH, it would be suitable for cloud storage applications. The users only need to encrypt an image and then transmit to the cloud server. Then, the server can embed some messages in the encrypted images for management. With the corresponding decryption and recovery procedures, the original cover image can be recovered.

In some application scenarios, when a patient's medical image with privacy is stored in the image database system, if the medical image is encrypted before storage, the patient privacy can be increased. Then the database system administrator may need to embed some messages for management, such as the origin information, image notation or authentication data [15]. There is also the same architecture of the cloud system. Some steps that are outsourced, such as users must encrypt the image before uploading to cloud database system. Then, the system administrator is responsible for it. At this stage, the manager can embed some messages for management.

However, the traditional RDH is difficult to directly apply in encrypted media. The main reason is that such RDH methods concern image correlation very heavily (so-called the correlation issue, and more details are described below). For example, in 2009, Tai et al. [10] partitioned images into non-overlapping pixel pairs and then based on pixel pair difference value to embed messages. When the difference value is small to a critical section, the difference expansion [11] can work for embedding. However, if the pixels of the encrypted image are too complex, then the difference value will be very large and hard to work for embedding messages.

In 2011, Hong et al. [12] made use of the image's reference pixels to generate an interpolated image. The difference value between cover and interpolation image are calculated and used in histogram shifting [13]. The message is embedded in the difference value. The correlation issue in Hong et al.'s method still exists, which means that the correlation among all the pixels in the encrypted image is not high. Therefore, as the same above, if the difference between the interpolation image and cover image is very large, it cannot be used for embedding.

Observing the correlation issue, it is not easy to use the traditional RDH to embed a message in the encrypted image. A new algorithm is needed to be presented to overcome the correlation

issue.

As a significant breakthrough, Zhang [15] proposed a new reversible data hiding method in encrypted images (RDHEI) in 2011. This method cleverly avoids the traditional method by using XOR operations over the first five MSBs of the cover image and an encryption key to the complete image encryption. In the encrypted image, it is in the clear that the last three LSBs are unchanged before embedding. Then, the encrypted image is partitioned into non-overlapping blocks. The pixels in each block are divided into two groups: A and B. For each bit of the embedding messages, a bit of the last 3 LSBs of one group is flipped if the bit of the message is 1, or not changed if 0 to complete the embedding step. During the message extraction and image recovery, decryption must be performed for the first five MSBs, and then we can execute an evaluation to decide which group (A or B) after flipping was smoother for the block and extract the embedded message by such evaluation result. However, the evaluation may get an error if the partitioned block was too. The follow-up work of Hong et al. [16] slightly improve the evaluation to evaluate complex adjacent pixels. Hong et al.'s method leads to more effective than Zhang's [15] and has a significantly higher precision of correctness. In addition to concrete RDHEI methods, these works of [15] and [16] formally defined the framework of EIRDH, composed of content-owner and data-hider.

Recall the framework (see **Fig. 1**) where the cover image must be entrusted with the data-hider to embed the message due to certain reasons of security. However, if the image provided by the content-owner is very private or sensitive, the data-hider cannot know the entire image. Therefore, the content-owner required an encryption key to encrypt the cover image. The encrypted image is then entrusted with the data-hider to embed the message. Going into more details of the framework, **Fig. 2** shows the process of the receiver with the encryption key and data-hiding key to decrypt the image, extract the message and recover image.

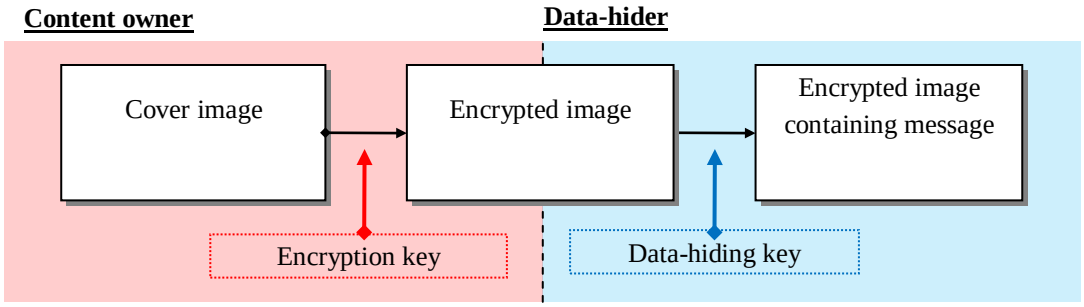


Fig. 1. Content owner and data-hider in the RDHEI framework

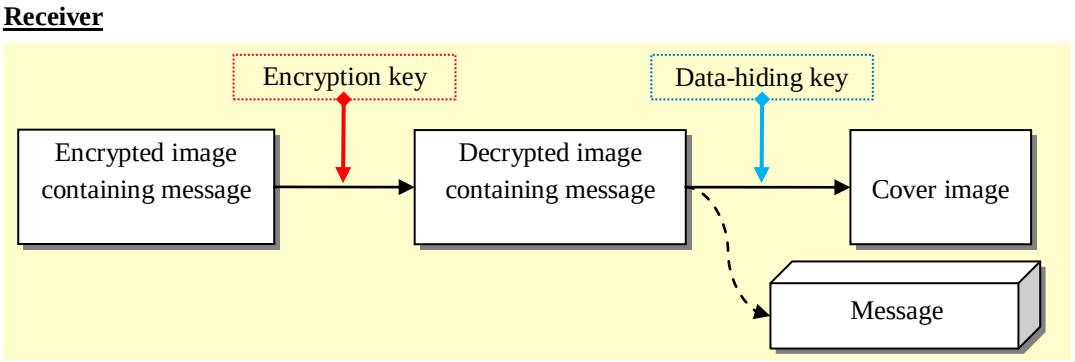


Fig. 2. Receiver in the RDHEI framework

In the first case, the receiver has only the encryption key to decrypt the image. In the second case, the receiver has both the encryption and data-hiding keys. The encryption key is used to directly decrypt the encrypted image, while the data-hiding key is used to identify block's clustering for extracting the message and recover the image. Under the framework presented by [15] and [16], there is no way to extract the message if we do not decrypt the encrypted image.

Acting as an important next-step to improve some weaknesses of RDHEI, Zhang [17] proposed a new framework, called RDHEI-2 (see **Fig. 3**), and accordingly introduced the new matrix operation method. Under the frameworks of RDHEI or RDHEI-2 of [15]-[17], the encryption key is used to decrypt the image. The receiver, thus can see a rough cover image, which implies that the content-owner privacy leaks without doubt.

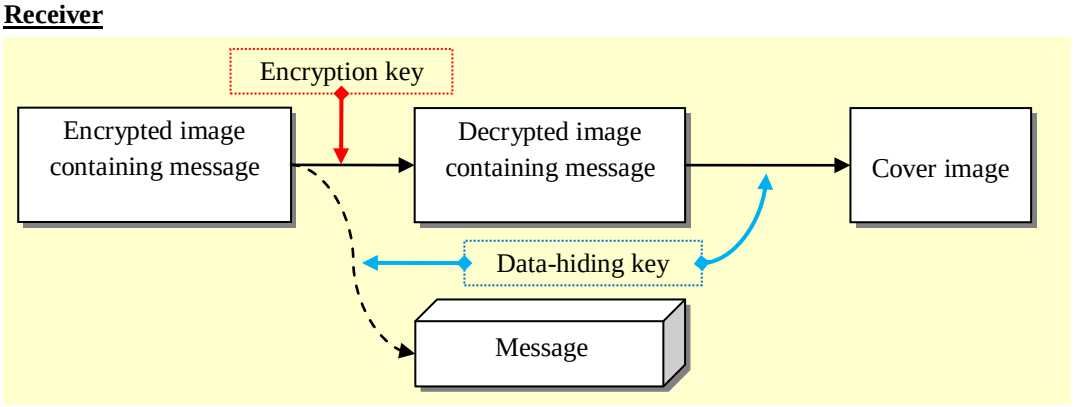


Fig. 3. Receiver in the RDHEI-2 framework

In fact, there is an independent research direction to build RDHEI methods based on public key cryptography system [18]-[20] can deal with different key. JPEG bitstream domain[21]-[22] in different domain conversion encrypted signal to add different applications. [23]-[24]using distributed source encoding and progressive recovery can achieve higher embedding capacity in encrypted images than other method. Those works presented a new framework to meet different purposes. They are out of scope of this paper, but will lead to some new ideas in the future work.

