2017 2nd International Conference on Multimedia and Image Processing

A High-capacity Reversible Data Hiding Method in Encrypted Images Based on Block Shifting

Ruiqi Jiang, Weiming Zhang, Hui Wang, Nenghai Yu
CAS Key Laboratory of Electro-magnetic Space Information
University of Science and Technology of China
Hefei, 232026, China
e-mail: jrq123@mail.ustc.edu.cn; zhangwm@ustc.edu.cn; whlory@mail.ustc.edu.cn; ynh@ustc.edu.cn

***Abstract*—Reversible data hiding in encrypted images (RDH-EI) is a new topic drawing more and more attention because of the privacy-preserving requirements from cloud data management. In this paper, a novel reversible data hiding scheme for encrypted image with large embedding capacity is proposed. And it is a client-free RDH-EI scheme in which the RDH algorithm used by the cloud is irrelevant with both the sender and receiver. Different from the previous methods, we propose a block shifting method to transform the original image into the scrambled image which not only looks throughly meaningless, but also obtains a sharp histogram by keeping block correlation meanwhile. For the scrambled image, it is easy for the data hider to reversibly embed data with any plaintext image RDH algorithm such as histogram shifting or lossless compression. The proposed method can achieve real reversibility, that is, data extraction and image recovery are executed with no error. Compared to the previous methods, the experimental results demonstrate that our proposed method increases the embedding capacity in a big range without loss of perfect secrecy.**

Keywords-reversible data hiding; image encryption; block shifting; privacy protection; outsoured storage in cloud

I. INTRODUCTION

Nowadays with the booming of cloud computation technique in the world, outsourced storage by cloud becomes a more and more popular service, especially for multimedia files which need large storage space. To manage the outsourced images, the cloud server may embed some additional data into the images, such as image category and notation information, and use such data to identify the ownership or verify the integrity of images. Obviously, the cloud service provider has no right to introduce permanent distortion during data hiding into the outsourced images. Therefore, reversible data hiding (RDH) technology is needed, by which the original image can be losslessly recovered after the embedded message is extracted. With the property of reversibility, RDH can also be used into many sensitive applications, such as medical imagery, military imagery and law forensics, where any distortion of the original cover is not allowed.

Many RDH techniques on plaintext images have been reported in past years. In theoretical aspect, Kalker and Willems [1] established a rate-distortion model for RDH and proposed a recursive code construction which, however, does not approach the bound. By improving the recursive code construction for binary covers, Zhang et al. [2] [3] proved that the generalized codes can reach the rate-distortion bound as long as the compression algorithm reaches entropy. In practical aspect, Fridrich et al. [4] proposed a compress-and-append based RDH technique which aims to find a compressible region to get redundancy space, into which data can be inserted. Introduced by Tian [5], RDH based on difference expansion (DE) embeds one bit of data by adding it to the expanded pixel difference. Ni's histogram shift (HS) [6] based RDH gets redundant space for data embedding by shifting the bins of gray-level histogram. To increase the embedding capacity (EC) while leading to less distortion in image content, researchers recently apply DE and HS to the prediction errors which have small entropy and can be easily compressed [7], [8]. This prediction-error expansion (PEE) method has been the most powerful RDH technique.

On the other hand, cloud service for outsourced storage makes it challenging to protect the privacy of image contents, such as private image, medical images of patients and military image. Even though RDH is helpful for managing the outsourced images, it can not protect the image content. To provide confidentiality for images, encryption previous to outsourced storage is a widespread and effective manner, whilst it brings much difficulty for the cloud service provider to manage images and ensure data integrity. So it is significant to implement reversible data hiding in encrypted images (RDH-EI), by which the cloud server can reversibly embed data into the images but can not get any knowledge about the image contents.

Inspired by the needs of privacy protection, some attempts on RDH-EI have been presented in recent years, which can be further divided into two frameworks. One is called “vacate room after encryption (VRAE)” [9]-[11], and the other is “reserve room before encryption (RRBE)” [12] depicted in Figure 1 and Figure 2.

