A Separable Reversible Data Hiding in Encrypted Image with Improved Performance

Rintu Jose,
M.Tech Student (CSE),
Viswajyothi College of Engineering and Technology,
Muvattupuzha, Kerala, India.
rintujose1989@gmail.com

Gincy Abraham,
Assistant Professor (CSE),
Viswajyothi College of Engineering and Technology,
Muvattupuzha, Kerala, India.
abraham.gincy@gmail.com

***Abstract* – This work proposes a novel scheme to reversibly hide data into encrypted grayscale image in a separable manner. During the first phase, the content owner encrypts the image by permuting the pixels using the encryption key. The data hider then hides some data into the encrypted image by histogram modification based data hiding, making use of data hiding key. At the receiver side, if the receiver has only encryption key, he can generate an image similar to the original one, but cannot read the hidden data. Peak Signal to Noise Ratio (PSNR) of this decrypted image is much higher than the existing methods. If the receiver has only data hiding key, he can extract the data, but cannot read the content of the image. If the receiver has both keys, he may first extract the data using data hiding key and then decrypt the image using encryption key. The method also has a higher data hiding capacity than the existing reversible data hiding techniques in encrypted image.**

***Key Terms* – Reversible data hiding, Image encryption, PSNR, Data hiding capacity, Chaotic sequence, Pseudorandom number.**

I. INTRODUCTION

Data hiding is the technique by which some data is hidden into a cover media. The data may be any text related to the image such as authentication data or author information. At the receiver side it must be able to extract the hidden data. In some high-precision applications such as medical, military and remote sensing, it is highly desired that the original image should be perfectly recovered after data extraction. A data hiding technique satisfying this requirement is known as *reversible data hiding*. They are also called *invertible*, *lossless* or *distortion free data hiding*.

Data hiding is usually performed by an inferior assistant or a channel administrator. The owner of the image cannot trust the assistant or channel administrator completely. In such cases, when the owner needs to keep the secrecy of the image, he may first encrypt the image using an encryption key. The channel administrator, without any knowledge about the original image content, has to hide data into the encrypted image using a data hiding key. It is also desired that the receiver can extract the hidden data and recover the original image in a separable manner. Separable means, if the receiver is having the data hiding key only, he can extract the data, but cannot decrypt the image. If he is having encryption key only, it is possible to decrypt the image, but cannot extract the hidden data. If the receiver is having both keys, he can extract the hidden data and recover the original image.

978-1-4673-5149-2/13/$31.00 ©2013 IEEE

Most of the works on data hiding focuses on data hiding and extraction on plain image [1]-[3]. Reversible data hiding by histogram shifting is described in [1]. In [2] data is hidden into the histogram of pixel differences. Data hiding in [3] stores data by making changes to LSB bits.

A number of image encryption techniques have also been developed over years. Encryption algorithms falls under two general categories: substitution and transposition. Some algorithms perform both to enhance security. Substitution based encryption makes changes to the pixel values to make the content unrevealed. A substitution based image encryption is discussed in [4]. Permutation based encryption algorithms are covered in [5] and [6]. In permutation based encryption the pixels are shuffled and no change is made to the pixel values. Image encryption methods combining both substitution and transposition are covered in [7] and [8].

There are a number of schemes which performs data hiding and encryption jointly. In some of them, a part of cover is used to carry additional data and rest of the cover is encrypted. For example, in [9], watermark is added to amplitude of DCT coefficients, and motion vector difference, intra-prediction mode and signs of DCT coefficients are encrypted. A reversible data hiding technique in encrypted image is described in [10], which hides data into completely encrypted image. But in this method image decryption and data extraction are not separable. The method in [11] hides data into an encrypted image in a separable manner.

The proposed method is a separable reversible data hiding in encrypted image with improved performance. The owner of the image first encrypts the image by permutation, making use of an encryption key. Since permutation only shuffles the pixels, the histogram of the image remains the same. The data hider, without any knowledge about the original image content, hides data into the encrypted image by histogram modification method. Before hiding the data, the data hider permutes image using data hiding key and after data hiding he performs inverse permutation. At the receiver side, if the receiver has only data hiding key, he can extract the data, but cannot read the content of the image. If he has only encryption key, he can decrypt the image to get an image similar to the original one. If he has both keys, he may first extract the data using data hiding key and then decrypt the image using encryption key. This decrypted image is exactly same as the original image.