Motivations. Zhang et al. [25] proposed a new method to solve the problem of weak content-owner privacy. The method is not LSB-based hiding, which conducts higher PSNR than [15]-[17] so far. Here, we do not describe the Zhang's method [25], but only highlight the resulting framework (called RDHEI-3). In the new RDHEI-3, the data-hider can independently complete message embedding, and also the receiver holding the data-hiding key is able to extract the message without doing decryption. The only disadvantage of RDHEI-3 is that the content-owner must share such data-hiding key with the data hider. Hence, it gives us a natural question. Our result of this paper stresses on this question.

- *Can we achieve a reversible data hiding in encrypted images with strong content-owner privacy where the content-owner is independent of the data hiding key?*

Our contributions and results. The ideal framework called RDHEI-4, is composed of four entities: content-owner, data-hider, extraction-receiver, and decryption-receiver. The content-owner shares an encryption key to the decryption-receiver, and the data-hider shares a data-hiding key to the extraction-receiver. Such framework can be interpreted as that of **Fig. 4** in which the content-owner performs encryption, the data-hider does message embedding, the extraction-receiver does message extraction, and finally the decryption-receiver does decryption.

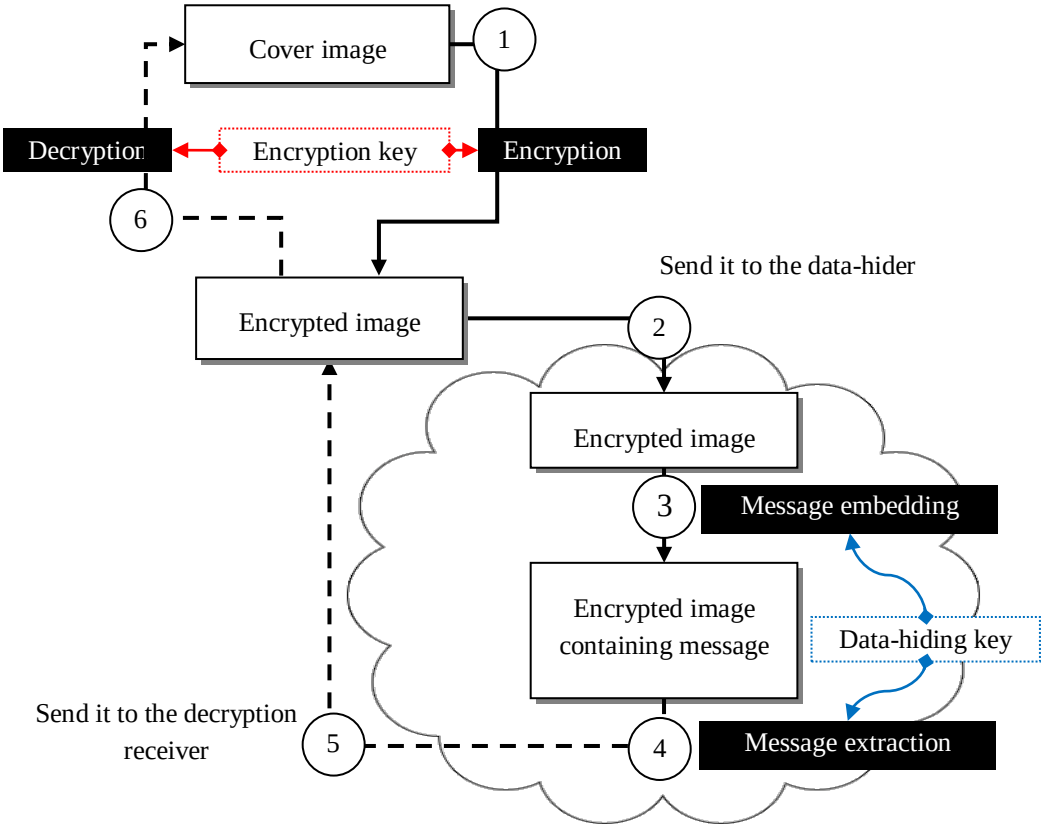


Fig. 4. The proposed framework of RDHEI-4

In the proposed RDHEI-4 method, permutation is applied to encrypt the cover image. Firstly, we will partition the image that into non-overlapping blocks with a Hilbert filling curve path. Secondly, the original image is transformed into an encrypted image by using random permutation. For each block, we rely on [14] to embed messages. We sort pixel value in each block and generating difference value according to their sorting order. Then, can be used some histogram-style technologies to embed messages. Intuitively, this type of encryption maintains the histogram information, so a histogram shifting technique can be used to embed the message. From the experimental results, the PSNR of the decrypted image averaged 50.85dB with full embedding and the average payload is 0.12bpp. Hence, our method is better than current exiting RDHEI methods. In this paper, will describe the required technology in section 2. the details of the proposed method will be explained in Section 3, the architecture of the security analysis in Section 4 and show experimental results in Section 5. The conclusion is given in Section 5. In the rest of this paper, we will directly call RDHEI-4 as RDHEI.

2. Preliminary

In our proposed, we will use the random permutation method of Fisher–Yates shuffle to generate the key, and we will use [14] to embed the message, so before entering the proposed method, we describe the relevant technical background.

2.1 Random Permutation

In the proposed method, we use random permutation to granting the key for the encrypteion process. The method used in the paper is Fisher–Yates shuffle. This is a well known and efficient random permutation method. The basic method given for generating a random permutation of the numbers 1 through *n* as follows.

Step 1: first generate a sequence of 1 to *n*.
Step 2: Then randomly pick two numbers in this sequence.
Step 3: The two picked numbers in step 2 must be selected for the first time. If not, re-execute step 2.
Step 4: Swap the position of each two numbers in the sequence.
Step 5: Repeat steps 2-4 until all values have been selected, or only remains a single number unselected.

2.1 Pixel Value Ordering Reversible Data Hiding

In 2013, Li et al. has proposed a new reversible data hiding method [14]. In [14], first partitioned an image I into $n \times n$ non-overlapping blocks $B = \{b_1, b_2, ..., b_L\}$, where L is the length of the blocks. The size of each block b_i is $n \times n$ and the block b_i represented as $\{b_{i,k}\}_{k=1}^{n \times n}$. The each block is re-ordered according to the pixel value. The result is b_i' and satisfy $b'_{i,k+1} \geq b'_{i,k}$. In [14] will use $b'_{i,2}$ and $b'_{i,n \times n-1}$ as predictors of $b'_{i,1}$ and $b'_{i,n \times n}$. Then calculate the difference value of $b'_{i,1}$ and $b'_{i,n \times n}$ The difference values between the first two pixels and last pixels in the sorted block b_i' are embedded message by histogram shifting. An process according to the same procedure to complete the remainder of the block.

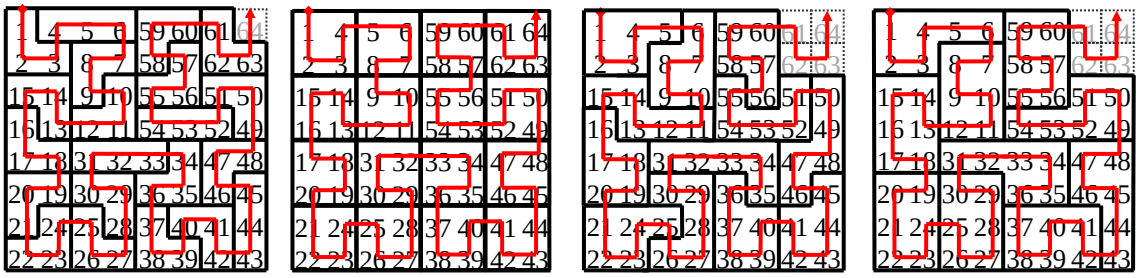
3. The Proposed Method

Now, we will introduce a new reversible data hiding in encrypted image that does not use XOR to do encrypt. At a high level view, our method can be (informally) composed of the following steps. The sequence of pixels is scrambled by random permutation acting as effective encryption. The cover image is divided into several groups initially. Then, following the order of groups, the sequence of the pixels in each group will be randomly permuted. This type of encryption method ensures that before and after permutation, the sequence of the pixels in each group is still identical. This property is suitable for [15]'s algorithm that is as our ingredient for embedding a message. Therefore, most of the traditional RDH can be directly used to provide good payload and image quality. Our result can improve the problems in [15]-[17]. For example, in [15]-[17] the receiver required the encryption and data-hiding keys in order to begin image recovery. On the other hand, the proposed method captures stronger scenario than [21], where the content-owner needs only to send the encrypted image directly to the data-hider. There is no collaboration between the content-owner and data-hider.

