Sādhanā (2018) 43:68
https://doi.org/10.1007/s12046-018-0848-4

New LSB-based colour image steganography method to enhance the efficiency in payload capacity, security and integrity check

MUSTAFA CEM KASAPBAŞI* and WISAM ELMASRY

Department of Computer Engineering, Istanbul Commerce University, Istanbul, Turkey
e-mail: mckasapbasi@ticaret.edu.tr; wisam.elmasry@istanbulticaret.edu.tr

MS received 16 February 2017; revised 17 October 2017; accepted 22 January 2018; published online 27 April 2018

Abstract. Steganography is the technique for hiding information within a carrier file so that it is imperceptible for unauthorized parties. In this study, it is intended to combine many techniques to gather a new method for colour image steganography to obtain enhanced efficiency, attain increased payload capacity, posses integrity check and security with cryptography at the same time. Proposed work supports many different formats as payload. In the proposed method, the codeword is firstly formed with secret data and its CRC-32 checksum, then the codeword is compressed by Gzip just before encrypting it by AES, and it is finally added to encrypted header information for further process and then embedded into the cover image. Embedding the encrypted data and header information process utilizes Fisher-Yates Shuffle algorithm for selecting next pixel location. To hide one byte, different LSB (least significant bits) of all colour channels of the selected pixel is exploited. In order to evaluate the proposed method, comparative performance tests are carried out against different spatial image steganographic techniques using some of the well-known image quality metrics. For security analysis, histogram, enhanced LSB and Chi-square analyses are carried out. The results indicate that with the proposed method has an improved payload capacity, security and integrity check for common problems of simple LSB method. Moreover, it has been shown that the proposed method increases the visual quality of the stego image when compared to other studied methods, and makes the secret message difficult to be discovered.

Keywords. Three bit LSB; image steganography; pseudo-random encoding; AES encryption; Gzip compression; CRC-32 Checksum.

1. Introduction

Secretly communicating with other parties has always been one of the well-known problems not only in this century, but also in ancient times. The aim of steganography is to hide the communication content in a medium, so that existence of hidden message can be concealed. Many surveys are published to indicate the-state-of-the-art of image steganography and its methods [1–3]. Mainly, technical steganography can be categorized into three areas according to the domain they are working; namely, spatial domain, temporal domain and frequency domain. Frequency and temporal steganography are generally used for processing audio signals, as carrier or message. This study can be regarded in spatial domain since it deals with LSB (least significant bits) of the cover image's pixels to hide secret data. A taxonomy is offered for smart phone steganography methods [3] which are categorized according to the targets; namely, Object methods (Image, QR, Audio, Video Text, etc.), Platform methods (SMS, MMS, Voice, Web/HTTP, Multimedia, etc.) and communication methods (Operating system and hardware).

Objectives of image steganography can be listed as imperceptibility, capacity and robustness. Imperceptibility is referred to as the resistance to both human visual system and statistical analyses and can be assessed with peak signal noise ratio (PSNR). Capacity is related to the amount of hidden data that can be embedded in the cover image. Robustness refers to the ability to recover hidden message despite processing the stego image such as cropping, scaling and filtering, etc. [4]. Moreover, security and integrity check can be added to these objectives. Security adds confidentiality dimension, while integrity check adds an insurance for transmission errors.

The rest of the paper is organized in the following order. The literature review about spatial domain image steganography is given in section 2. In section 3 and its sub-sections, the design of the proposed method is presented. In section 4, the proposed method's algorithms are given in discrete steps. Section 5 is dedicated to the performance analysis with comparison of other spatial image steganography techniques. Section 6 consists of security analysis including histogram, enhanced LSB and chi-square

*For correspondence

analyses. Section 7 is added for further discussion in regards to the compliance of the proposed method with the image steganography objectives. Finally, the conclusion is presented at the end of this paper.