In the framework VRAE shown in Figure 1, such as schemes in [10] and [11], the image owner (the sender) encrypts the image *I* into *E(I)* with a key *K*. The cloud server embeds data by compressing the encrypted image *E(I)* and generates *Ew(I)* that is stored in the cloud. When getting a retrieval request, the cloud server returns *E'w(I)* to the receiver, maybe an authorized third party, who generates *I* through a process of joint decompression and decryption with the key *K*. Herein, *E'w(I)* may be just *Ew(I)* or a modified version obtained by removing the embedded data. Note that the cloud server cannot restore *E(I)* from *Ew(I)*, since decompression should be joined with decryption with the help of *K*. In this framework, the complexity is taken on by the receiver who must join the process of decompression and decryption to get the original image. In other words, the compression-based RDH method used by the cloud server should be specified together with the receiver, i.e., the RDH method is receiver-related. Besides, as for VRAE, since the entropy of ciphertext image has been maximized, the methods belonging to this framework can only achieve small payload with the limited error rate at data extraction and cover restoration.

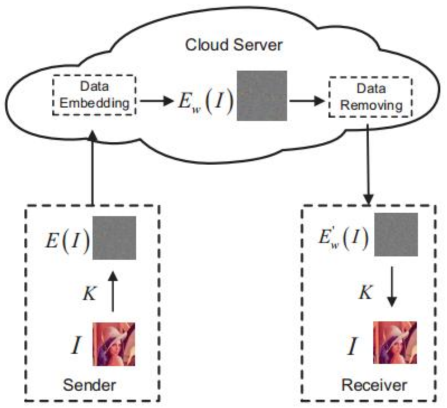


Figure 1. Framework of VRAE in RDH-EI

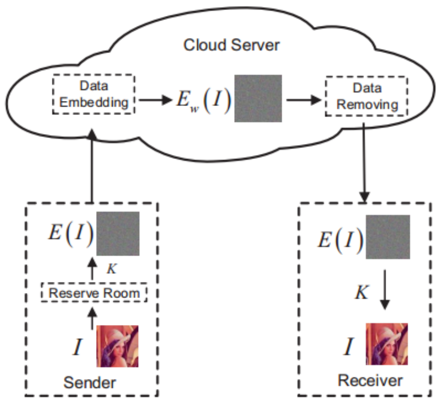


Figure2. Framework of RRBE in RDH-EI

In the framework RRBE shown in Figure 2, such as schemes in [12]-[14], the image owner (the sender) reserves room from the image *I* and encrypts it into *E(I)* with a key *K*, and then sends it to the cloud server who embeds data into the reserved room and generates *Ew(I)*. *Ew(I)* is stored in the cloud, from which the cloud server can extract the data that is used for management. When an authorized user (the receiver) wants to retrieve the image, the cloud server can restore *E(I)* from *Ew(I)* and send *E(I)* to the user who can decrypt *E(I)* and get *I* with the key *K*. In the framework RRBE, the complexity is borne by the sender who should reserve room for RDH by exploiting the redundancy within the image and thus the RDH method used by the cloud should be specified with the sender, that is, the RDH method used by cloud is sender-related.

In this paper, we propose a novel algorithm for RDH-EI based on the framework of VRAE. This method can outcome the shortcomings of receiver or sender related and can achieve the real reversible goal. Different from the previous methods which encrypt bits through exclusive-or operation, a novel scramble-based encryption method is proposed to generate a more sharply distributed pixel histogram which is significant to improve RDH-EI algorithm performance. The main contributions of this novel algorithm include:

- In the novel algorithm for RDH-EI, the cloud server can easily embed data into the “encrypted image” by arbitrarily selecting any RDH method for plaintext images such as those in [4]-[6]. In other words, the RDH used by the cloud is irrelevant with both the sender and receiver, which is called a client-free RDH-EI scheme by us. “Client free” is important for the scenarios of public clouds, in which it is hard for the cloud server to ask the clients how to encrypt or decrypt their data, because the cloud is thought to be only semi-honest [15].
- Within the outsourced storage service, meaningless images after encryption are easy to cause the attention of lawbreakers who may try to dig out private information. For conveniently tracing, temper proofing and integrity authentication, cloud server usually needs to embed a large amount of information which records operations and other behaviors from malicious attackers into the meaningless images in a reversible way. Therefore it needs a high-capacity reversible data hiding technique for the encrypted images. The algorithm in this paper focuses on obtaining a large embedding capacity (EC) within the meaningless images in order to meet this demand.

The rest of this paper is organized as follows. Section II presents the details of the proposed RDH algorithm based on scrambled ciphertext image. The experimental results with analysis and comparison are given in Section III. Finally a conclusion is drawn in Section IV.

II. PROPOSED SCHEME

A. *Generation of Scrambled Image*

A. Block Shift Lee et al. [16] proposed a new secure image transmission technique which transforms automatically a secret image into a so-called secret-fragment-visible mosaic image of the same size according to their color similarity. However, their method can not recover secret image perfectly, which as a result has a blocking effect to the image content. Based on the standpoint of reversibility and large embedding capacity, referring to the “transformation” idea, we propose a novel method called “block shifting” to transform a secret image to another form in this paper. The aim of transformation is to generate a more sharply distributed pixel histogram. As well known, the more sharply distributed the pixel histogram of an image is, the higher the embedding capacity will be. The algorithm to be presented can be used for both gray-level images and color images. Take the grey-level images for example.

Given an 8-bit gray-level image *I*, let p_i be the pixel value of on pixel in the image, ranged from 0 to 255. First divide the image into several square blocks of $a \times a$ size. We compute the mean of every block respectively by the following formula:

$$\mu_b = round\left(\frac{1}{n}\sum_{i=1}^{n} p_i\right) \tag{1}$$

Where *n i*s the number of pixel in one block, that is, $n = a \times a$. Next we generate the histogram of all means of blocks and obtain the mode mean value represented as $\mu_{\text{mod}e}$. Then we compute a new value $p_i^{'}$ for each p_i according to equation (2).

$$p_i^{'} = p_i + \mu_{\text{mod}e} - \mu_b \tag{2}$$

Note that after this transformation of block shift, the mean of every block will become $\mu_{\text{mod}e}$. According to the following formulas, the standard deviation of each block before and after shifting will keep the same, that is, $\sigma_i = \sigma_i^{'}$.

$$\begin{aligned} \sigma_i &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - \mu_b)^2} \\ \sigma_i^{'} &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p'_i - \mu_{\text{mod}e})^2} \end{aligned} \tag{3}$$

To achieve the reversible goal we only need to record $\Delta\mu$ and embed it into the image by a reversible data hiding method, which is calculated from equation (4).

$$\Delta\mu = u_{\text{mod}e} - u_b \tag{4}$$

With block shift as (2) it may lead to the overflow/underflow problem of some pixels in one block. Thus $\Delta\mu$ should be a little modified to solve the overflow/underflow problem absolutely as far as reversibility is concerned. For each block, we represented the maximal overflow pixel as $OV_{\max}$ and the minimal underflow pixel $UV_{\min}$. The operator is as follows:

$$\Delta u = \begin{cases} \Delta u = (OV_{\max} - 255) \\ \Delta u = (UN_{\min} - 0) \end{cases} \tag{5}$$

By this way all the pixels values are controlled into the range of [0,255].