3.1 Image Partition

We use the method of [14] to complete the embedding. First, the image is partitioned into non-overlapping blocks. The best cast is that each block are embedded 2 bits. At the same time, each block at the most will have two pixels modified by 1 unit. The size of the block can be used to control payload and image quality. If the block size is smaller, the payload is higher , but image distortion is higher accordingly, and vice versa. As the block size increases, the number of pixels in the block linearly increases as well. Therefore, this selection method is inflexible for control payload and image quality.

In order to resolve the above issue, the partitioning method in [14] is slightly modified. First, let the size of a cover image I be $M \times M$. Next, the Hilbert curve is applied to scan the entire image. For every scanning N pixels, these N- pixel will be grouped into a group (i.e. each group sized is $1 \times N$). Finally, there are L groups $G = \{G_i\}_{i=1}^{L}$, where $L = \lfloor (M \times M)/N \rfloor$. Each group G_i consists of N pixel $\{G_{i,k}\}_{k=1}^{N}$. For example, **Fig. 5** shows how to do scan an 8×8 image for different group sizes.



(a) (b) (c) (d)

Fig. 5. Hilbert curve grouping of various size (a) $N = 3, L = 21$, (b) $N = 4, L = 16$, (c) $N = 5, L = 12$ and (d) $N = 6, L = 10$

3.2 Image Encryption

After the group partition, the encryption key K^e is used to randomly generate 2 orders of random permutation P^A and P^B with lengths N and L, respectively. P^A is used to scramble the sequence pixels in each group. Set $P^A = \{P_k^A \mid P_k^A \in [1, N]\}_{k=1}^N$ where $P_x^A \neq P_y^A \mid x \neq y$. P^B is used to permute the order of the groups. Set $P^B = \{P_i^B \mid P_i^B \in [1, L]\}_{i=1}^L$ where $P_x^B \neq P_y^B \mid x \neq y$. Next, apply Eq.(1) and Eq.(2) to encrypt. After encrypting, group G^E is written back into the image to arrive at encrypted image I^E.

$$G_i^e = \{G_{i,k}^e \mid G_{i,k}^e = G_{i,P_k^A}\}_{k=1}^N, \text{ where } 1 \le i \le L. \quad (1)$$

$$G^E = \{G_i^E \mid G_i^E = G_{P_i^B}^e\}_{i=1}^L. \quad (2)$$

For more details, **Fig. 6(a)** shows the example for encryption. **Fig. 6(b)** shows the result of grouping. Support $P^A = \{1, 3, 2, 5, 4, 6\}$ and $P^B = \{2, 1, 3, 4, 6, 7, 5, 9, 10, 8\}$. Use Eq. (1) to generate **Fig. 6(c)**. Then we apply Eq. (2) to get **Fig. 6(d)**. Finally, the encrypted image is done as **Fig. 6(e)**.

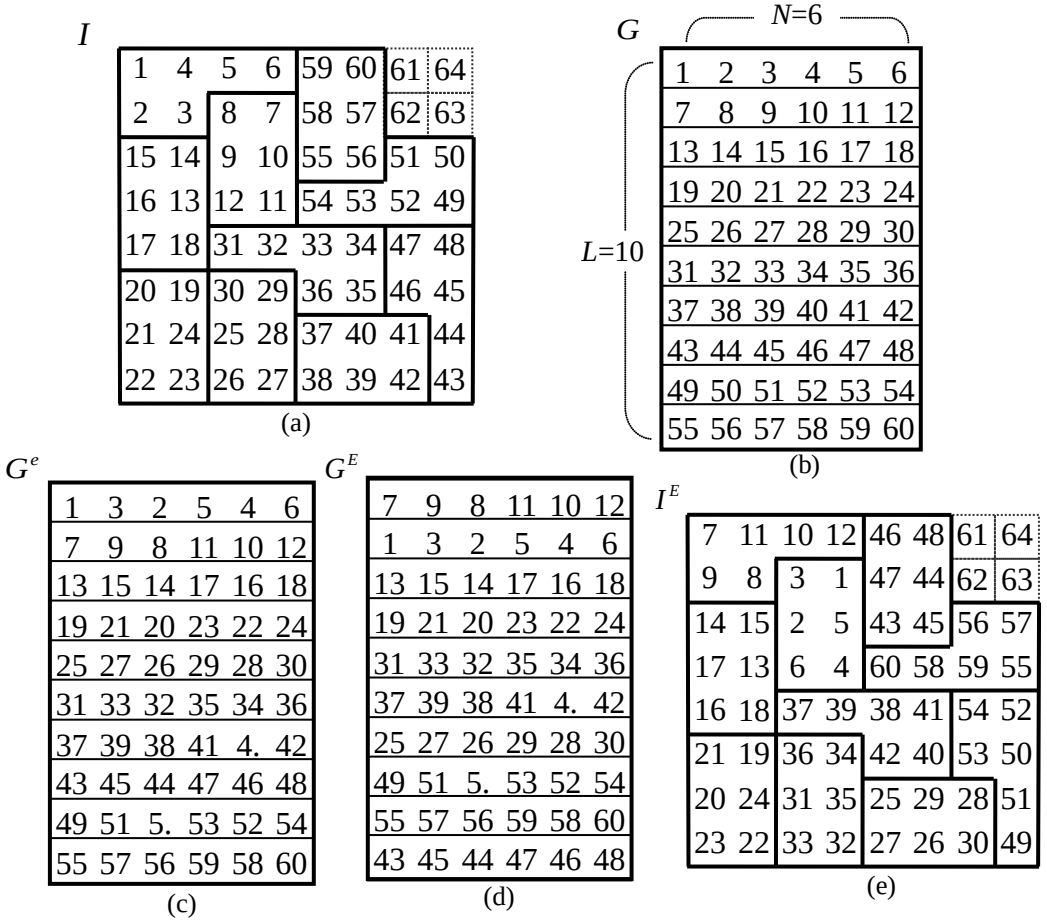
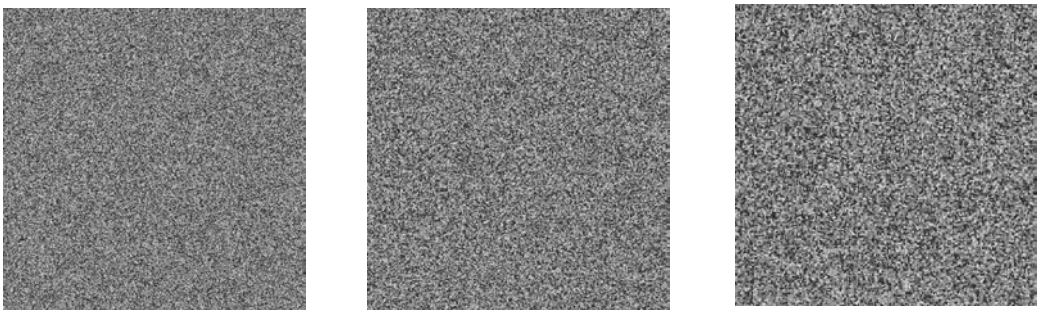


Fig. 6. An example of image encryption

Fig. 7 shows the encryption for Lena. Group sizes are 3, 5 and 10. The encrypted image is completely scrambled and not showing any shape or style of Lena. This means the encryption is effective.



(a) (b) (c)

Fig. 7. Lena using various grouping size of the encryption result (a) $N=3$, (b) $N=5$ and (d) $N=10$

3.3 Embedding procedures

After image encryption, the next step is to embed a message. First, the data-hider uses the data-hiding key K^h to encrypt messages with AES (further increases the security of the message). Next, sequentially extract each group G_i^E from the encrypted image I^E. The pixel values in each group G_i^E are sorted according to the ascending resulting in G_i^S where $G_{i,x}^S \le G_{i,y}^S \mid x < y$. Next, use histogram bin shifting to embed the secret message in the smallest valued $G_{i,1}^S$ and largest $G_{i,N}^S$.

Before embedding, it is necessary to filter out some $G_{i,1}^S$ and $G_{i,N}^S$ whose values are equal to 0 or 255. This is done to trivially avoid over/under flow problems. Next, we also need to filter out the complexity regions so as to increase embedding efficiency. When $(G_{i,N-1}^S - G_{i,2}^S) > TH$, we do not do anything. TH is the threshold and $TH \in [0,255]$. Apply Eq. (3) to record location map *LM*.

$$LM_i = \begin{cases} 0 & \text{if} \quad G_{i,1}^S=0 \text{ or } G_{i,1}^S=255 \text{ or } (G_{i,N-1}^S - G_{i,2}^S) > TH, \\ 0 & \text{if} \quad G_{i,1}^S=0 \text{ or } G_{i,1}^S=255 \text{ or } (G_{i,N-1}^S - G_{i,2}^S) > TH, \\ 1 & \textit{otherwise}, \end{cases} \tag{3}$$

$$\text{where } 1 \le i \le L.$$

After all LMs are recorded and compressed with JBIG-2 coder [26], we embed the compressed codes in the LSB of the first K^Δ-th groups. The number of pixels in the K^Δ-th group must be larger than the length of the compressed codes. The LSB of the K^Δ-th group and the message are concatenated together to be used in the later recovery stage.