II. PROPOSED SCHEME

We propose an efficient separable reversible data hiding in encrypted image. A sketch of the proposed scheme is given in Fig.1. The owner of the image first encrypts the image using the encryption key. Then, the data hider, without any knowledge about the original image content, hides data using the data hiding key. At the receiver side the receiver extracts the data and decrypts the image in a separable manner, making use of data hiding and encryption keys.

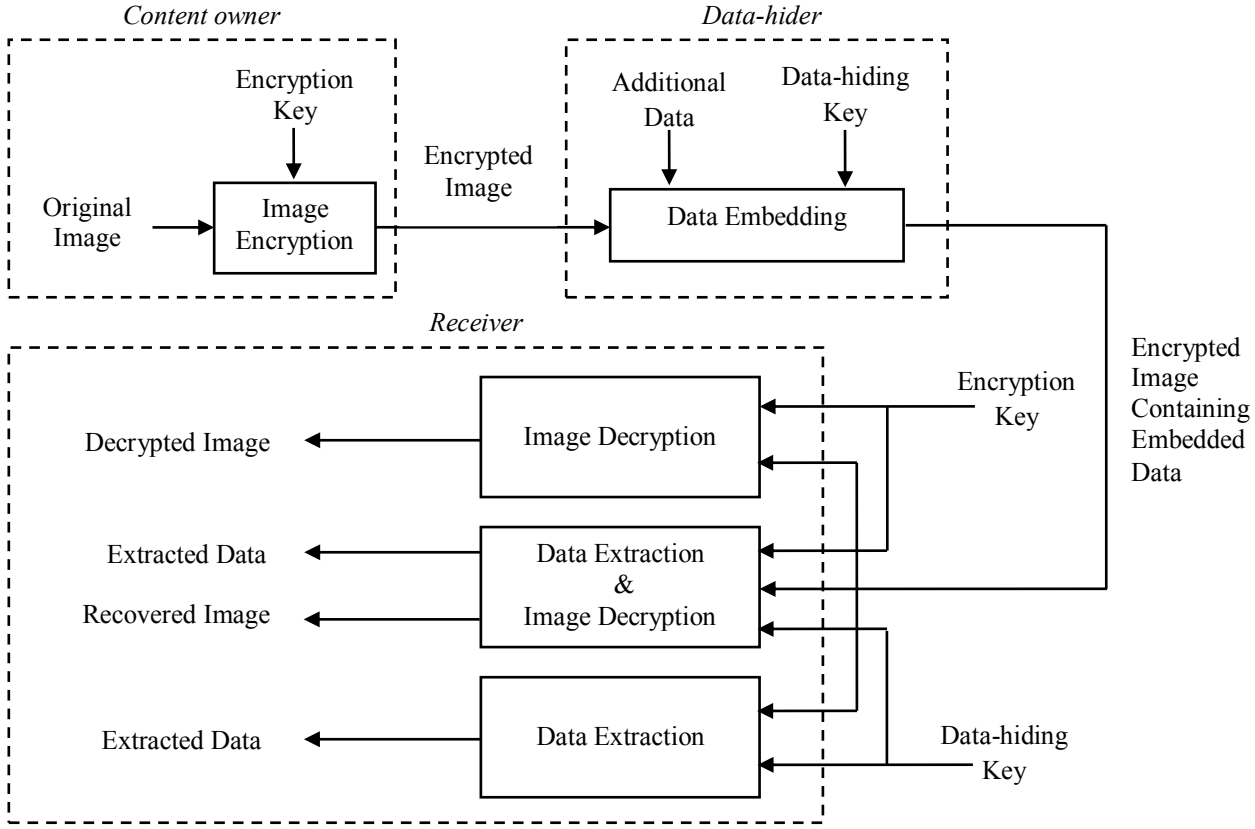

Fig. 1. Sketch of the scheme.

A. *Image Encryption*

Consider a grayscale image of size $m \times n$. Scan the image into a matrix sized $m \times n$. For a grayscale image, the pixel values fall in the range [0, 255].

We use the chaotic based permutation method in [5] to encrypt the image. A sequence of pseudorandom numbers are generated by using the logistic function,

$$a_{p+1} = \mu * a_p * (1 - a_p) \quad (1)$$

where, $0 < \mu < 4$ and $a_p \in [0,1]$. a_0 is an initial seed and combination of a_0 and μ serves as the encryption key.

