A new steganography method which preserves histogram: Generalization of LSB++

Article *in* Information Sciences · September 2014

DOI: 10.1016/j.ins.2014.02.007

2 authors:

Kazem Qazanfari
George Washington University

Reza Safabakhsh
Amirkabir University of Technology

A New Steganography Method which Preserves Histogram: Generalization of LSB^{++}

Kazem Qazanfari, Reza Safabakhsh*
Department of Computer Engineering and Information Technology
Amirkabir University of Technology
Tehran, Iran
{Kazemmit, Safa}@aut.ac.ir

Abstract— **Histogram-based steganalysis methods diagnose abnormalities in the stego histogram. LSB^{+} and outguess are two steganography methods which preserve the cover histogram completely. These methods embed some extra bits to retain the cover histogram. However, these techniques adversely affect the statistical and perceptual attributes of the cover media. LSB^{++} was proposed to improve LSB^{+} by prohibiting some pixels from changing, resulting in the reduction of the extra bits. In this paper, we improve the LSB^{++} method by proposing a technique to distinguish sensitive pixels and protect them from extra bit embedding, which causes lower distortion in the co-occurrence matrices. In addition, we extend LSB^{++} to preserve the DCT coefficients histogram of jpeg images and generalize this method to the case where more than one bit of the cover elements are used. Experimental results show that the improved LSB^{++} method produces fewer traces in the co-occurrence matrices than the LSB^{++} method. Furthermore, histogram based attacks cannot detect stego images produced by the proposed method with or without extra bits embedding. Therefore, the visual quality of the cover can be improved by elimination of extra bit embedding.**

Keywords—steganography; preserving histogram; LSB^{++} embedding, co-occurrence matirx.

I. INTRODUCTION

Steganography is the art and science of hidden communication. A steganography system embeds hidden content in unremarkable cover media so as not to arouse an eavesdropper's suspicion. Essentially, the information hiding process in a steganographic system starts by identifying a cover medium's redundant bits. The embedding process creates a stego medium by replacing these redundant bits with data from the hidden message [1]. In an effort to hide secret message in redundant bits, Yang et al.[2] proposed an adaptive LSB steganography method using adjacent pixel value differencing. This method determines the number of message bits which could be hidden into these pixels. More message bits are embedded for a higher difference value. In addition, Hong et al. [3] proposed a steganography method based on pixel pair matching (PPM). This method utilizes the values of pixel pairs as a reference coordinate. To hide message bits, this method first searches for a coordinate in the neighborhood set of this pixel pair based on the message bits. Then, this method replaces the pixel pair by the selected coordinate to embed the message bits. Matrix embedding techniques are well known data hiding methods to embed message bits in redundant bits [4-6]. Fridrich and Soukal proposed two methods based on matrix embedding[4]. The first technique is based on a family of codes constructed from simplex codes and the second one is based on random linear codes of small dimension. One of weaknesses of matrix embedding techniques is low robustness against active attacks. Considering this weakness, Sarkar et al. [5] proposed a matrix embedding hiding method using powerful repeat accumulate (RA) codes for error correction, to solve this difficulty. Furthermore, Wang et al. [6] introduced a steganography method to improve the embedding speed of matrix embedding by extending the matrix via some referential columns.

Modifying the cover medium changes its statistical properties, so eavesdroppers can detect the distortions in the resulting stego medium's statistical properties. The process of finding these distortions is called statistical steganalysis [1]. Many steganalysis methods are proposed which use these properties to detect stego images. Also Lyu and Farid [7] proposed a novel steganalysis method which extracts first and higher order magnitude and phase statistics of images to detect stego from cover images. In another effort, Fillatre [8] found an adaptive statistical test for detecting hidden bits in the Least Significant Bit whose probability distribution is independent from cover parameters. In this test, the likelihood ratio test is used and the

* Corresponding author, Tehran 15914, Iran,
Phone: (+98)(21)64542728,
Fax: (+98)(21)66495521,
Email: safa@aut.ac.ir

unknown parameters are determined using a local linear regression method. Some steganalysis methods use these properties in different ways. For example, Dumitrescu et al. [9] used a finite state machine whose states are selected as sets of sample pairs called trace multi sets. LSB embedding changes the statistical relations between the cardinalities of trace multi sets. By using these statistics changes, this method is able to detect stego images. Furthermore, Pevny et al.[10] investigated the difference between neighboring pixels before and after embedding using first and second order Markov chains. They extracted some statistical features from transition probability matrices and used a SVM classifier to detect stego images.

Histogram and co-occurrence matrices are two important statistical representations used in some steganalysis methods. These techniques utilize the changes made by data hiding to these properties to detect stego covers, and can be applied to any digital media in any embedding domain.

Westfeld and Pfitzmann [11] proposed a histogram based steganalysis method. They found that embedding process alters the frequencies of cover values. Therefore, they proposed a Chi-Square test as a method for detecting stego images from clear covers. Also, Harmsen and Pearlman [12] defined a Histogram Characteristic Function Center of Mass (HCF-COM) to detect changes in image histogram after embedding. By extracting this feature, they were able to detect stego signal. To detect the stego images under more general conditions, this method was extended by Ker [13]. He then proposed a steganalysis method for the case when the two least significant bits of the pixels are used [14]. Also, Fridrich and Goljan discussed some histogram based steganalysis methods to detect the LSB substitution method [15].

To defeat the histogram based steganalysis methods, many efforts have been made by researchers to protect the histograms of images. One of the first solutions to defeat these attack was LSB matching. LSB matching increases or decreases the pixels values with the same probabilities when the least significant bit of the pixel value is not equal to the message bit. LSB matching revised (LSBMR) [16] and LSBMR-based edge-adaptive [17] are two versions of LSB matching steganography methods. Also, Tan and Li [18] showed that the readjusting step of LSBMR-based edge-adaptive [17] produces some effects in the long exponential tail of the histogram of the absolute difference of the pixel pairs. By using these effects, they proposed a steganalysis technique that could detect stego images and could precisely estimate the used threshold in the data hiding process. Ghazanfari et al. [19] proposed an adaptive steganography method based on LSB matching method which increases the capacity up to 150%.

Sun et al. [20] presented a low capacity data hiding approach which completely preserved the histogram of the image. Also, a method with excessive complexity was proposed by Franz [21] that retains the histogram of cover, but not perfectly.

Marçal and Pereira [22] proposed a steganography method based on reversible histogram transformation functions (RHTF) for digital images. By using a secret key and RHTF, the secret information can be successfully embedded into the LSBs of an image. Lou et al. [23] showed that this method causes some artifacts; so the stego images could be detected easily. They proposed an improved version of this method by using multi embedding keys. Although both methods are secure against histogram based steganalysis, but they do not preserve the cover histogram completely.