Apply Eq.(4) and Eq.(5) to get the difference value(D_i^{Min}, D_i^{Max}) of ($G_{i,1}^S, G_{i,N}^S$). The group size must be $N \geq 3$. This is used to avoid using ($G_{i,1}^S, G_{i,N}^S$) themselves to generate difference value.

$$D_i^{Min} = G_{i,2}^S - G_{i,1}^S, \text{ where } K^\Delta < i \leq L. \quad (4)$$

$$D_i^{Max} = G_{i,N}^S - G_{i,N-1}^S, \text{ where } K^\Delta < i \leq L. \quad (5)$$

If $LM_i = 1$, let D_i^{Min} and D_i^{Max} and apply in Eq. (6)-(7) to embed the message; otherwise (if $LM_i = 0$), then do not do anything.

$$G_{i,1}^{S\prime} = \begin{cases} G_{i,1}^S & \text{if} \quad (D_i^{Min} = 0) \text{ or } (D_i^{Min} = 1 \text{ and } b = 0), \\ G_{i,1}^S - 1 & \text{if} \quad (D_i^{Min} > 1) \text{ or } (D_i^{Min} = 1 \text{ and } b = 1), \end{cases} \quad (6)$$

where $K_J \leq i \leq L$. b is denoted by 1 bit of message.

$$G_{i,N}^{S\prime} = \begin{cases} G_{i,N}^S & \text{if} \quad (D_i^{Max} = 0) \text{ or } (D_i^{Max} = 1 \text{ and } b = 0), \\ G_{i,N}^S + 1 & \text{if} \quad (D_i^{Max} > 1) \text{ or } (D_i^{Max} = 1 \text{ and } b = 1), \end{cases} \quad (5\text{-}7)$$

where $K^\Delta < i \leq L$. b is denoted by 1 bit of message.

Finally, we rewrite all the groups with embedded message back into the encrypted image. Let the encrypted image containing embedded message be $I^{E\prime}$, and (K^h, K^Δ) be the data-hiding key.

3.4 Image decryption, message extraction and image recovery

The receiver has three types of received image, so that we can say three different roles of receivers. Respectively, **type (I):** encrypted image containing embedded message, **type (II):** decrypted image containing embedded message; and **type (III):** encrypted image without embedded message.

The receiver will perform different processes depending on the image condition, and the full flowchart is located in **Fig. 8**. For example, the receiver upon receiving type (I) takes the encryption key and the data-hiding key to begin image decryption or data extraction and image recovery. The result could be either type (II) or (III). If the receiver received image type (II), then the data-hiding key is used to do data extraction and together recover the cover image. If the receiver received image type (III), the encryption key is used to recover to the original image.

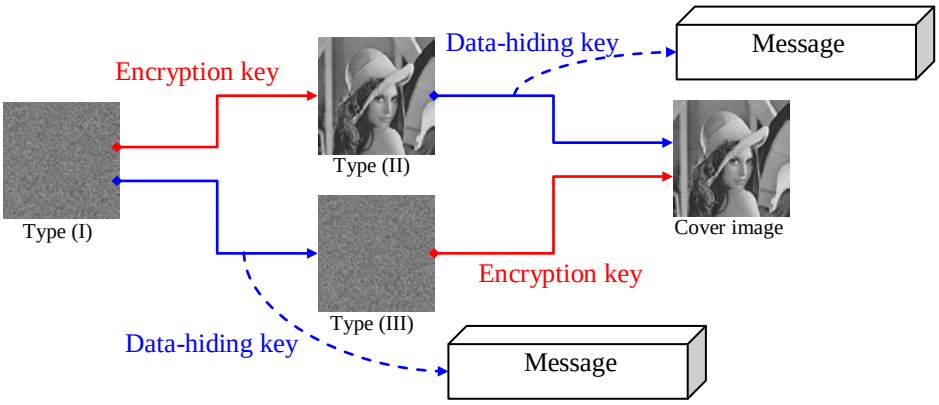


Fig. 8. All cases at receiver side of the proposed method.

Accordingly, the receiver has the encryption-key or data-hiding key, which can roughly be divided into two functions, the image decryption and data extraction.

Red line of Fig. 8. When the receiver performs image decryption, the involved image type is a type (I) or type (III). First, the image is divided into groups as described in Section 2.1. Next, according to the encryption key, we generate two sets by random permutation P^A and P^B. Finally, we can apply Eq. (8) and Eq. (9) to recover the original sequence from the group. The group needs only to be rewritten back to the image to get the decrypted image. The result is either type (II) or the original cover image.

$$G_i^d = \{G_{i,P_k^A}^d \mid G_{i,P_k^A}^d = G_{i,k}^E\}_{k=1}^N, \text{where } 1 \le i \le L. \qquad (8)$$

$$G^D = \{G_{p_i^B}^D \mid G_{p_i^B}^D = G_i^d\}_{i=1}^L. \qquad (9)$$

Blue line of Fig. 8. If the receiver needs to preform data extraction and image recovery simultaneously, the types of processing are Type (I) and Type (II). The image is divided into several groups at first as mentioned in Section 2.1. Each group is sorted in ascending order as $G_i^{S\prime}$ and then we can correctly compute $D_i^{Min\prime} = G_{i,2}^{S\prime} - G_{i,1}^{S\prime}$ and $D_i^{Max\prime} = G_{i,N}^{S\prime} - G_{i,N-1}^{S\prime}$.

The LSB of K^Δ-th group is extracted and decompressed by the JBIG-2 coder to get location map LM. If the image is Type (II), it must go back to Type (I) by encrypting again to ensure that the extracted compressed code and scan order are correct. If $LM_i = 1$, this group contains the embedded message; therefore, $D_i^{Min\prime}$ and $D_i^{Max\prime}$ are substituted into Eq.(10) to extract the message; otherwise ($LM_i = 0$), nothing will be performed.

$$b = \begin{cases} 0 & \text{if} \quad D_i^{Min\prime} = 1 \text{ or } D_i^{Max\prime} = 1, \\ 1 & \text{if} \quad D_i^{Min\prime} = 2 \text{ or } D_i^{Min\prime} = 2, \end{cases} \text{where } K^\Delta < i \le L. \qquad (10)$$

Upon extracting the message, Eq. (11) and Eq. (12) are applied to recover minimum $G_{i,1}^{S\prime}$ and maximum $G_{i,N}^{S\prime}$.

$$G_{i,1}^{S}=\begin{cases}G_{i,1}^{S\prime} & \text{if} \quad (D_i^{Min\prime}=0) \text{ or } (D_i^{Min\prime}=1),\\ G_{i,1}^{S\prime}+1 & \text{if} \quad D_i^{Min\prime}\geq 2,\end{cases} \text{ where } K^{\Delta}<i\leq L. \tag{11}$$

$$G_{i,N}^{S}=\begin{cases}G_{i,N}^{S\ \prime} & \text{if} \quad (D_i^{Max\prime}=0) \text{ or } (D_i^{Max\prime}=1),\\ G_{i,N}^{S\ \prime}-1 & \text{if} \quad D_i^{Max\prime}\geq 2,\end{cases} \text{ where } K^{\Delta}<i\leq L. \tag{12}$$

Finally, we can recover LSB of the previous K^{Δ} group. The group is in sequence rewritten back in the image. The result is either Type (III) or the cover image. The final part of the message is decrypted using key K^{Ω}. This completes data extraction and image recovery.

3. Security Discussion

We divide the security analysis into four perspectives. In the statistical point of view. Zhang [15] and Hong [16] through the XOR generated encrypted image. When the x pixels of the 8 bits are using in XOR operation, it takes up 2^{8x} times to recovery original image. The proposed method is based on the permutation to generate an encrypted image. For simplify the explanation, we do not consider permutations in the image partition procedure. Thus, the permute x pixels is $x!$, where ! represents the factorial operation. Therefore, the relationship between XOR and permutation in mathematics is “power” and “factorial”.