B. Embed parameters with RDH Once the original secret image has been transformed by block shifting method, we should embed the $\Delta\mu$ as a location map into the image for real reversibility. Due to the reason that the transformation of block shifting still keeps the image local correlation, any RDH technique of plain-image can be implemented in the transformed image. For simplicity we just adopt RDH in [7] using sorting and prediction. Previous to embed $\Delta\mu$ we apply Huffman coding to further compress $\Delta\mu$ to obtain a smaller amount of parameter bits. Suppose that the original image is "Lena" shown in Figure 3(a) and its histogram is displayed in Figure 4(a). After block shifting, the resultant image called "mosaic image" is shown in Figure 3(b) and its histogram is displayed in Figure 4(b). According to the given result it proves that the block shifting method generates a sharply distributed histogram.

C. Block Permutation It seems that the obtained mosaic image still remains edge information of original image. To further camouflage the image content we scramble the image next. We adopt a block permutation method based on MD5 algorithm [17], a hash function to generate pseudo-random numbers. It is one of one-way hash functions, which cannot obtain *x* given hash function value *H(x)*. The procedure of MD5 is described briefly as follows. After appending padded bits to the message and initializing MD buffer, the MD5 algorithm loop performs a compression function to output a 128-bit hash code. Suppose that the number of blocks in a whole image is *A*, $i \in (1, \text{A})$, the number of blocks in a row is *X* and the number of blocks in a column is *Y*.

For each block based on a raster scan order, we can get a pseudo-random output and a unique number $j \in (1, \text{A})$ by computing following formulas.

$$\begin{aligned} \nu &= (\nu + MD5(\mu, key1)) \bmod Y \\ \mu &= (\mu + MD5(\nu, key2)) \bmod X \\ \nu &= (\nu + MD5(\nu, key3)) \bmod Y \\ j &= \nu X + \mu \end{aligned} \tag{6}$$

In the formulas of (6), *key1, key2* and *key3* are the three secret keys. To enhance the safety of the permutation algorithm, every key is made up of 128-bit data. So it is impossible to find the correct key by using an exhaustive search method. Since it is probably that each modular operation generates a repeated number, we combine all the results of modular operation to finally get a unique number *j*.

After block permutation, a scramble image is created as Figure 4(c) shows. It looks thoroughly meaningless since the execution of block permutation itself is a method of encryption. As only the location of every block is moved to another, the histogram of the scrambled image remains the same as that of the mosaic image.

B. Data Embedding in Scrambled Image

Now the scrambled image is sent to the cloud for storage. Because of local correlation it can be inferred that the standard derivation of each block is small. Moreover, after block shift operation, the mean of every image block turns out to be almost the same. As a consequence the shape of pixel histogram will appear single-peak-like distributed.

Because of the sharp histogram of the scrambled image, two RDH techniques can be used in the scrambled image with good performance. One is to adopt HS based RDH to embed additional data, the other is to adopt compress-and-appended RDH technique. Both of them belong to plaintext