Generate $N = mn$ pseudorandom numbers using (1) and store them in a vector P. Sort the vector P in ascending order. Note the position changes during sorting to generate a location map L. Location map is such that, $L(1)$ is the original index of smallest value, $L(2)$ that of second smallest and so on.

Arrange pixels of the image into a vector V. Rearrange V according to the location map L. Reshape vector V into matrix sized $m \times n$ to get the encrypted image.

B. *Data Hiding*

As permutation based encryption do not change the histogram of original image, histogram modification based method in [1] can be used to hide data into the encrypted image. The method follows:

1) Pseudorandomly permute the encrypted image pixels using data hiding key.
2) First Q pixels of the permuted image are used to hide parameters of data hiding. The parameters include, number of *MAX-MIN* pairs and *MAX-MIN* pixel values. Let there are *S MAX-MIN* pairs. Then, the number of pixels needed to hide the parameters are given by,

$$Q = (2 * S + 1) * 8 \quad (2)$$

when, eight bits are used to represent a value.

3) Generate histogram of the remaining $N - Q$ pixels and find the maximum (*MAX*) and minimum (*MIN*) points. A *MAX* point is the grayscale value having maximum number of pixels in the image. A *MIN* point is the grayscale value having minimum number of pixels in the image. Data

hiding with a single *MAX-MIN* pair is explained here. Without loss of generality, assume *MAX* < *MIN*. Data are embedded into pixels with grayscale value equal to *MAX*. The pixel positions with grayscale value equal to *MIN* are stored as overhead information and will be hidden into the image along with pure data.

4) The N - Q pixels used for generating the histogram are scanned in a sequential order. The grayscale values of those pixels in the range [*MAX*+1, *MIN*-1] are incremented by 1, leaving histogram of *MAX*+1 empty. This is the histogram shifting process.
5) Let there be p pixels corresponding to the grayscale value *MAX*. These p pixels are used to hide the data. Data to be embedded is converted into binary format. The $N - Q$ pixels are scanned in a sequential order. Whenever a pixel with grayscale value *MAX* is encountered, check the bit to be embedded. If the corresponding bit to be embedded into that pixel is "1", the pixel value is incremented by 1. If the bit to be embedded is "0", the pixel value remains unchanged.
6) The parameters of data hiding namely, number of *MAX-MIN* pairs and *MAX-MIN* pixel values, are hidden into the image by replacing the LSB of first Q pixels. The original Q LSBs are also hidden as overhead information along with pure data.
7) Perform inverse permutation on the image.

The above steps complete data hiding process. It is observed that data hiding capacity of the method is (p-O- Q) bits, where O is the amount of overhead information due to *MIN* positions. Increased data hiding capacity can be achieved by using multiple *MAX* and *MIN* pairs as explained in [1].

C. *Data Extraction and Image Recovery*

At the receiver side we will consider three cases such that the receiver has:

i. data hiding key only
ii. encryption key only
iii. both data hiding and encryption keys

i. *Data hiding key only*

When the receiver has only data hiding key and encrypted image containing hidden data, he must be able to extract the hidden data, but should not be able to read the original image.

In order to extract the hidden data, the receiver may first pseudorandomly permute the image using data hiding key as in data hiding phase. Then, extract the LSB of first eight pixels to find the number of *MAX-MIN* pairs S. LSB of next $(2 * S) * 8$ pixels are then extracted to find the *MAX-MIN* pair values. He then scans the remaining pixels in the sequential order. If a pixel value with grayscale value *MAX* + 1 is encountered, a bit "1" is extracted. If a pixel value *MAX* is encountered, a bit "0" is extracted. The extracted bits are concatenated to get the hidden data. Thus, the pure data as well as overhead information are exactly extracted.

In order to recover the original pixel values after data extraction, the whole image is scanned again in sequential order. The LSB of first Q pixels are replaced by the first Q bits of extracted overhead information.

For the remaining pixels, if the pixel value is in the range [*MAX*+1, *MIN*], the pixel value is decremented by 1. The extracted overhead information also contains the original pixel positions with *MIN* value. Replace these pixels with *MIN* value. After data extraction, inverse permutation is performed to get the encrypted image without any hidden data.

ii. *Encryption key only*

When the receiver has only data hiding key and encrypted image containing hidden data, he must be able to generate an image similar to the original image, but should not be able to read the hidden data.