In **Fig. 9**, we show the relationship between $\log(2^{8x})$ and $\log(x!)$, where $\log(\cdot)$ in order to reduce the value to make it easier to draw the situation relationship. From the **Fig. 9** can be found the x greater than 693, the factorial has been greater than the power. This means more the number of pixels, the factorial growth rate is much larger than the power. Indicating that using permutation would be safer to use XOR's encryption method.

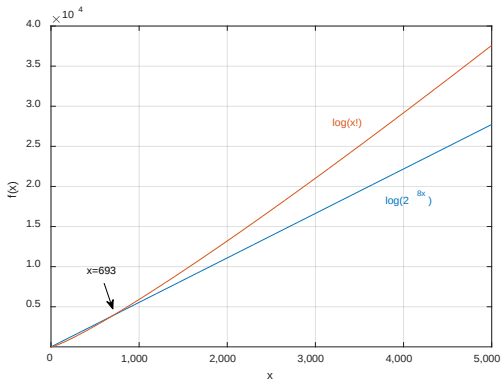


Fig. 9. power and factorial growth curve

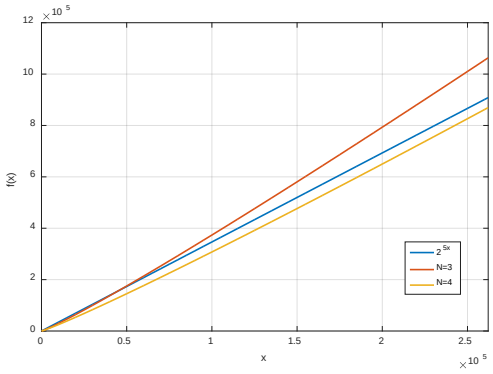


Fig. 10. Proposed and XOR of permutations results

In the actual process, [15] and [16] they only take the first five MSBs for encryption. Therefore, the permutations is 2^{5x}. Our method cannot directly use number of pixel to calculate permutation, must be converted according to group size. Thus the permutations is $N!^L \times L!$, where $L = \lceil x / N \rceil$ is the number of groups。Then, the different number pixel into the formula. In addition, N were set to 3 and 4, the results shown in **Fig. 10**. The results show that when N = 3 can be more secure than [15] - [16], and N = 4 is otherwise. Users can decide different group size depending on security requirements.

In the practical examples, The size of the cover image is 512×512 and group size is 4. According to our formula all the possibility is $4! \times 65536!$. Then, we perform decryption process 10000 times, it takes an average time is 0.2483078 seconds. Therefore, using brute force calculation time will be more than 10,000 years, as follows.

$$\frac{4! \times 65536! \times 0.2483078\text{s}}{60 * 60 * 24 * 365} \approx 9.75647176014910 \times 10^{287186} \text{ years}$$

Even greatly increase the computational efficiency, it still takes a lot of time cost. However, if someone wants to try break through our method, it will take huge computing costs to avoid to damage.

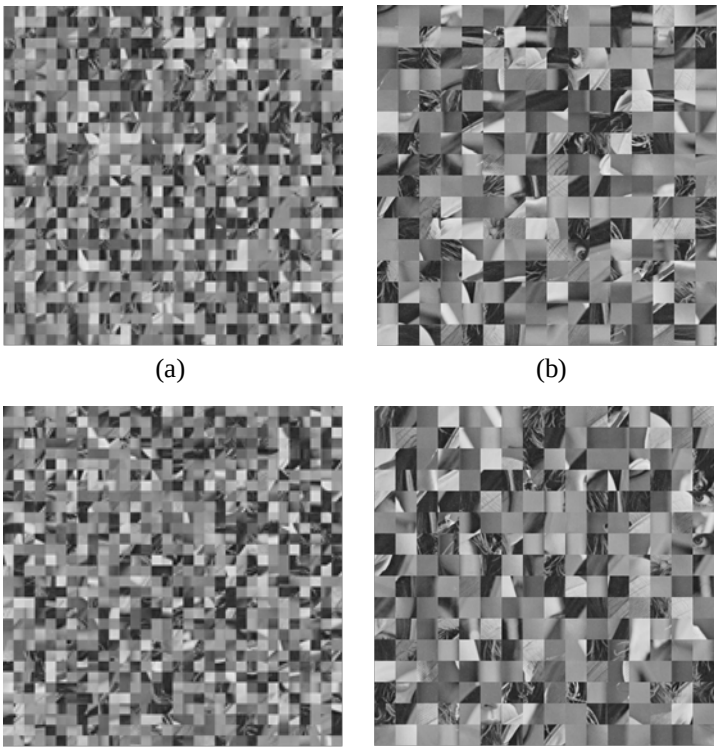
In the second part, we will list the key functions in **Table 1.** Detailed descriptions are shown below. [15]-[16] must correspondingly provide an encryption key K^e and data hiding key K^h. Zheng [17] can achieve separable for extract message, but in recovering image still need used two keys to complete the task which also caused a security problem. Zheng [25] can be accomplished according to a different key. However, when the user independently extracts the message, it needs to obtain from K^e to get vacating room information. The proposed method can be fully independent of the implementation. The receiver can perform different processing depending on the difference image as shown in **Table 1.** In this way, our approach will be more secure on the key management, also provide more application.

Table 1. The receiver procedure of different image type

Image type	Zheng[15]	Hong et al.[16]	Zheng[17]	Zheng[25]	Proposed
Encrypted image containing embedded message	$K^e + K^h$ Decryption, extraction and Recover	$K^e + K^h$ Decryption, extraction and Recover	$K^e + K^h$ Decryption, extraction and Recover	$K^e + K^h$ Decryption, extraction and Recover	$K^e + K^h$ Decryption, extraction and Recover
	K^e Decryption	K^e Decryption	K^e Decryption	K^e Decryption	K^e Decryption
			K^h Extraction	K^h + vacating room information from K^e Extraction	K^h Extraction
Decrypted image containing embedded message				K^h + vacating room information from K^e Extraction and Recover	K^h Extraction and Recover
Encrypted image without embedded message				K^e Decryption and Recover	K^e Decryption and Recover

In the third part, we analyze in the encrypted image of the vision complexity. [29] is based on [30] to design a suitable encryption algorithm. Simultaneously, in their method is based block-based and our method is based on scan order to parting group. When the shape of our group is square, the encrypted image will be very similar to [29], as shown in **Fig. 11.** Which for the simplicity of observation, the block size is set to 16×16 and 32×32 (i.e. **Fig. 11 (a) - (b)**) and proposed group size is set to 256 and 1024(i.e. **Fig. 11 (c) - (d)**).

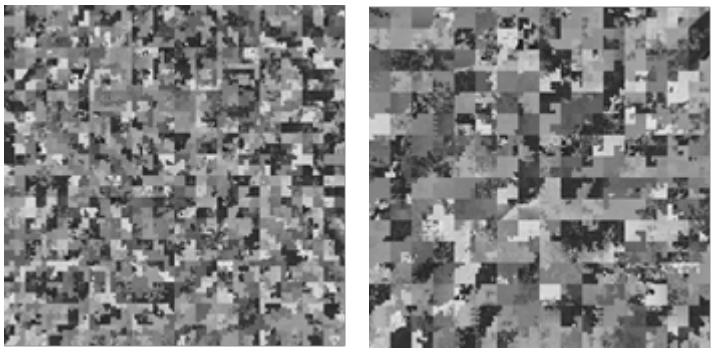
Proposed and [29] when the block is bigger the security will be weaker. The features of the original image are more easily apparent. In our method can be different parameters to avoid. We avoid selecting a parameter to make the group into squares. In this manner, we can reduce the features of the original image as shown in **Fig. 12.** Where we substitute 256 to 257 and 1024 to 1025. As a result, the encrypted image can get messier. Since [29] is based on block, they cannot achieve the same effect.