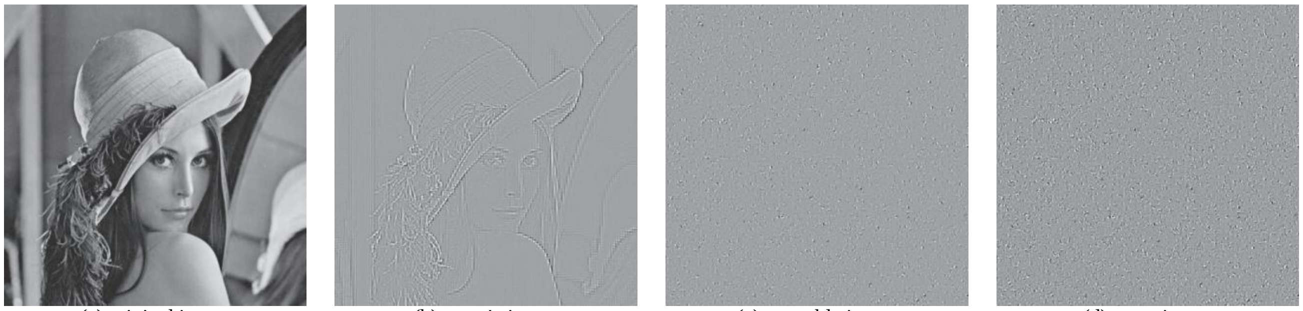
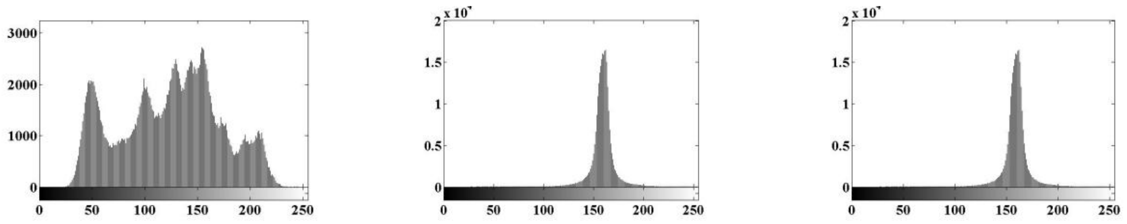

(a) original image (b) mosaic image (c) scramble image (d) stego image

Figure 3. The resultant of image after block shifting and block permutation and data embedding



(a) histogram of original image (b) histogram of mosaic image (c) histogram of scramble image

Figure 4. The histogram of original image, mosaic image and scramble image

images RDH technique. Let us firstly introduce the procedure of HS based RDH method.

For an $M \times N$ image with pixel value $x \in [0, 255]$, data embedding proceeds as follows:

1) Generate its histogram $H(x)$.
2) In the histogram find K pairs of maximum and minimum points, where K is determined from the embedding capacity. Divide the histogram into two parts of bins, that is, left part and right part, based on the peak point as a separator bin. Without loss of generality, in order to decrease the shift length of bins we group maximum and minimum point both in the same part into a pair. The minimum points should be chosen to satisfy as follows to prevent overflow /underflow problem.

$$P_{LMIN} > \frac{K}{2} \text{ OR } P_{RMIN} < 255 - \frac{K}{2} \qquad (7)$$

where P_{LMIN} represent the left-most minimum point and P_{RMIN} the right-most minimum point.

3) Suppose a and b is a pair of maximum and minimum point. Without loss of generality, it meets the condition $a<b$. For each pair, if $h(b)>0$ recode the coordinate of those pixels and the pixel grayscale value as overhead information. Then set $h(b)=0$.
4) Move the whole part of the histogram in the range of (a, b) to the right by 1 bin so as to vacate a bin $a+1$.
5) Scan the image in a raster sequence. Once the pixel's grayscale value is a, check the to-be-embedded bit. If the bit is "1", the pixel's grayscale value is added by 1 to turn into $a+1$. Otherwise if the bit is "0", the pixel value remains unchanged.
6) Repeat step 3-5 from $k=1$ to $k=K$ until all the data is embedded into the image.

Since the shape of scrambled histogram is close to Laplacian distribution, the histogram of the stego image still remains sharp. That is to say, we can still embed additional data into the marked scrambled image after the first layer of data embedding. The stego image generated by embedding data into the scrambled image is shown in Fig.1 (d), which looks more meaningless than the scramble-based encrypted image. But it should be mentioned here, with the increasing of layers, more overhead information should be recorded, which will decrease the effective payload in turn.

As for the compress-and-appended RDH technique, due to the reason of sharp distribution histogram, it is simply for the cloud administrator to implement lossless entropy coding to compress the scramble image. In the present paper, we simply adopt Huffman compress coding to obtain redundant space for data embedding.

C. Data Exaction and Image recovery

For the compress-and-appended RDH technique, the cloud administrator just extract message from the redundant space and decompress bit stream to get the message data and restore the scrambled image.