The LSB^+ method suggested by Wu et al. [24] preserved the image histogram in spatial domain by embedding some extra bits in images. This method, however, results in statistical and perceptual distortions. Also, Provos [25] proposed a similar approach which preserves the primary histogram, but in the discrete cosine transform(DCT) domain. In our previous work [26], a new technique for image steganography, called LSB^{++}, was proposed, which improves the LSB^+ by keeping some pixels from changing, result in reducing the number of extra bits.

In this paper, we improve the LSB^{++} method by proposing a technique to distinguish sensitive pixels and keep them from extra bit embedding, as the embedding process causes fewer traces in the co-occurrence matrixes. Our previous work considers only one least significant bit of the cover elements for hiding the secret message, whereas the improved LSB^{++} method hides the secret message in more than one least significant bit of the cover elements. In addition, we present a procedure to apply our previous work to preserve the DCT coefficients histogram of jpeg images.

In the following sections, we first describe some background material including histogram and co-occurrence matrix traces, LSB^+ and our previous improvement on LSB^+ method (i.e. LSB^{++}). Then, in section 3, a method is proposed for selecting a suitable lock key among candidate keys. In section 4 the generalization of LSB^{++} to use more than one least significant bit is presented. We discuss using the LSB^{++} method in the DCT domain in section 5. The performance of the proposed method is investigated and discussed in section 6. In section 7 the experimental procedures and results are shown. Finally, concluding remarks are given in section 8.

II. BACKGROUND

In this paper, the image pixels and DCT coefficients are both called **cover elements**. Therefore, the cover elements in spatial and transform domain would be pixels and DCT coefficients, respectively. In this section, some prerequisites including histogram and co-occurrence matrix traces, LSB^+ and our previous improvement on LSB^+ (i.e. LSB^{++}) are described.

A. *Histogram trace*

Suppose that only the LSB of the cover elements is used in the embedding process. If the value of other seven bits is equal to *2i* (i= 0…127 for an 8-bit gray scale image) and the least significant bit is zero [one], then the value of the cover element would be *2i* [*2i+1*]. Figure 1 shows the changing probabilities of LSB as a FSM, where *P* is the alteration probability of the least significant bit from zero to one or vice versa. Let h_{2i} and h_{2i+1} give the frequencies of the cover element values *2i* and *2i+1*, respectively. After the embedding process, these frequencies change to h^*_{2i} and h^*_{2i+1}, respectively. Eqs. (1), (2), and (3) show how these frequencies chang [26]:

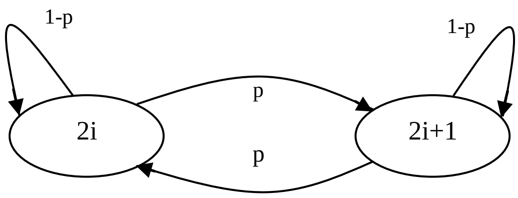

Figure 1. Finite state machine of LSB alternation [26].

$$h^*_{2i} = (P)h_{2i+1} + (1-P)h_{2i} \tag{1}$$

$$h^*_{2i+1} = (P)h_{2i} + (1-P)h_{2i+1} \tag{2}$$

$$|h^*_{2i} - h^*_{2i+1}| = |1-2P|\,|h_{2i} - h_{2i+1}| \tag{3}$$

Generally, the secret message is encrypted before data hiding process; therefore, the occurrence probabilities of 0 and 1 in the encrypted message are equal. As the length of the encrypted message increases, the probability *P* converges to *0.5*; therefore, |*1-2P*| is a very small value. This shows the decrease of the frequency differences of cover values *2i* and *2i+1*. Histogram based steganalysis methods use these traces to detect stego signals [26].

B. *Co-occurance Matrix trace*

This property is a two-dimensional matrix, each entry of which gives the frequency for the co-occurrence of two values at two cover elements separated by a fixed distance and direction. Relations (Δx, Δy) describe distance and direction. Since the relations between pairs of cover elements are considered, this property is considered as second-order statistics. For given (Δx, Δy), the entry c_{ij} of a co-occurrence matrix describes the frequency of the pairs of cover elements satisfying Eq. (4):

$$(g(x,y) == i)and(g(x+\Delta x, y+\Delta y) == j) \tag{4}$$

Therefore, there is one co-occurrence matrix for each pair (Δx, Δy). In a real world cover, the values of the neighboring cover elements are close to each other, making the dependency of adjacent cover elements very high. Therefore, many steganalysis methods compute the co-occurrence matrices only for some or all of the close neighboring cover elements from the following set [27-29]:

$$(\Delta x, \Delta y) \in [(1,1),(-1,1),(0,1),(1,-1),(-1,-1),(0,-1),(1,0),(-1,0)]$$

Since the value of a cover element is closer to its neighboring elements' values than other cover elements' values, the values of entries close to the main diagonal of the co-occurrence matrix (called **vulnerable entries**) are greater than the other entries. Furthermore, since the embedding process reduces the closeness of adjacent cover elements' values, after embedding, the values of **vulnerable entries** are reduced and spread to other entries. The co-occurrence based steganalysis methods use these variations to detect stego signals. Figure 2 shows a typical representation of this matrix.

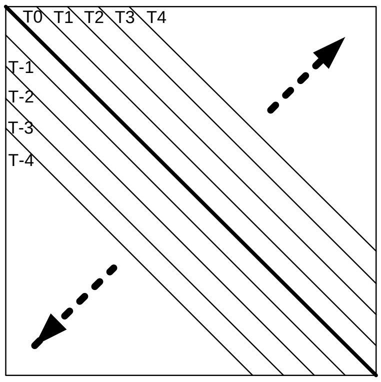

Figure 2. A typical representation of co-occurrence matrix.

T_0 is the main diagonal of this matrix. Other entries are placed on $T_1T_2…T_{MAX}$ and $T_{-1}T_{-2}…T_{-MAX}$. Max shows the maximum possible value of cover elements.

C. *LSB^+* method

The LSB^+ algorithm considers each value of the cover elements as a *bin* and defines *unit* as a set of *2 adjacent bins*. Therefore, for an 8-bit gray scale image, there are 256 (0…255) *bin*s and 128 (0…127) *units*. In addition, *bins 2i* and *2i+1* (*i*=0...127) are placed in the *unit I* twice. Suppose also that h_{2i} and h_{2i+1} are the pixel frequencies with values *2i* and *2i+1*, respectively.

The LSB^+ method, in the first step, embeds and extracts the secret message based on the *unit*s' frequencies. Considering *unit i*, this method embeds h_{2i} (or h_{2i+1}) zeros (or ones) of the encrypted message in the cover [26].

In the second step, LSB$^+$ embeds some extra 0s or 1s in unused cover elements in order to preserve the image histogram completely. This process causes additional distortions in the cover, as compared to the LSB substitution method. This method extracts the message bits from each *unit* until all LSBs of one of its *bin*s are extracted [26]. After processing all *units*, the message extraction is complete.