(a) (b)

(c) (d)

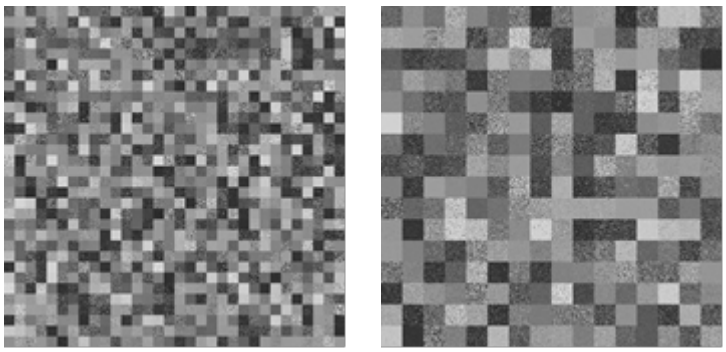
Fig. 11. Lena encrypted image when the shape of the group is square with [29] (a) block size=16×16, (c) block size= 32×32, (b)group size=256, (d) group size=1024



(a) (b)

Fig. 12. Encrypted results using non-square parameters on Lena (a) $N=257$, (b) $N=1025$

In addition, proposed in the actual implementation process. The proposed is not only for the exchange order between groups. Moreover, we will disrupt the pixel in the group, as shown in **Fig. 13**. From the results, our method can increase the complexity of the encrypted image. [29] must select a pixel as a basic pixel that could not disrupt the pixels, Otherwise the secret message cannot be correctly extracted. These two reasons, the proposed is significantly better than [29] in the encrypted image of the vision complexity.



(a) (b)

Fig. 13. Permutations within group (a) $N = 256$, (b) $N = 1024$

The final part, false alarm rate (FAR) and true positive rate (TPR) test, we use RS Scheme for steganalysis. In RS scheme [31] segment the group with *n* connected pixels. Then uses a discrimination function with a mask *M* to categorize *G* into three disjoint groups-regular, singular, and unusable groups. The ratio of the regular groups R_M , R_{-M} and the ratio of the singular groups S_M , S_{-M} are the calculated. If R_{-M} similar to S_M and R_M similar to S_{-M} , it is determined that the image without embedd message, otherwise the image is embedd message. In our experimental results, we use 1500 test images from [32]. Simultaneously, we set four kinds of situations. In addition, used 100%, 75%, 50%, and 25% of the images in 1500 images as stego images. The image that was selected, as stego will be used fully embed of proposed method. Then we use the RS detection value RS_V of [33] to classify the true positive (TP), false positive (FP), false negative (FN), and true negative (TN) scenarios. When RS_V is greater than the threshold, it is determined as stego image otherwise cover image, where the threshold value is 0.05. The experimental result in **Table 2** and the results show that the determin rate is very low. Moreover, FAR and TPR is very low, indicates that our method can resist the steganalysis attack.

Table 2. FAR and TPR result

100%

TP= 150	FP= 0	150
FN=1500	TN= 0	1,500
1,500	0	3,000

FAR=NaN

TPR=0.100

75%

TP= 118	FP= 57	175
FN=1007	TN=318	1,325
1,125	375	3,000

FAR=0.152

TPR=0.105

50%

TP= 76	FP=122	198
FN= 674	TN=628	1,302
750	750	3,000

FAR=0.163

TPR=0.101

25%

TP= 42	FP=179	221
FN=333	TN=946	1,279
375	1,125	3,000

FAR=0.159

TPR=0.112

4. Experimental Results

In this section, several experimental tests are performed. Six test images are picked from USC-SIPI Image Database [27] (see **Fig. 14**.). In our experiments, the results are considered location map. Simultaneously, the payload in all of our experiments has been deducted from any side information that is meant pure payload. We embed six test images in a fully embedded method as shown in **Table 2**. In the "*LM* size" column, where * represents without over/under flow in the array of records . Therefore, this is the result of compressing the empty array. The results show that the number of location maps we generate is small and that there are almost without over/under flow on many smooth images.

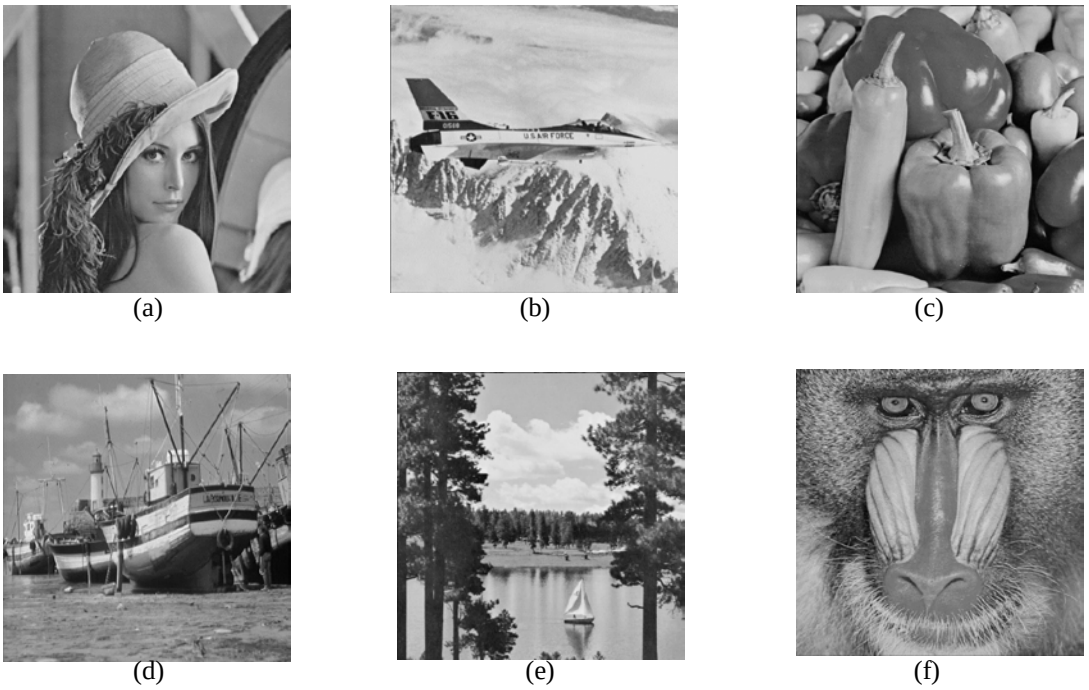


Fig. 14. Six 8-bit test grayscale images of size 512×512 . (a) Lena, (b) Jet, (c) Peppers, (d) Boat, (e) Sailboat and (f) Baboon

Two test images Lena and Baboon are used to test the effect of threshold *TH* on image quality. **Fig. 15** shows the result of this test. Alg. (I) shows the result with threshold *TH* and Alg. (II) is the result without using the threshold. From the results, we conclude that the threshold can effectively increase image quality in Lena and Baboon. Therefore, in this dissertation, threshold *TH* is used to increase image quality.

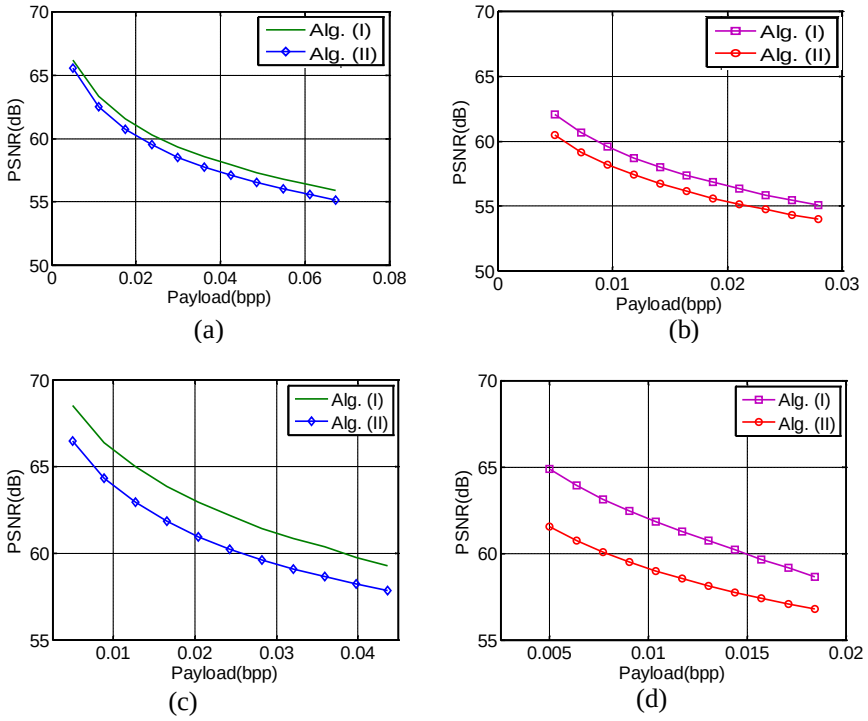


Fig. 15. Compare the results of using the threshold. (a) Lena *N*=4 (b) Baboon *N*=4. (c) Lena *N*=8 and (d) Baboon *N*=8.