On the other hand, for the HS-based RDH technique since multiple histograms are shifted to embed data in a sequence order, decoding should be proceeded in a reverse order to guarantee the accuracy of extracted data and scrambled image recovery. In concrete terms, it means that decoding is conducted from $k=K$ to $k=1$. For the sake of brevity, let us describe the case of $k=i$, where $k \in (1,K)$. The data extraction of other cases is the same as following steps.

Assume that the grayscale value of the maximum point and the minimum point are a and b, respectively. For the left part it is $a>b$ and $a<b$ for the right part. The stego image is of size $M \times X$, each pixel grayscale value $x \in [0,255]$. Then do the following steps.

1) Scan each pixel in a raster sequential order as that in the embedding procedure. If a pixel's grayscale value *a+1* is encountered, a message bit "1" is extracted. If the pixel value is *a*, a message bit "0" is extracted.
2) After extracting all the data bit, again scan the image in the same order as 1). Once a pixel value falls into the range of *(a,b]*, the pixel value x is subtracted by one.
3) If there exists overhead information in the extracted data, one should set the pixels' values of those locations to be *b*.

After all the embedded data is extracted and the scrambled image is obtained, the cloud administrator can analyze what has been done with this image according to the extracted data. If the owner wants to get the original plain-text image, it needs to execute the following procedures.

1) Since the owner has the three scramble keys *key1*, *key2* and *key3*, the mosaic image can be obtained from the scrambled image in an accurate way which will be the same as the mosaic image after block shifting and still remain edge information of original image.
2) Extract compressed $\Delta\mu$ bit stream as the process described in [7] and decompress it to acquire the parameter $\Delta\mu$.
3) For each block of the mosaic image, shift it back with current $\Delta\mu$ to get the original block by the following equation.

$$p_i = p_i^{'} - \Delta\mu \tag{8}$$

4) Repeat the shift procedure in a raster-scan order until the last block, and finally the original image will recovered perfectly.

III. EXPERIMENTAL RESULTS

Due to the reason that our method is a reversible way to embed data into meaningless image, we test the embedding capacity in this experiment. We utilize 8 USC-SIPI grayscale images with the size of 512 × 512 [18] as test images in this experiment. As mentioned above, our method in this paper emphasizes on the creation of sharp distribution of image histogram. The sharper it is, the higher capacity and larger compression rate will be achieved. Thus we firstly compare the image histogram between original image and scrambled image.

In Figure 5, the histogram of four test mosaic images named "Baboon", "Airplane", "Barbara" and "Peppers" are displayed. It can be concluded that the histogram of scrambled image for this four test images are close to the distribution of Laplacian, as expected. More peak points and more zero points can be gathered together after the block shifting operation, which is suitable for performing HS-based RDH method. In additional, it is also much more efficient to adopt Huffman entropy coding to compress the image and earn extra space to embed data. We further select ten pairs of maximum and minimum points both in original image and scrambled image for comparison. Even though the scrambled images can have more than 10 suitable pairs of extremum value to select, we just set the same parameter for comparison. Since the pure payload of HS-based RDH depends on the number of maximum points when there exists zero minimum points, our method will gain a larger embedding capacity. It is verified by the result for different test images in Table I.

TABLE I THE NUMBER OF MAXIMUM AND MINIMUM PIXELS FOR ORIGINAL AND SCRAMBLED IMAGE

Images	**Original Images**		**Scrambled Images**	
	Max total	***Min total***	***Max***	***Min total***
Barbara	20160	0	99508	0
Boat	48550	0	141764	0
Airplane	7003	0	170336	0
Lena	25371	0	147721	0
Peppers	27194	0	127730	0
Baboon	26514	0	50215	17

Next we give the embedding capacity of the compress-and-appended RDH method for different test images in Table II. As for texture images such as "Baboon" and "Barbara", the payload are smaller than that containing a large smooth area in the image content. One reasonable explanation is that the range of pixel is larger than that of smoother images.