The receiver can decrypt the encrypted image containing hidden data by using the encryption key only. For this, he generates the same chaotic sequence as in encryption phase using (1) and the initial conditions. The logistic map is generated by sorting the sequence in ascending order. Using the logistic map, pixels of the encrypted image are rearranged to their original position to get the decrypted image.

The decryption process can bring back all the pixels to their original position. Only distortion in the decrypted image is a difference of 1 in grayscale value for those pixels used for data hiding. The lower bound of Peak Signal to Noise Ratio (PSNR) of this decrypted image can be proved to be larger than 48 dB as follows.

It is observed that the grayscale value of pixels between *MAX* and *MIN* will either be incremented or decremented by 1 during data hiding. In the worst case, the value of every pixel differs by a value of 1 from their original value. Thus, the Mean Square Error (MSE) of worst case is 1 and the lower bound of PSNR of the decrypted image containing hidden data is given by,

$$PSNR = 10log_{10}\ \frac{255 \times 255}{MSE}$$

$$= 10\ log_{10}(255^2$$

$$= 48.13\ dB \qquad (3)$$

This resultant lower bound of PSNR is much higher than that of reversible data hiding in encrypted image techniques, [10] and [11], reported in the literature.

iii. *Both data hiding and encryption key*

When the receiver has both keys, he first extracts the hidden data using data hiding key as in case (C. i). This will

recover the original value of distorted pixels due to data hiding. Then, the receiver decrypts the image as in case (C. ii). Thus, the hidden data is extracted exactly and the original image is recovered completely.

III. EXPERIMENTAL RESULTS

We tested the proposed separable method with 4 commonly used test images namely Lake, Einstein, Lena, and Tulips, each of size 256×256. In our implementation we used two peak points of histogram to hide data. When the receiver is having the encryption key only, he can decrypt the image and the PSNR of all directly decrypted images were observed to be above 48.13dB and verifies the theoretical result. When the receiver is having data hiding key only, he can extract the hidden data without any error. When the receiver is having both encryption key and data hiding key, he can extract the hidden data exactly and recover the image. The recovered image is exactly same as the original image as in [1]. Fig.2 shows the images during different phases of the algorithm.

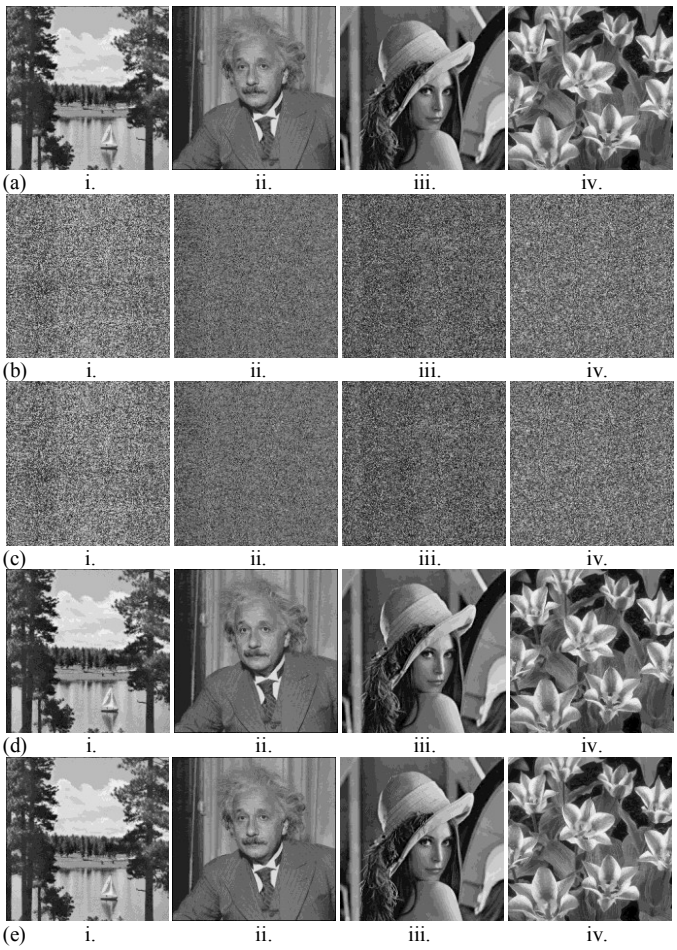

Fig. 2. Test images during different stages of the algorithm.
(a)Original images (i)Lake (ii)Einstein (iii)Lena (iv)Tulips
(b)Encrypted images (i)Lake (ii)Einstein (iii)Lena (iv)Tulips
(c) Encrypted image with hidden data (i)Lake (ii)Einstein (iii)Lena iv)Tulips
(d)Directly decrypted images (i)Lake (ii)Einstein (iii)Lena (iv)Tulips
(e)Recovered images (i)Lake (ii)Einstein (iii)Lena (iv)Tulips

Table I. shows the data hiding rate in bits per pixel (bpp), PSNR of directly decrypted images and PSNR of recovered images. The "+α" PSNR of recovered image indicates that the original image is completely recovered without any error.