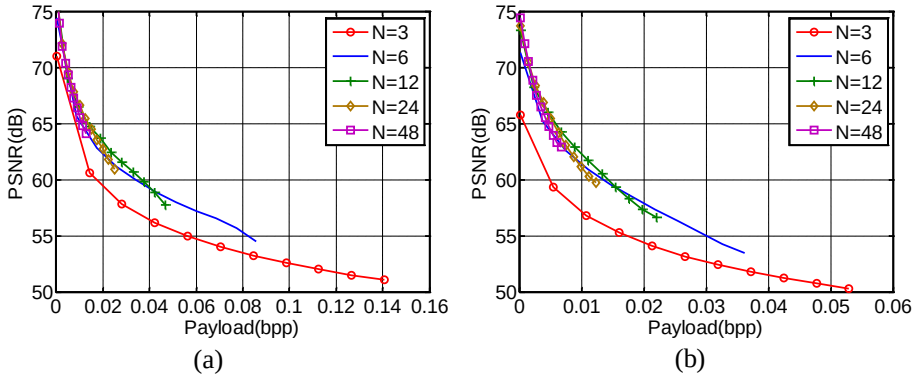


Fig. 16. Comparison of various group size results (a) Lena (b) Baboon.

Next, experimental tests are show for different group sizes. **Fig. 16** shows the images Lena and Baboon are separately divided into group sizes of *N*={3,6,12,24,48}. From the experimental results, the bigger *N* can be obtained better image quality, but the payload is low. However, the smaller *N* provides that image quality is poorer but the payload is higher.

The next experiments point out different payloads by using dynamic group size *N* and threshold *TH*. The method is conceptually simple. Firstly, we estimate whether the payload is enough to embed the message. If the payload is not enough, then gradually increase *TH*. If the payload is still not enough for embedding, then gradually decreased *N* until the message can be completely embedded. *TH* can be a gradually increase in this range 0~255. *N* will be gradually

decreased from 64 to 3. For different group size, the threshold will gradually increase from 0. **Fig. 17** shows the experimental tests from 6 test images based on this procedure. As a result, the complexity image shows poorer performance while the smoother images showed larger payload and higher PSNR.

Table 2. PSNR, payload and location map results

	Group size N=3		
Image	PSNR(dB)	Payload(bits)	*LM* size(bits)
Lena	51.07	36,891	288*
Jet	51.61	47,108	288*
Peppers	50.81	31,542	288*
Boat	50.65	26,017	328
Sailboat	50.66	26,560	288*
Baboon	50.27	13,886	360
Avg.	50.85	30,334	307

	Group size N=4		
Image	PSNR(dB)	Payload(bits)	*LM* size(bits)
Lena	52.62	31,807	272*
Jet	53.22	38,647	272*
Peppers	52.29	27,844	272*
Boat	52.11	23,857	312
Sailboat	52.09	23,270	272*
Baboon	51.62	12,813	384
Avg.	51.33	26,373	348

	Group size N=5		
Image	PSNR(dB)	Payload(bits)	*LM* size(bits)
Lena	53.60	25,778	264*
Jet	54.20	30,225	264*
Peppers	53.32	23,096	264*
Boat	53.11	19,442	296
Sailboat	53.11	19,129	264*
Baboon	52.60	10,607	336
Avg.	53.32	21,380	316

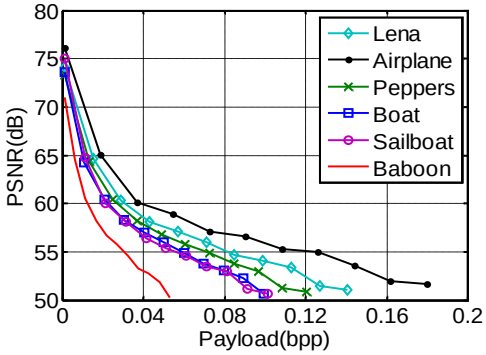


Fig. 17. Performance comparison of various test images for the proposed method

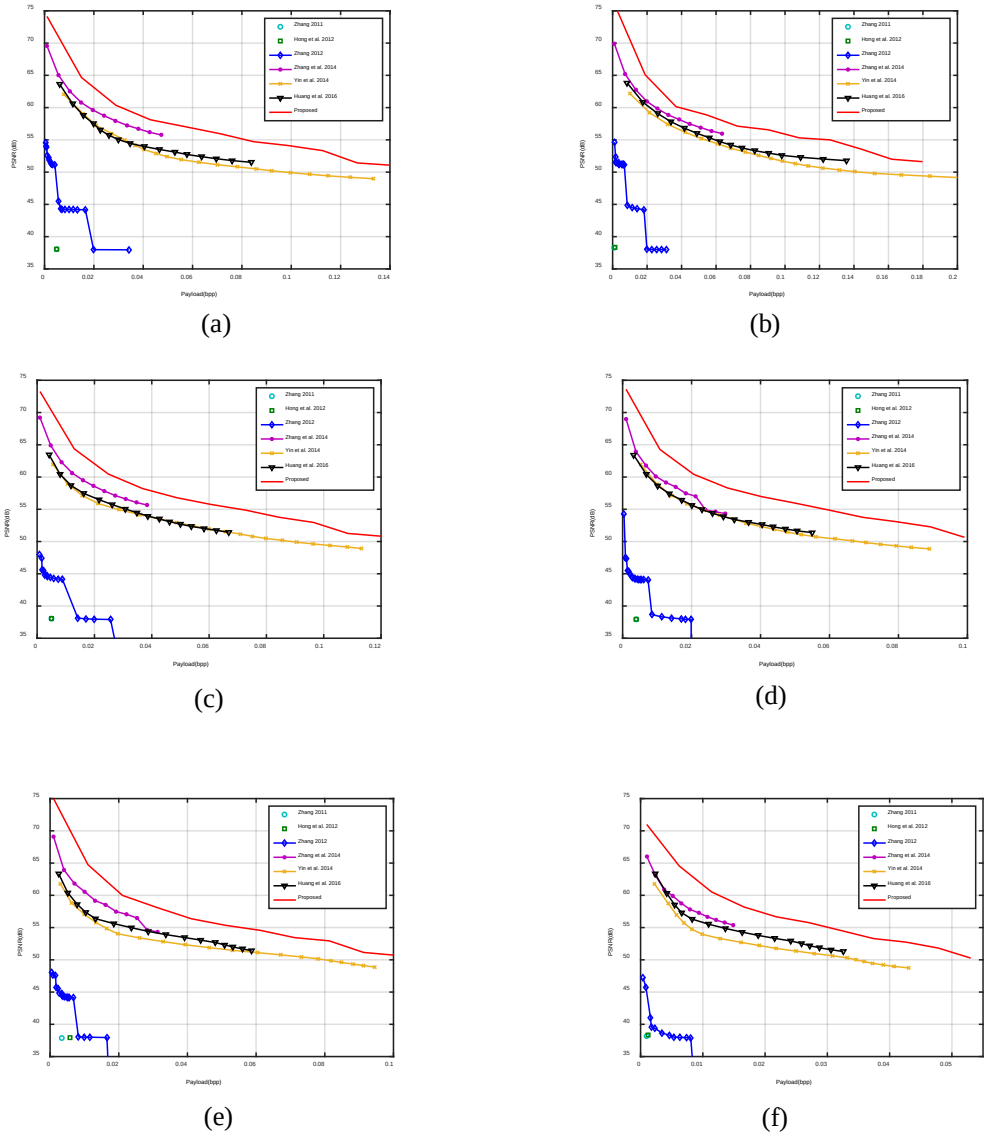


Fig. 18. PSNR comparison of various test images. (a) Lena, (b) Jet, (c) Peppers, (d) Boat, (e) Sailboat and (f) Baboon

Finally, comparisons with other existing RDHEI methods; namely, different methods with cover image and decrypted image containing embedded message by showing the relationship between payload and PSNR as in **Fig. 18**. [15] - [17] is based on XOR encryption method, they require a large number of pixel sets for restore image after embedding. Therefore, their capacity is usually not high, while the image of larger destruction in embedded process. [21] approach is similar to traditional RDH of histogram shifting, but they need to provide a lot of space to record location map, thus capacity is also limited. In the experiment, [21] payload cannot achieve 0.05bpp. [28] - [29] Although the use of traditional RDH, but their encryption algorithms make their embedding inefficiencies. Therefore, our capacity and image quality has a better performance. However, the proposed method is sensitive to the complexity of the original image. Consequently, the complexity of the image on the capacity will be limited, for example baboon capacity about 0.05bpp. Relatively smooth images such as Lean or F-16, the capacity can exceed 0.14bpp. According to the results, we claim that regardless of any images, the proposed method leads to higher PSNR than the other methods. Also, the proposed method leads to larger payload than the others.