D. LSB^{++} Method

LSB^{++} prohibits some cover elements from changing using a locking process. Locked cover elements are not to be selected in the embedding process [26]. To lock the appropriate cover elements, first the cover histogram and the frequency difference of the two adjacent *bin*s in a *unit* are computed. Then, the method locks some cover elements for each *unit* based on a lock key. Consider *unit i* and its *bins 2i* and *2i+1*. Suppose that $A_i = |h_{2i} - h_{2i+1}|$ gives the frequency difference of these two *bins*. Then, the locking process locks A_i cover elements with value $2i$ [$2i+1$] when $h_{2i} > h_{2i+1}$ [$h_{2i} < h_{2i+1}$] .

In the embedding process, for each *unit i*, embedding the encrypted message bits continues until Min (h_{2i}, h_{2i+1}) zeros [or ones] are embedded in the *bin 2i* [or *2i+1*].

As mentioned before, the occurrence probabilities of zero and one are almost equal. In addition, since the frequencies of the unlocked cover elements with values *2i* and *2i+1* are equal, the difference between frequencies of two adjacent *bin*s *2i* and *2i+1* may be scarcely changed after the embedding process. Finally, to preserve the cover histogram completely, it is needed to embed α_{2i} zeros or α_{2i+1} ones (i=0…127) in the unlocked and unused elements in each *unit i*. As explained in [20], α_i (i=0…255) is a very small value, resulting in remarkable reduction of distortions caused by the extra embedding [26].

To extract the encrypted message, first the locked elements have to be determined based on the lock key. Then, the encrypted message bits are extracted using the embedding key. Similar to the embedding process, for each *unit i*, extracting the encrypted message bits continues until Min (h_{2i}, h_{2i+1}) zeros [or ones] are extracted from the *bin 2i* [or *2i+1*].

III. SELECTION OF THE LOCK KEY

The statistical information of media covers is generally divided into first and higher order statistical properties. Histogram is the most important first order statistical information. In our previous work, a new method which preserves the histogram of cover media is illustrated [26]. But there are some novel steganalysis methods which use higher order statistical information. The higher order statistical properties consider the dependency of cover elements. The co-occurrence matrices provide important higher order statistical information which has been used in several steganalysis methods [27-29]. As mentioned in the previous section, such methods use only some entries of these matrices, called **vulnerable entries**. LSB^{++} method locks some cover elements based on a lock key. Locked elements are not used in the embedding process. If the locked elements are selected ingeniously so that the **vulnerable entries** change less, the detection precision of the corresponding steganalysis methods will be reduced. In this section, an efficient technique for selecting the best lock key among all possible keys is proposed. This technique is based on preserving the **vulnerable entries** from changing. The details are described in the following lines.

Suppose that n cover elements in cover **C** must be locked. Also suppose that there are m keys from which one can be used as the lock key. Then key_k locks the following cover elements:

$$C^k = \{C^{k,1}, C^{k,2} \dots C^{k,n}\};\ k=1, 2 \dots m$$

As mentioned before, co-occurrence matrix based steganalysis methods consider only eight neighbors of each cover element. Therefore, to select the best lock key, our technique considers these neighbors for each cover element. In other words, for each cover element $C^{k,x}$(i, j), the following cover elements must be considered:

$$C^{k,x}(i+u, j+v);\ u, v = -1, 0, 1$$

If changing the value of a cover element results in significant effects in the **vulnerable entries**, this cover element is called a sensitive cover element. Therefore, the best lock key is the key which locks the largest number of sensitive cover elements. Eq. (5) can be used as a measure to compute the sensitivity of a cover element $C^{k,x}$ (i, j):

$$D_{k,x}(i,j) = \sum_{u\in(-1,0,1)} \sum_{v\in(-1,0,1)} (|C^{k,x}(i,j) - C^{k,x}(i+u, j+v)|)^p \tag{5}$$

where p is the sensitivity parameter. As mentioned before, the entries near the main diagonal of the co-occurrence matrix are more important than the other ones. This parameter controls this feature. Finally, Eq. (6) can be used as a measure to evaluate the lock key k. It is expected that the *S-measure* value for the best lock key be minimum among other lock keys.

$$S-measure_k = \sum_{x=1..n} D_{k,x}(i,j) = \sum_{x=1..n} \sum_{u\in(-1,0,1)} \sum_{v\in(-1,0,1)} (|C^{k,x}(i,j) - C^{k,x}(i+u, j+v)|)^p \tag{6}$$

IV. GENERALIZATION OF LSB^{++} METHOD

Generally, only the LSB bit of the cover pixels or coefficients is used to conceal the secret message. However, some steganography methods use more than one bit for this purpose. In this section, the extension of the proposed method for when more than one bit are used in steganography is presented.

Extended definitions of *bin* and *unit*, described in section II(c), will be used here. Suppose that S LSB bits are used in the embedding process. Therefore, there are $B=2^S$ *bin*s in each *unit*. Suppose that the cover elements range from 0 to *MAX*. For example, if an 8-bits gray scale image is used as the cover, *MAX* will be 255. Based on the above definitions, there are $M=MAX+1$ *bin*s (from 0 to $M-1$). Furthermore, there are N *unit*s (from 0 to $N-1$) where N is given by Eq. (7):

$$N = Number\ of\ units = \left\lfloor M/B \right\rfloor \quad (7)$$

Therefore, the following *bin*s are placed in *unit* i:

$$B\times i,\ B\times i+1, \ldots,\ B\times i+B-1$$

For example, if S and *MAX* are 2 and 255 respectively, then B=4, M=256, N=64 and *bin*s(0,1,2,3) are placed in *unit* 0, *bins* (4,5,6,7) are placed in *unit* 1, …, and *bin*s(252,253,254,255) are placed in *unit* 63.

Now, suppose that the *bin* frequencies of *units i* $i=0\ldots N-1$ are $h_{B\times i}$, $h_{B\times i+1}, \ldots$, and $h_{B\times i+B-1}$, respectively. The minimum frequency of *unit i* is computed by:

$$Min_i=Min\ (h_{B\times i}, h_{B\times i+1}, \ldots, h_{B\times i+B-1}),\ i=0\ldots N-1 \quad (8)$$

Then for each *bin* of *unit I*, i.e. $B\times i$, $B\times i+1$, … , $B\times i+B-1$, locks $h_{B\times i} - Min_i$, $h_{B\times i+1} - Min_i$, … , and $h_{B\times i+B-1} - Min_i$ cover elements having values $B\times i$, $B\times i+1$, … , $B\times i+B-1$, respectively. Eq. (9) shows the locking process:

$$Cover^{\wedge}=Lock\ (Cover, [h_{B\times i} - Min_i, B\times i], [h_{B\times i+1} - Min_i, B\times i+1] \ldots [h_{B\times i+B-1} - Min_i, B\times i+B-1], lock\ key),\ i=0, 1\ldots N-1 \quad (9)$$

The output of this step is $Cover^{\wedge}$. In the embedding step, the locked cover elements will not be used for data hiding. After the locking process, the frequencies of unlocked cover elements with values $B\times i$, $B\times i+1, \ldots$, $B\times i+B-1$ are equal to Min_i. Figure 3 shows this process. In this figure, S and B are equal to 2 and 4, respectively. The left hand side histogram is the histogram of the cover and the right hand side one is the histogram of unlocked cover elements of $Cover^{\wedge}$.