Meanwhile we have compared the proposed method in [9]-[12]. As for the reversibility, methods in [9]-[11] which belong to the framework of "VRAE" maybe introduce some errors on data extraction and image restoration, while our proposed method and [12] are free of any error for all kinds of images. What's more, large payload is the other light spot in this paper. Since the embedding capacity of methods in [9]-[11] is much small, we just give the maximum value and represent them as one class VRAE, shown in Table II. In [12], Ma et al. select two LSB-planes of part A, where messages can be accommodated. Since the LSBs of part A are reversibly embedded into part B with a standard RDH algorithm, at most half size of the whole image will become part A. Therefore the maximum embedding capacity is 1 bit per pixel. In fact with the increasing of embedding capacity, the size of boundary map in [12] will become larger. Thus it will decrease the pure payload. It can be proved that for all of the test images from the result shown in Table II, our proposed method gains a much larger payload.

TABLE II. ALGORITHM PERFORMANCE COMPARISON BETWEEN VRAE, [9] AND OURS

Images	**VRAE**	**Ma (12)**	**Proposed**
Barbara	0.03	0.63	1.89
Boat	0.03	0.79	2.56
Airplane	0.05	0.92	2.98
Lena	0.03	0.86	2.85
Peppers	0.04	0.81	2.69
Baboon	0.01	0.43	1.38

IV. CONCLUSION

In this paper, a novel method called block shifting is proposed. It transforms the original image into the scrambled image which not only looks throughly meaningless, but also has a sharply

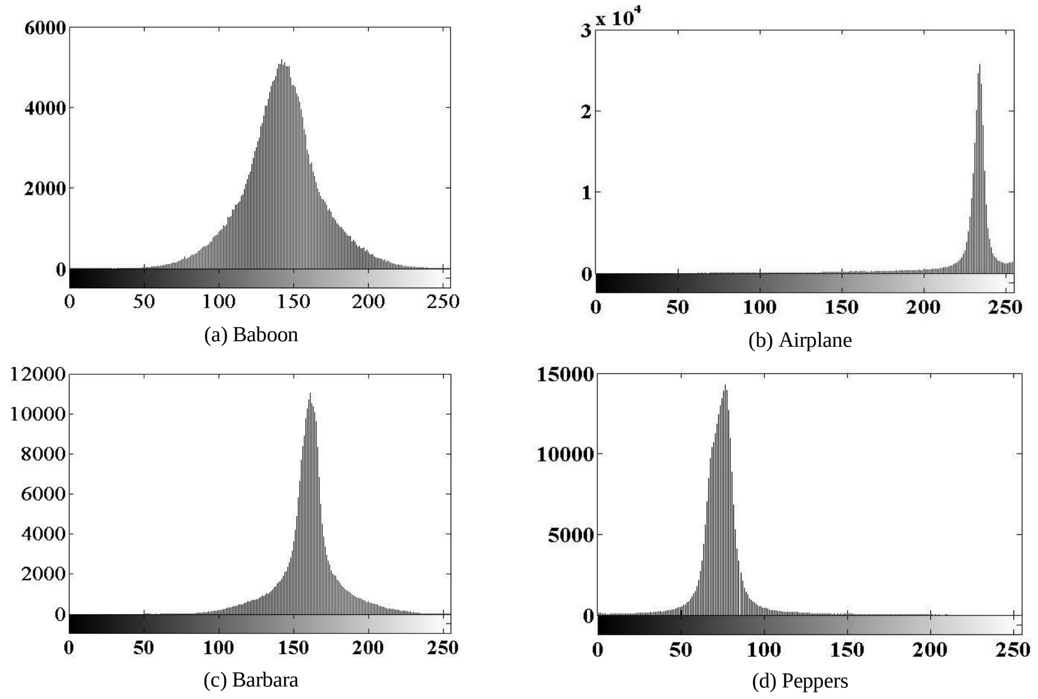


Figure 5. The histograms of different mosaic images

distributed histogram by keeping the block correlation. The data hider can easily embed data into the generated scrambled image with any traditional RDH algorithm for plaintext images. Since the shape of the pixel histogram of the scrambled image is like Laplace distributed, excellent performance without loss of perfect secrecy can be achieved. In addition, the proposed method is client free and can realize real reversibility which means data extraction and image recovery are free of any error. Compared to the previous method, the experimental results demonstrate that our method can increase the embedding capacity in a big range with the same image quality.