5. Conclusion

In this paper, we introduce a new RDHEI method based on random permutation to encrypt the cover image. This unified encryption strategy allows that most of histogram-based reversible data hiding to embed messages in the encrypted image. From the experimental results, the proposed method provides better results with respect to image quality and payload than other existing RDHEI. Finally, we claim that our method achieves strong content-owner privacy by decoupling the decryption and extraction abilities of the receiver.

References

[1] S.A. Parah, F. Ahad, J.A. Sheikh, G.M. Bhat, "Hiding clinical information in medical images: A new high capacity and reversible data hiding technique," *Journal of Biomedical Informatics*, 66, pp. 214-230, 2017. Article (CrossRef Link)

[2] J. Chen, T.S. Chen, W. Hong, G, Horng, H.Y. Wu, and C.W. Shiu, "A new reference pixel prediction for reversible data hiding with reduced location map," *KSII Transactions on Internet and Information Systems*, 8, pp. 95-98, 2014. Article (CrossRef Link)

[3] J. Wang, J. Ni, and X. Zhang, and Y.Q. Shi "Rate and distortion optimization for reversible data hiding using multiple histogram shifting," *IEEE Transactions on Cybernetics*, 47, pp. 315-326, 2017. Article (CrossRef Link)

[4] L.C. Huang, L.Y. Tseng, and M.S. Hwang, "A reversible data hiding method by histogram shifting in high quality medical images," *Journal of Systems and Software*, 86, pp.716-727, 2013. Article (CrossRef Link)

[5] D.C. Lou, C.L. Chou, H.Y. Wei, and H.F. Huang, "Active steganalysis for interpolation-error based reversible data hiding," *Pattern Recognition Letters*, 34, pp. 1032-1036, 2013. Article (CrossRef Link)

[6] Y.Q. Shi, X. Li, X. Zhang, H.T. Wu and B. Ma, "Reversible data hiding: advances in the past two decades," *IEEE Access: Latest Advances and Emerging Applications of Data Hiding*, 4, pp. 3210-3237, 2016. Article (CrossRef Link)

[7] W. Hong, G. Horng, C.W. Shiu, T.S. Chen, and Y.C. Chen, "Reversible steganographic method using complexity control and human visual system," *The Computer Journal*, 58, pp. 2583-2594, 2015. Article (CrossRef Link)

[8] S. Lian, Z. Liu, Z. Ren, and H. Wang, "Commutative encryption and watermarking in video compression," *IEEE Transactions on Circuits and Systems for Video Technology*, 17, pp. 775-778, 2007. Article (CrossRef Link)
[9] M. Cancellaro, F. Battisti, M. Carli, G. Boato, F.G.B. De Natale, and A. Neri, "A commutative digital image watermarking and encryption method in the tree structured haar transform domain," *Signal Processing: Image Communication*, 26, no. 1, pp. 1-12, 2011. Article (CrossRef Link)
[10] 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*, 19(2009), pp. 906-910, 2009. Article (CrossRef Link)
[11] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Transactions on Circuits and Systems for Video Technology*, 13, pp. 890–896, 2003. Article (CrossRef Link)
[12] W. Hong and T.S. Chen, "Reversible data embedding for high quality images using interpolation and reference pixel distribution mechanism," *Journal of Visual Communication and Image Representation*, 22, pp. 131-140, 2011. Article (CrossRef Link)
[13] Z. Ni, Y.Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology*, 16, pp. 354–362, 2006. Article (CrossRef Link)
[14] X. Li, J. Li, B. Li, and B. Yang, "High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion," *Signal Processing*, 93, pp. 198-205, 2013. Article (CrossRef Link)
[15] X. Zhang, "Reversible Data Hiding in Encrypted Image," *IEEE Signal Processing Letters*, 18, pp. 255-258, 2011. Article (CrossRef Link)
[16] W. Hong, T.S. Chen, and H.Y. Wu, "An improved reversible data hiding in encrypted images using side match," *IEEE Signal Processing Letters*, 19, no. 4, pp. 199-202, 2012. Article (CrossRef Link)
[17] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Transactions on Information Forensics and Security*, 7, pp. 1556-6013, 2012. Article (CrossRef Link)
[18] Y.C. Chen, C.W. Shiu, and G. Horng, "Encrypted signal-based reversible data hiding with public key cryptosystem," *Journal of Visual Communication and Image Representation*, 25, pp. 1164-1170, 2014. Article (CrossRef Link)
[19] X. Zhang, J. Long, Z. Wang, and H. Cheng, "Lossless and reversible data hiding in encrypted images with public key cryptography," *IEEE Transactions on Circuits and Systems for Video Technology*, 26, pp. 1622-1631, 2015. Article (CrossRef Link)
[20] C.W. Shiu, Y.C. Chen, and W. Hong, "Encrypted image-based reversible data hiding with public key cryptography from difference expansion," *Signal Processing: Image Communication*, vol. 39, pp. 226-233, 2015. Article (CrossRef Link)
[21] Z. Qian, X. Zhang, and S. Wang, "Reversible data hiding in encrypted JPEG bitstream," *IEEE Transactions on Multimedia*, 16, pp. 1486-1491, 2014. Article (CrossRef Link)
[22] Z. Qian, H. Zhou, X. Zhang, and W. Zhang, "Separable reversible data hiding in encrypted JPEG bitstreams," *IEEE Transactions on Dependable and Secure Computing*, 2016. Article (CrossRef Link)
[23] Z. Qian, and X. Zhang, "Reversible data hiding in encrypted image with distributed source encoding," *IEEE Transactions on Circuits and Systems for Video Technology*, 26, pp. 636-646, 2016. Article (CrossRef Link)
[24] Z. Qian, X. Zhang, and G. Feng, "Reversible data hiding in encrypted images based on progressive recovery," *IEEE Signal Processing Letters*, 23, pp. 1672-1676, 2016. Article (CrossRef Link)
[25] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted Images," *Signal Processing*, 94, pp. 118-127, 2014. Article (CrossRef Link)
[26] P. Howard, F. Kossentini, B. Martins, S. Forchhammer, and W. Rucklidge, "The emerging JBIG2 standard," *IEEE Transactions on Circuits and Systems for Video Technology*, 8, pp. 838-848, 1998. Article (CrossRef Link)
[27] The USC SIPI Image database. Available: http://sipi.use.edu/database/

[28] F. Huang, J. Huang, and Y. Q. Shi, "New framework for reversible data hiding in encrypted domain," *IEEE Trans. Inf. Forensic Security*, 11, pp. 2777-2789, 2016. Article (CrossRef Link)
[29] Z. Yin, B. Luo and W. Hong, "Separable and error-free reversible data hiding in encrypted image with high payload," *The Scientific World Journal*, pp. 1-8, 2014. Article (CrossRef Link)
[30] P. Tsai, Y.C. Hu and H.L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," Signal Processing, 89, pp. 1129-1143, 2009. Article (CrossRef Link)
[31] J. Fridrich, M. Goljan, R. Du, "Reliable detection of LSB steganography in color and grayscale images," pp. 27-30, 2001. Article (CrossRef Link)
[32] The Kodak image database, Available: http://r0k.us/graphics/kodak/
[33] J.C. Joo, H.Y. Lee, and H.K. Lee, "Improved steganographic method preserving pixel-value differencing histogram with modulus function," *EURASIP Journal on Advances in Signal Processing 2010*, 2010. Article (CrossRef Link)

Chih-Wei Shiu received his Ph.D. degrees from Department of Computer Science and Engineering, National Chung Hsing University, Taiwan, in 2014. Currently, he is an assistant professor at Department of Education Industry and Digital Media, National Taitung University, Taitung, Taiwan. His current research interests include steganography, digital game design and visual reality.

Yu-Chi Chen received the B.S., M.S., and Ph.D. degrees from Department of Computer Science and Engineering, National Chung-Hsing University, Taiwan, in 2008, 2009, and 2014 respectively. In 2013, he was a visiting scholar at Department of Electrical Engineering, University of Washington. He was a postdoctoral fellow at the Institute of Information Science, Academia Sinica, Taiwan from 2014 to 2017. He is currently an assistant professor at Department of Computer Science and Engineering, Yaun Ze University, Taiwan. His research interests include cryptography, information security, and blockchain techniques.

Wien Hong received his M.S. and Ph.D. degree from the State University of New York at Buffalo, USA in 1994 and 1997, respectively. He is currently a researcher at Nanjing University of Information Science & Technology and Nanfang College of Sun Yat-Sen University. His research interests include steganography, watermarking and image compression.