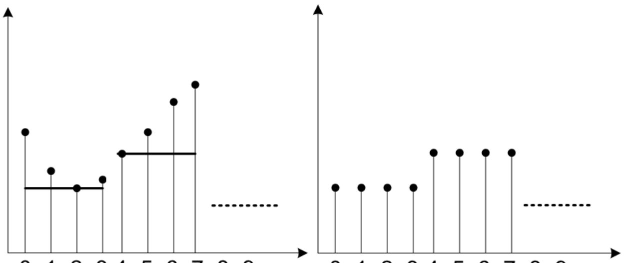

Figure 3. Left: histogram of cover, right: histogram of unlocked cover elements of $Cover^{\wedge}$.

Before embedding the secret message, this message has to be encrypted using an encryption key that generates $Message^*$ as Eq. (10):

$$Message^*=Encrypt\ (Secret\ message, Encryption\ key) \quad (10)$$

Then, the $message^*$ bits are embedded in the unlocked cover elements of $Cover^{\wedge}$ using the embedding key. For *unit i*, embedding the $message^*$ bits continues until Min_i binary strings with length S are embedded in one of the *unit*'s *bin*s. It is given by Eq. (11):

$$Stego^+=Embed\ (Cover^{\wedge}, Message^*, Embedding\ key) \quad (11)$$

Since the occurrence probabilities of zeros and ones are almost equal, and the frequencies of unlocked cover elements with values $B\times i$, $B\times i+1\ldots$ $B\times i+B-1$ are equal, the frequency difference of *unit bin*s may be scarcely changed by the embedding process. So, the histogram based steganalysis methods will not be able to detect the $stego^+$ images. This claim which is verified in experiments section is expressed by Eqs. (12) and (13). In these equations, L is the length of $message^*$, h_i and h_i^+ are the frequencies of elements with value i in the cover and $stego^+$, respectively. In addition, α_i is the difference between h_i and h_i^+ and is a very small positive value.

$$P\ (Message^*_i=1) \approx P\ (Message^*_i=0),\ i=1\ldots L \quad (12)$$

$$|h_i^+ - h_i|=\alpha_i\ ,\ i=0\ldots MAX \quad (13)$$

To protect the histogram of the cover image completely, α_i binary strings with length S have to be embedded in the unlocked and unused elements of $Stego^+$ in each *unit i* as Eq. (14):

$$Stego^*=Intentional\_Embedding\ (Stego^+, \alpha_i, \{0,1\}^*),\ i=0\ldots MAX \quad (14)$$

After embedding extra binary strings, the histograms of $stego^*$ and cover will be identical as shown by Eq. (15).

$$|h_i^* - h_i|=0\ ,\quad i=0\ldots MAX \quad (15)$$

where, h_i^* is the frequency of cover elements having the value i in the $Stego^*$. To extract the message bits from $Stego^*$, unlocked elements must be distinguished from locked elements of $Stego^*$ as Eq. (16).

$$Stego^{\wedge}=Lock\ (Stego^*, [h^*_{B\times i} - Min_i, B\times i], [h^*_{B\times i+1} - Min_i, B\times i+1] \ldots [h^*_{B\times i+B-1} - Min_i, B\times i+B-1], lock\ key),\ i=0, 1\ldots N-1 \quad (16)$$

where h^*_j is the frequency of *bin j* for $Stego^*$ and Min_i is computed using Eq. (8) but for $stego^*$. Then, the $message^*$ bits are extracted using the extraction key as Eq. (17):

$$Message^*=Extract\ (Stego^{\wedge}, Extraction\ key) \quad (17)$$

Bit extraction in *unit i* continues until Min_i binary strings with length S are extracted from one of its *bin*s. After extracting the encrypted message, this message is decrypted using the decryption key as Eq. (18).

$$Message = Decrypt\ (Message^*, Decryption\ key) \quad (18)$$

V. ALLPYING THE METHOD IN THE DCT DOMAIN

In this section, we focus on data hiding in the quantized DCT coefficients of Jpeg images using the proposed technique. The Discrete Cosine Transform (DCT) is used for Jpeg images to transform them into the frequency domain. It does this by grouping the pixels into 8×8 blocks and transforming them from 64 values into 64 frequency components. The next step of Jpeg compression quantizes these frequencies using a quantization table. The goal of this step is to eliminate the high frequency values. To illustrate how this processes works, consider Figure 4. This figure shows an example of applying DCT to an 8×8 image block and quantization of computed coefficients. From left, the first matrix is an 8×8 block of the image data, the second one shows the DCT coefficients before quantization, the third part a typical quantization table, and the rightmost part gives the quantized DCT coefficients.

5. Encrypting the secret message using encryption key (Eq. 10).
6. Embedding:
   a. Embed the encrypted secret message using embedding key in unlock cover elements (Eq. 11).
   b. Compute difference between the cover (quantized DCT coefficients) histogram and *stego*$^+$ histogram (Eq. 13).
   c. For each *bin* i, embed a_i extra binary strings with length S to preserve the primary histogram completely (Eq. 14).

The extraction process of the proposed method is done using the following steps:

1. Providing cover elements:
   a. Extract the quantized DCT coefficients from the *stego**.
2. Locking *Stego** elements using selected lock key (Eq. 16).
3. Extracting message based on the extraction key.

(a)

81	79	75	70	63	59	42	35
80	77	73	65	59	55	39	30
78	75	69	60	53	50	34	28
75	71	65	52	48	43	30	25
70	66	61	49	43	39	25	21
64	62	57	50	37	35	22	16
58	56	52	44	32	31	17	15
52	51	46	39	28	24	13	11

(b)

390	129	-13	10	-7	-2	8	-5
78	0	-5	6	0	-1	0	1
-2	-5	-4	-1	1	1	0	-2
7	0	-2	0	0	1	0	-1
-2	-1	1	0	0	-1	0	1
2	0	0	0	0	0	0	0
0	0	0	0	0	0	-1	-1
0	0	0	0	0	0	1	0

(c)

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	54	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	100	121	120	101
72	92	95	98	112	100	103	99

(d)

24	11	-1	0	0	0	0	0
6	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Figure 4. An example of applying DCT to an 8×8 image block and quantization of computed coefficients. (a): An 8x8 block of image data, (b): DCT coefficient before quantization, (c): typical quantization table, (d): quantized DCT coefficients.