REFERENCES

[1] T. Kalker and F.M. Willems, “Capacity bounds and code constructions for reversible data-hiding,” *14th Int. Conf. Digital Signal Processing,* pp. 71-76, 2002.

[2] W. Zhang, B. Chen and N. Yu, “Capacity-approaching codes for reversible data hiding,” in *Proc 13th Information Hiding, LNCS 6958,* pp. 255-269, 2011.

[3] W. Zhang, B. Chen and N. Yu, “Improving various reversible data hiding schemes via optimal codes for binary covers,” *IEEE Trans. Image Process.*, vol. 21, no. 6, pp. 2991-3003, 2012.

[4] J. Fridrich and M. Goljan,“Lossless data embedding for all image for- mats,” *Proc. Photonics West, Electronic Imaging, Security and Water- marking of Multimedia Contents.*, vol. 4675, pp. 572-583, 2002.

[5] J. Tian, “Reversible data embedding using a difference expansion,” in *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890-896, 2003.

[6] Z. Ni, Y. Shi, N. Ansari, and S. Wei, “Reversible data hiding,”. *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354-362, 2006.

[7] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y. Shi, “Reversible watermarking algorithm using sorting and prediction,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 7, pp. 989-999, 2009.

[8] X. Hu, W. Zhang and N. Yu, “Minimum rate prediction and optimized histogram modification for reversible data hiding,” *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 3, pp. 653-664, 2015.

[9] X. Zhang, “Reversible data hiding in encrypted images,” *IEEE Signal Process. Lett.*,, vol. 18, no. 4, pp. 255-258, 2011.

[10] W. Hong, T. Chen, and H. Wu, “An improved reversible data hiding in encrypted images using side match,” *IEEE Signal Process. Lett.*,vol. 19, no. 4, pp. 199-202, 2012.

[11] X. Zhang, Z. Qian, G. Feng, and Y. Ren, “Efficient reversible data hiding in encrypted images,” *Journal of Visual Communication and Image Representation.*, vol. 25, no. 2, pp. 325-328, 2014.

[12] K. Ma, W. Zhang, X. Zhao, N. Yu and F. Li “Reversible data hiding in encrypted images by reserving room before encryption,” in *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 3, pp. 553-562, 2013.

[13] W. Zhang, K. Ma and N. Yu, “Reversibility improved data hiding in encrypted images,” Signal Processing, vol. 94, pp. 118-127, Jan. 2014.

[14] X. Cao, L. Du, X. Wei, et al., “High capacity reversible data hiding in encrypted images by patch-level sparse representation,” IEEE Trans. on Cybernetics, vol. 46, no. 5, pp. 1132-1143, May. 2016.

[15] S. Yu, C. Wang and K. Ren,“Achieving secure, scalable, and finegrained data access control in cloud computing,” IEEE Proceeding of INFOCOM 2010, pp. 1-9, Mar. 2010.

[16] Y. Lee and W. Tsai, “A new secure image transmission technique via secret-fragment-visible mosaic images by nearly reversible color transformations,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 4, pp. 695-703, 2014.

[17] R. Rivest, “The MD5 message-digest algorithm”, “tools.ietf.org”, *MIT Laboratory for Computer Science and RSA Data Security, Inc.*, 1992.

[18] The USC-SIPI Image Database [Online]. Available: http://sipi.usc.edu/database