TABLE I. EXPERIMENTAL RESULTS FOR SOME COMMONLY USED IMAGES

Image	Data hiding rate (bpp)	PSNR of directly decrypted image (dB)	PSNR of recovered image (dB)
Lake	0.0285	48.4735	+α
Einstein	0.0355	48.2885	+α
Lena	0.0173	52.9996	+α
Tulips	0.0204	50.6855	+α

IV. CONCLUSION

We proposed a novel separable reversible data hiding in encrypted image with improved performance. The method consists of image encryption, data hiding, and data extraction and image recovery phases. In the first phase, owner of the image encrypts the image by chaotic permutation using encryption key. The data hider without knowing the original content can hide data into the encrypted image using data hiding key. For this histogram modification based method is used. Data hiding capacity of this method is much higher than that of the data hiding methods used in existing reversible data hiding in encrypted image techniques. At the receiver side, data extraction and image recovery are performed in a separable manner. The receiver with data hiding key only can extract the hidden data, but cannot decrypt the image. The receiver with encryption key only can generate an image similar to the original image by decryption, but cannot read the hidden data. The lower bound PSNR of this decrypted image is 48.13 dB, which is much higher than that of existing reversible data hiding techniques in encrypted image. If the receiver has both keys, he can extract the data and recover the original image completely.

V. REFERENCES

[1] Zhicheng Ni, Yun-Qing Shi, Nirwan Ansari, Wei Su, "Reversible data hiding", IEEE *Trans. on Circuits and Systems for Video Technology*, vol. 16, No. 3, Mar. 2006.

[2] Jun Tian, "Reversible data embedding using a difference expansion", IEEE *Trans. on Circuits and Systems for Video Technology*, vol. 13, No. 8, Aug. 2003.

[3] Mehmet Utku Celik, Gaurav Sharma, Ahmet Murat Tekalp, Eli Saber, "Lossless generalized-LSB data embedding", IEEE *Trans. on Image Processing*, vol. 14, No. 2, Feb. 2005.

[4] Pramod Kumar, Pushpendra Kumar Pateriya, "RC4 enrichment algorithm approach for selective image encryption", *International Journal of Computer Science & Communication Networks*, vol. 2(2), 181-189.

[5] Chinmaya Kumar Nayak, Anuja Kumar Acharya, Satyabrata Das, "Image encryption using an enhanced block based transformation algorithm", *International Journal of Research and Reviews in Computer Science*, vol. 2, No. 2, Apr. 2011.

[6] M. Kiran Kumar, S. Mukthyar Azam, Shaik Rasool, "Efficient digital encryption algorithm based on matrix scrambling technique", *International Journal of Network Security & its Applications*, vol.2, No.4, Oct. 2010.

[7] Mohammad Ali Bani Younes, Aman Jantan, "An image encryption approach using a combination of permutation technique followed by encryption", *International Journal of Computer Science and Network Security*, vol.8 No.4, April 2008.

[8] Hiral Rathod, Mahendra Singh Sisodia, Sanjay Kumar Sharma, "Design and implementation of image encryption algorithm by using block based symmetric transformation algorithm", *International Journal of Computer Technology and Electronics Engineering*, vol. 1, Issue 3.

[9] Shiguo Lian, Zhongxuan Liu, Zhen Ren, Haila Wang, "Commutative encryption and watermarking in video compression", IEEE *Trans. on Circuits and Systems for Video Technology*, vol. 17, No. 6, Jun 2007.

[10] Xinpeng Zhang, "Reversible data hiding in encrypted image", IEEE *Signal Processing Letters*, vol. 18, No. 4, Apr. 2011.

[11] Xinpeng Zhang, "Separable reversible data hiding in encrypted image" IEEE *Trans. on Information Forensics and Security*, vol. 7, No. 2, Apr. 2012".