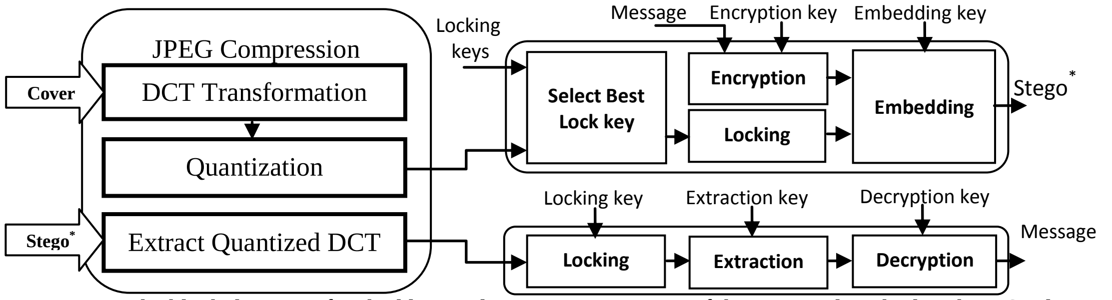

Figure 5. The block diagram of embedding and extraction processes of the proposed method in the DCT domain

Figure 5 shows the block diagram of the proposed embedding and extraction processes in the DCT domain. The embedding process includes the following steps:

1. Providing cover elements:
   a. Transform cover image into DCT domain (Figure 4 – a, b).
   b. Quantize computed DCT coefficients using a quantization table (Figure 4 – c, d).
2. Selecting best lock key among all candidate lock keys using the procedure proposed in section III.
3. Computing the cover histogram and determining Min_i for each *unit i* (Eq. 7, 8).
4. Locking cover elements using selected lock key (Eq. 9).

   a. Extract the encrypted secret message using extraction key from the unlock elements of *Stego** (Eq. 17).
4. Decrypting the secret message using decryption key (Eq. 18).

VI. PERFORMANCE EVALUATION

Suppose that S LSB bits are used for hiding the secret message and suppose that h_i, i=1…2^S are the frequencies of *bin*s in a *unit* and *Min* is the minimum value among h_i, i=1…2^S. The maximum and minimum lengths of the secret message which could be hidden in this *unit* are 2^S×*Min* and *Min* bits, respectively. In addition, the maximum and minimum capacities are obtained when h_i s (i.e. h_1,

$h_2 \ldots h_{2^S}$) of this *unit* are equal and when *Min* is equal to zero, respectively.

Furthermore, suppose that h_i, $i=1\ldots2^S$ are the frequencies of *bin*s in a cover *unit*, $\hat{h}_i$, $i=1\ldots2^S$ are the frequencies of *bin*s in the same *unit* of the $\widehat{cover}$ (only unlocked cover elements) and h^+_i, $i=1\ldots2^S$ are the frequencies of *bin*s in the same *unit* of the *Stego*$^+$ (all of the locked and unlocked elements of *Stego*$^+$). Since the encrypted message is pseudorandom (P(Message$^*_i$=1) ≈ P (Message$^*_i$=0), $i=1\ldots L$) and the frequencies of the unlocked cover elements for a *unit* are equal ($\hat{h}_i == \hat{h}_j$; $i,j=1\ldots2^S$); the embedding process may hardly change the frequency difference of same *bin* in the cover and *Stego*$^+$; i.e. $|h^+_i - h_i| = \alpha_i$, $i = 1\ldots2^S$ where α_i is a very small positive value. In addition, experimental results show that without using extra bits, the *stego*$^+$ histogram is similar to the cover histogram without any remarkable difference and histogram based attack could not detect the *stego*$^+$ images. In this situation, intentional changes would be zero, but the receiver has to have the cover histogram either by having the main cover image or any information that could be used to extract the cover histogram.

To preserve the cover histogram completely, let us consider $D=\sum a_i=\sum|h_i^+ - \mathrm{h_i}|, i=1...2^s$ as the number of unlocked and unused elements in a *unit* of the *Stego*$^+$. As mentioned before, since α_i is a very small positive value, the summation of them (i.e. D) would be small, too. Therefore, the maximum and minimum numbers of extra bits made to a *unit* will be $D\times S$ and zero, respectively.

The remaining of this section compares the time complexity of the proposed method to the LSB$^+$ and Outguess methods. Since LSB$^+$ and Outguess use only one LSB bit, the comparison is presented for when S=1. Suppose that h_i and $h_{j=i+1}$ ($i=2k$, $k=0\ldots N-1$) are the frequencies of two bins in a cover *unit*. Without loss of generality, suppose that $h_i \geq h_j$. Also D is the number of unlocked and unused elements in the *stego*$^+$ for this *unit*. The embedding steps of the LSB$^+$ and Outguess methods are as follows:

- S-1) Computing the histogram of the cover.
- S-2) Encrypting the secret message.
- S-3)For each *unit*, embedding process continues until h_i or h_j cover elements are used.
- S-4) For each *unit*, $h_i - h_j - 1$ to $h_i - h_j - 1 + D$ *extra bits* are embedded in unused cover elements.

The embedding steps of the proposed method are as follows:

- P-1) Computing the histogram of the cover.
- P-2) Encrypting the secret message.
- P-3) For each *unit*, Lock ($h_i - h_j$) cover elements.
- P-4) For each *unit*, continue embedding until h_j cover elements are used.
- P-5) For each *unit*, embed *0* to D_i extra bits in the unused and unlocked cover elements.

The first two steps (i.e. S-1, S-2 and P-1, P-2) of both techniques have identical time complexities. Also, based on our consideration i.e. $h_i \geq h_j$, then, the time complexity of P-4 is always less than S-3. The extra embedding process of our method (P-5) runs *0* to D_i times, while this process for other methods (S-4) runs $h_i - h_j - 1$ to $h_i - h_j - 1 + D$ times. Therefore, step P-5 runs almost $h_i - h_j$ times less than S-4. This reduction in run time could compensate for step 3 of our method (i.e. P-3), which runs $h_i - h_j$ times to lock some cover elements. As a result, it can be concluded that the time complexities of these methods are almost equal.

VII. EXPERIMENTS

We used the images of UCID [30] database (Version 2 – includes 1338 images with different textural properties) in our experiments. Each image is converted to grayscale, and resized to 500×500 pixels. Figure 6 shows four images of this database.

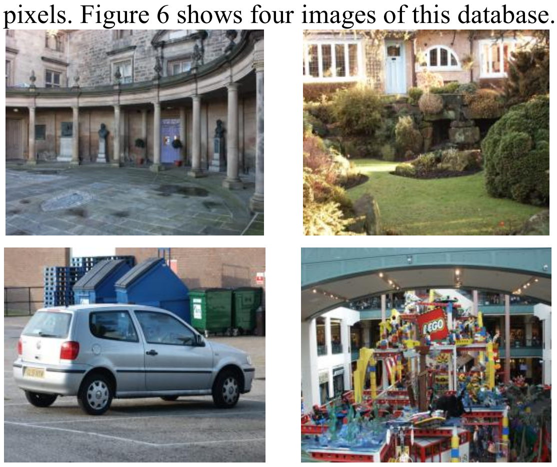

Figure 6. Four images of UCID database [30].

In the first experiment, we show that how our improvement on LSB^{++} (selecting best lock key among all candidate lock keys) reduces data hiding traces in co-occurrence matrixes. For this, we hide a pseudorandom message with 0.3 and 0.6 bpp (bit per pixel) in the test images and calculate the average energy of vulnerable entries to the main diagonal of the co-occurrence matrix for pairs (Δx , Δy) ϵ [(-1 , -1), (-1 , 1), (1 , -1), (1 , 1), (0 , -1), (0 , 1), (-1 , 0), (1 , 0)] using Eqs. (19) , (20) and (21):

$$E_i = \sum_{j=1}^{1338} \sum_{(\Delta x,\Delta y)\in[(1,1),(-1,1),(0,1),(1,-1),(0,-1),(-1,-1),(1,0),(-1,0)]} C_{i,j}(\Delta x,\Delta y) \tag{19}$$

$$C_{i,j}(\Delta x,\Delta y)=\sum_{x=1}^{500}\sum_{y=1}^{500}[(im_j(x,y)-im_j(x+\Delta x,y+\Delta y))==i] \tag{20}$$

$$T_i = E_i \,/ \sum_{k=-255}^{255} E_k \tag{21}$$

Figure 7 shows the average energies of vulnerable entries for the cover and stego images produced by the proposed method with 5 different lock keys.

The following results could be concluded from this figure:

*a) As the message size is increased, the amount of **vulnerable entries** energy that spread through other entries is increased.*

b) Different lock keys cause different traces in the co-occurrence matrixes.

*c) In this experiment, our method using the lock key (d) changes the energy of **vulnerable entries** less than other keys.*

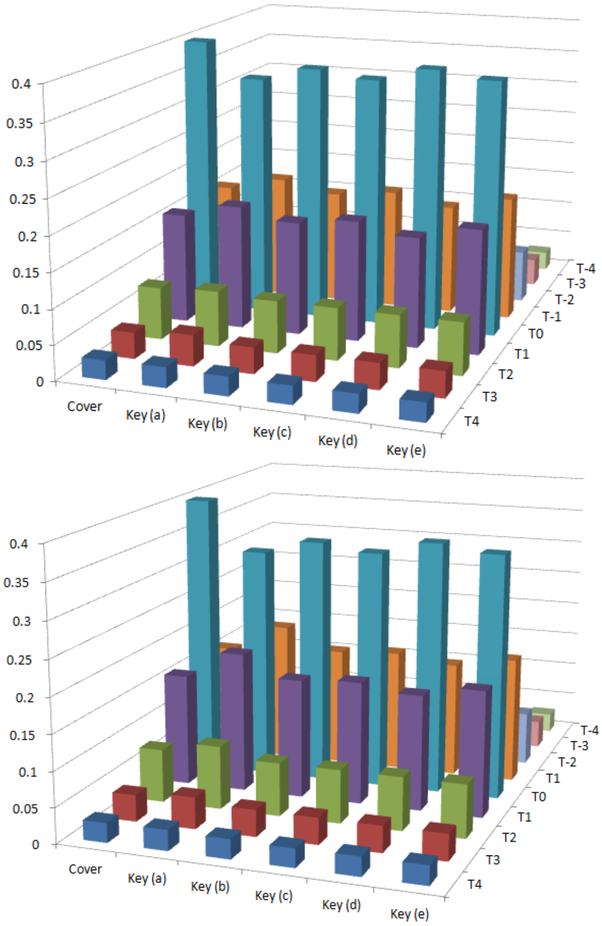

Figure 7. The average energies of **vulnerable entries** for cover and stego images produced by proposed method with 5 different lock keys (a-e). (A): message length 0.3 bpp, (B): message length 0.6 bpp.

In the next experiment, we show that the Chi-Square attack cannot detect stego images produced by our method either using intentional embedding (ILSB^{++}-B) or not using intentional embedding (ILSB^{++}-A). Therefore, we hide a pseudorandom message with 0.1- 0.8 bpp in the test images with the proposed and the LSB substitution methods. Figure 8 shows that the Chi-Square attack is in general unable to detect stego images produced by the proposed method for both situations, although it could weakly detect stego images produced by ILSB^{++}-A when the message size is higher than 0.7 bpp. The reason of these results is described in section VI (Performance Evaluation) where we explained how large the min-max number of intentional embedding bits is. In addition, since LSB substitution is not equipped with any histogram preserving technique, the Chi-Square attack can detected the LSB substitution method easily even in low capacities.

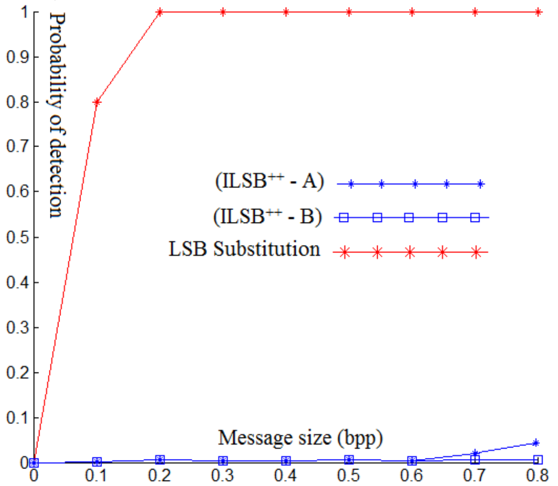

Figure 8. The results of Chi-Square attack against our method when intentional embedding is used (ILSB^{++} - B) and when intentional embedding is not used (ILSB^{++} - A) and LSB substitution.

Figure 9 shows a small section of histograms for a UCID image (A), stego image produced by LSB substitution method (B) and stego images produced by improved LSB^{++} with (C) and without (D) intentional embedding. It shows that the histogram of the stego image produced by the improved LSB^{++} without any extra bits is similar to the histograms of cover and stego image produced by improved LSB^{++} with extra bits embedding.

In the next experiment we compare the proposed method to some histogram attack-secure steganography methods including RHTF [22], improved RHTF [23], and LSB^{+} [18] in terms of visual quality.

In the RHTF method [22], first a histogram transformation function f^* is applied to the cover image. This function f^* compresses the range of levels used in the original image to a narrower range using the constant parameter α. The secret message is embedded in the resulting image f^*, producing an image E (f^* (image)). The final stego image is obtained through the application of a histogram transformation function f which expands the range of values back to the initial range. Lou et al. in their method [23] showed that the RHTF method cause to some artifacts and the stego images can be detected easily. Therefore, they proposed an improvement version of this method by splitting image into multi segments and using different parameter α for each segment.

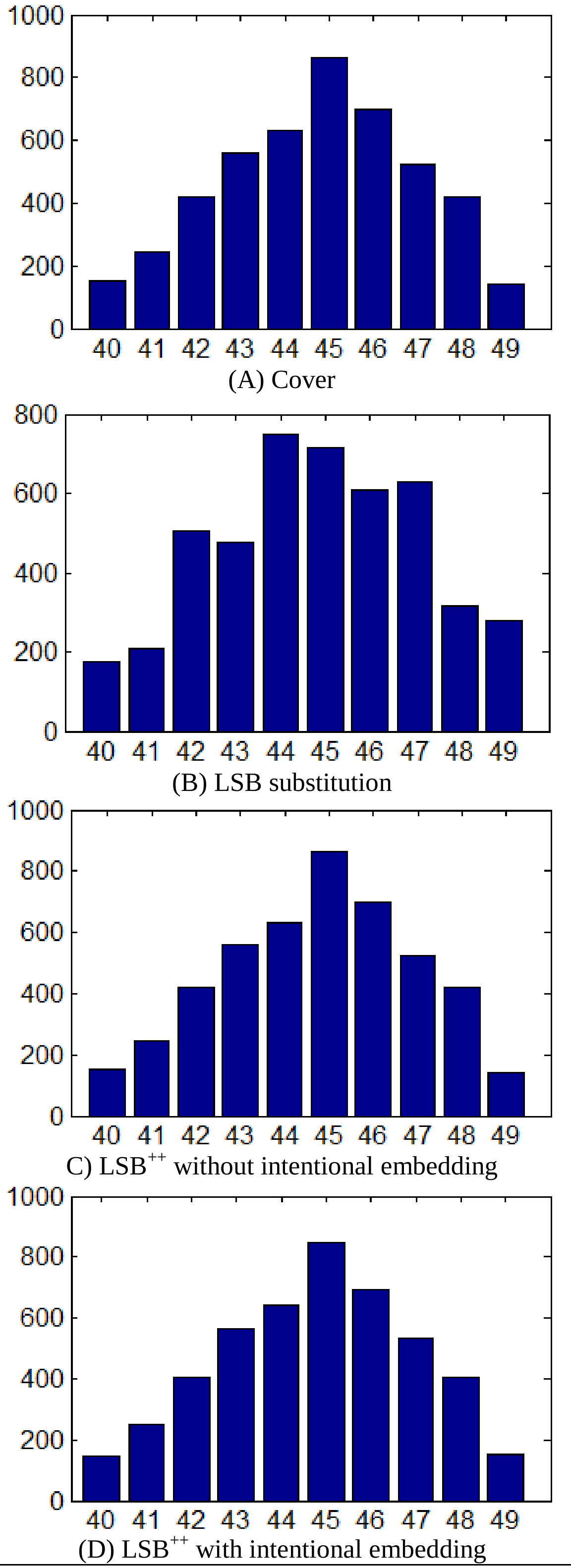

Figure 9. Histograms of Lena image (A), stego image produced by LSB substitution method (B), stego images produced by improved LSB^{++} with (C) and without (D) intentional embedding.

Peak Signal-to-Noise Ratio (PSNR) criterion is used to evaluate and compare steganalysis methods in terms of visual quality. PSNR is computed using Eq. (22) and (23):

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j)-k(i,j)]^2 \tag{22}$$

$$PSNR = 10.\log_{10}\left(\frac{MAX_I^2}{MSE}\right) \tag{23}$$

Figure 10 shows the computed PSNR vs. the embedded message length in the range of 0.1-0.8 bpp. The following results are concluded from this figure:

a) *The perceptual distortion by our method* (ILSB^{++}-A and ILSB^{++}-B) *is less than other methods.*
b) *The perceptual distortion by our method with international embedding is slightly higher than without intentional embedding. This happens because of extra embedding a_i bits for preserving image histogram perfectly, although stego images produced by our method without intentional embedding could not be detected by the Chi-Square attack method.*
c) *The perceptual distortion by* RHTF [22] *and improved* RHTF [23] *is remarkably higher than that in other methods. These methods change the pixel values twice. First, when the range of pixel value levels is compressed to a narrower range, and second when the secret message bits are embedded in the compressed pixels values.*

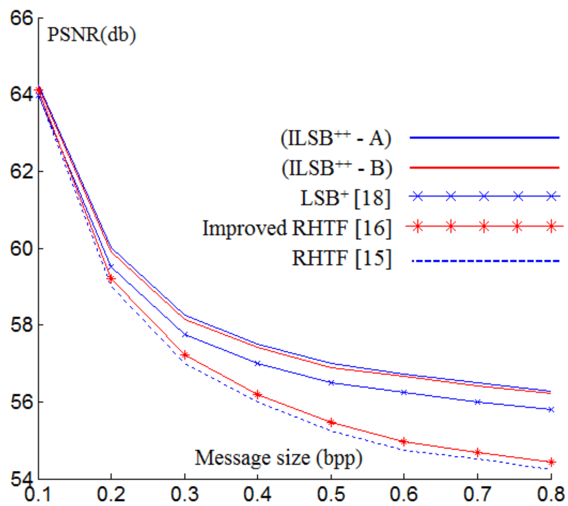

Figure 10. The PSNR versus the embedded message length for RHTF [15], improved RHTF [16], LSB^{+} [18], improved LSB^{++} without intentional embedding (ILSB^{++} - A) and improved LSB^{++} with intentional embedding (ILSB^{++} - B).

I. Conclusions

Histogram is the most important first order statistics used by some steganalysis methods. LSB^{+} and outguess are two well-known steganography methods that are secured against these attacks. Our previous work (LSB^{++}) was an improvement on LSB^{+} method which prohibits some cover elements from changing, resulting in reducing the amount of extra bits.

The LSB^{++} is secured against histogram based attacks; but there are some steganalysis methods which use higher order statistics such as the co-occurrence matrices. Co-occurrence matrix based steganalysis methods use vulnerable entries of these matrices to detect stego images. LSB^{++} locks some cover elements based on a lock key. The locked elements are not used in the embedding process. If the locked elements are selected ingeniously so that the vulnerable entries change less, the detection precision of co-occurrence matrices based steganalysis methods will be reduced. Therefore, this paper proposes an efficient technique to distinguish the sensitive cover elements and keep them from extra bit embedding, which causes lower distortions in the co-occurrence matrices. Also, a generalization of LSB^{++} method was presented for when more than one least significant bit of the cover elements are used as carrier. Finally an extension of LSB^{++} which preserves the DCT histogram of jpeg images is illustrated. Experimental results show that the improved LSB^{++} results in fewer traces in the co-occurrence matrices than LSB^{++} and the cover histogram is preserved completely. In addition, the Chi-Square attack cannot detect stego images produced by our method with or without extra bits embedding. Therefore, by eliminating the extra bits embedding, the visual quality of the improved LSB^{++} is improved.

REFERENCES

[1] Provos, Niels, and Peter Honeyman. "Hide and seek: An introduction to steganography." Security & Privacy, IEEE 1, no. 3 (2003): 32-44.

[2] Yang, Cheng-Hsing, Chi-Yao Weng, Shiuh-Jeng Wang, and Hung-Min Sun. "Adaptive data hiding in edge areas of images with spatial LSB domain systems." Information Forensics and Security, IEEE Transactions on 3, no. 3 (2008): 488-497.

[3] Hong, Wien, and Tung-Shou Chen. "A novel data embedding method using adaptive pixel pair matching." Information Forensics and Security, IEEE Transactions on 7, no. 1 (2012): 176-184.

[4] Fridrich, J.; Soukal, D., "Matrix embedding for large payloads," Information Forensics and Security, IEEE Transactions on , vol.1, no.3, pp.390,395, Sept. 2006

[5] Sarkar, Anindya, Upamanyu Madhow, and B. S. Manjunath. "Matrix embedding with pseudorandom coefficient selection and error correction for robust and secure steganography." Information Forensics and Security, IEEE Transactions on 5, no. 2 (2010): 225-239.

[6] Wang, Chao, Weiming Zhang, Jiufen Liu, and Nenghai Yu. "Fast matrix embedding by matrix extending." Information Forensics and Security, IEEE Transactions on 7, no. 1 (2012): 346-350.

[7] Lyu, Siwei, and Hany Farid. "Steganalysis using higher-order image statistics." Information Forensics and Security, IEEE Transactions on 1, no. 1 (2006): 111-119.

[8] Fillatre, Lionel. "Adaptive steganalysis of least significant bit replacement in grayscale natural images." Signal Processing, IEEE Transactions on 60, no. 2 (2012): 556-569.

[9] Dumitrescu, Sorina, Xiaolin Wu, and Zhe Wang. "Detection of LSB steganography via sample pair analysis." Signal Processing, IEEE Transactions on 51.no. 7 (2003): 1995 - 2007

[10] Pevny, Tomas, Patrick Bas, and Jessica Fridrich. "Steganalysis by subtractive pixel adjacency matrix." information Forensics and Security, IEEE Transactions on 5, no. 2 (2010): 215-224.

[11] A. Westfeld and A. Pfitzmann, "Attacks on steganographic systems" 3rd International Workshop on Information Hiding , vol. 1768 pp. 61--76, Oct. 1999.

[12] J. J. Harmsen and W. A. Pearlman, "Steganalysis of Additive Noise Modelable Information Hiding," In Proc. SPIE Electronic Imaging, Santa Clara, CA, vol. 5022, pp. 131--142, Jan. 2003.

[13] A. D. Ker, "Steganalysis of LSB matching in grayscale images," IEEE Signal Process. Letters, vol. 12, pp. 441--444, Jun. 2005.

[14] Ker, Andrew D. "Steganalysis of embedding in two least-significant bits." Information Forensics and Security, IEEE Transactions on 2, no. 1 (2007): 46-54.

[15] J. Fridrich and M. Goljan, "Practical steganalysis of digital images—State of the art," in Proc. SPIE Security Watermarking Multimedia Contents, vol. 4675, pp. 1--13, 2002.

[16] J. Mielikainen, "LSB Matching Revisited", IEEE Signal Processing Letters. 13 (5) (2006) 285–287.

[17] W. Luo, F. Huang, J. Huang, "Edge adaptive image steganography based on lsb matching revisited", Information Forensics and Security, IEEE Transactions 5 (2) (2010) 201–214.

[18] Tan, Shunquan, and Bin Li. "Targeted steganalysis of edge adaptive image steganography based on LSB matching revisited using B-Spline Fitting." Signal Processing Letters, IEEE 19, no. 6 (2012): 336-339.

[19] Ghazanfari, Kazem, and Reza Safabakhsh. "Adaptive method for hiding data in images." Journal of Electronic Imaging 21, no. 1 (2012): 013022-1.

[20] Hung-Min Sun, Yao-Hsin Chen, and King-Hang Wang, "An Image Data Hiding Scheme being Perfectly Imperceptible to Histogram Attacks," Image and Vision Computing New Zealand IVCNZ 2006, pp. 27--29, Nov. 2006.

[21] E. Franz, "Steganography preserving statistical properties" 5th International Workshop on Information Hiding, vol. 2578, pp. 278--294, , The Netherlands, Oct. 2002.

[22] A.R.S. Marçal, P.R. Pereira, A steganographic method for digital images robust to RS steganalysis, in: International Conference on Image Analysis and Recognition, Toronto, Canada, Lecture Notes in Computer Science, vol. 3656, 2005, pp. 1192–1199.

[23] Lou, Der-Chyuan, and Chen-Hao Hu. "LSB steganographic method based on reversible histogram transformation function for resisting statistical steganalysis." Information Sciences 188 (2012): 346-358.

[24] H.-T. Wu, J.-L. Dugelay, and Y.-M. Cheung, "A data mapping method for steganography and its

application to images" 10th International Workshop on Information Hiding ,vol. 5284, pp. 236--250, USA, May 2008.

[25] Provos, Niels. "Defending against statistical steganalysis." In 10th USENIX security symposium, vol. 10, pp. 323-336. 2001.

[26] Ghazanfari, Kazem, Shahrokh Ghaemmaghami, and Saeed R. Khosravi. "LSB++: An improvement to LSB+ steganography." In TENCON 2011-2011 IEEE Region 10 Conference, pp. 364-368. IEEE, 2011.

[27] Abolghasemi M, Aghaeinia H, Faez K. "Detection Of LSB±1 steganography based on Co-Occurrence Matrix and bit plane clipping", Journal of Electronic Imaging, 19 (2010) 013014.

[28] Xuan G, Shi Y Q, Huang C, Zhu D, Fu X, Chai P, Gao J. "Steganalysis using high-dimensional features derived from co-occurrence matrix and class-wise non-principal components analysis", IWDW (2006), LNCS 4283, pp. 49-60

[29] Sun Z, Hui M, Guan C. "Steganalysis Based on Co-occurrence Matrix of Differential Image", iih-msp, International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 3(2008) 1097-1100.

[30] Image Data base downloaded from website: "http://www.staff.lboro.ac.uk/~cogs/datasets/UCID/ucid.html"