2. Literature review

This section is intended to give a brief literature review about spatial domain image steganography since the proposed method is based on LSB steganography. Numerous image steganography applications based on LSB are introduced, some of most recent ones are listed in [5]. LSB steganography relies on the fact that replacing one or more of the last 1-4 bits of cover image's pixels is not perceptible by human visual system, but some statistical tests could detect that they are replaced in appropriate locations [6]. Many methodologies are proposed to conform fundamental requirements, however fundamentals of LSB steganography are detailed in [7].

One of the methods offers three replacement candidates and the one that has the closest value of the source pixel, called optimal pixel, is used for replacement [7]. A more recent LSB technique offers a method, called bit inversion, to further improve the PSNR (peak signal noise ratio) [5]. In this inversion technique, certain LSBs of the cover image's pixels are changed if they match with a particular pattern. The Pixel Value Differencing method (PVD) has inspired steganography researchers after it was introduced in [8, 9]. In this method, cover image is partitioned in non-overlapping blocks using difference values which are calculated for each two consecutive pixel values. Then, these values are used for replacing the payload. Different areas of the cover image have different payload capacity, so it is possible to hide more payloads around edges with this method. Applying randomization concept to LSB method is an LSB improved method, which works on the basis of the theory that the reaction of human eyes to Red, Blue and Green is different [10].

Kukapalli *et al* [11] have proposed an enhanced Pixel Indicator Method (PIM) by comparing three MSB bits at each pixel to embed data inside three LSB bits of that pixel. They also used Blowfish algorithm to convert message to cipher text. Dighe and Kapale have proposed random insertion using data parity steganography technique, in which secret data bits are embedded randomly by selected components of pixel [12]. Bashardoost *et al* have proposed in 2013 [13] an enhanced LSB image steganography method by using Knight Tour algorithm, Vigenere encryption and LZW compression. Although the proposed method in [13] increases both the payload capacity and quality of the stego image, it still suffers from problems in security and the lack of integrity check. Dadgostar and Afsari used interval-valued intuitionistic fuzzy edge detection in combination with the modified LSB substitution method, to obtain image quality and capacity increase [14].

In order to take some precaution against stego analysis, some guidelines are summarized in [15]. These are: embedding less information as much as possible, not to use cover images with computer art as much as possible, low number of colours and images with unique semantic content (such as fonts). Due to the fact that the quantization process of JPEG format reveals very small changes, such image formats for selection of cover image should be avoided.

3. Proposed method and design

The proposed image steganography method is composed of embedding phase and extraction phase. In the embedding phase which takes place on the sender side, the secret data is compressed and encoded with the proposed algorithm, and then resultant stream is embeded into the cover image. On the receiving side, the extraction phase takes place in order to comprehend the secret data within the stego image. This section introduces every necessary terms and concepts in the design of our new method. Figure 1 depicts the proposed method's framework and process flow diagram. Their details are presented in the following sub-sections.

3.1 *Data integrity*

One of the objectives of image steganography was the robustness against the manipulation of the image like compression, resizing, cropping, etc. When any of these manipulation is performed, there is a risk for losing the secret message. Therefore, a mechanism that ensure the data integrity with optimum payload cost is added, so that the receiver can realize if a transmission error or a manipulation has occurred. In this study, a well-known Cyclic Redundancy Check (CRC) error code is introduced to ensure data integrity, as it is commonly used to detect accidental changes in the row data which can happen in the storage devices and digital networks. CRC is light weight, easy to analyze mathematically and can provide fast and acceptable assurance for the integrity of the message [16, 17].

In the implementation of CRC, the sender calculates a 32-bit length CRC-32 checksum for the whole secret data block and appends it to the secret data block to form the codeword. This codeword length is equal to the sum of the length of secret data block plus 32 bits (4 bytes) of the CRC-32 checksum. When a codeword is received, the last 4 bytes are separated to obtain the received CRC-32 checksum. A new CRC-32 checksum is also calculated for the remaining bytes of the codeword, and then they are compared with each other to both check the integrity and accept if there is a match. Otherwise, the message is rejected and regarded as tampered or modified.

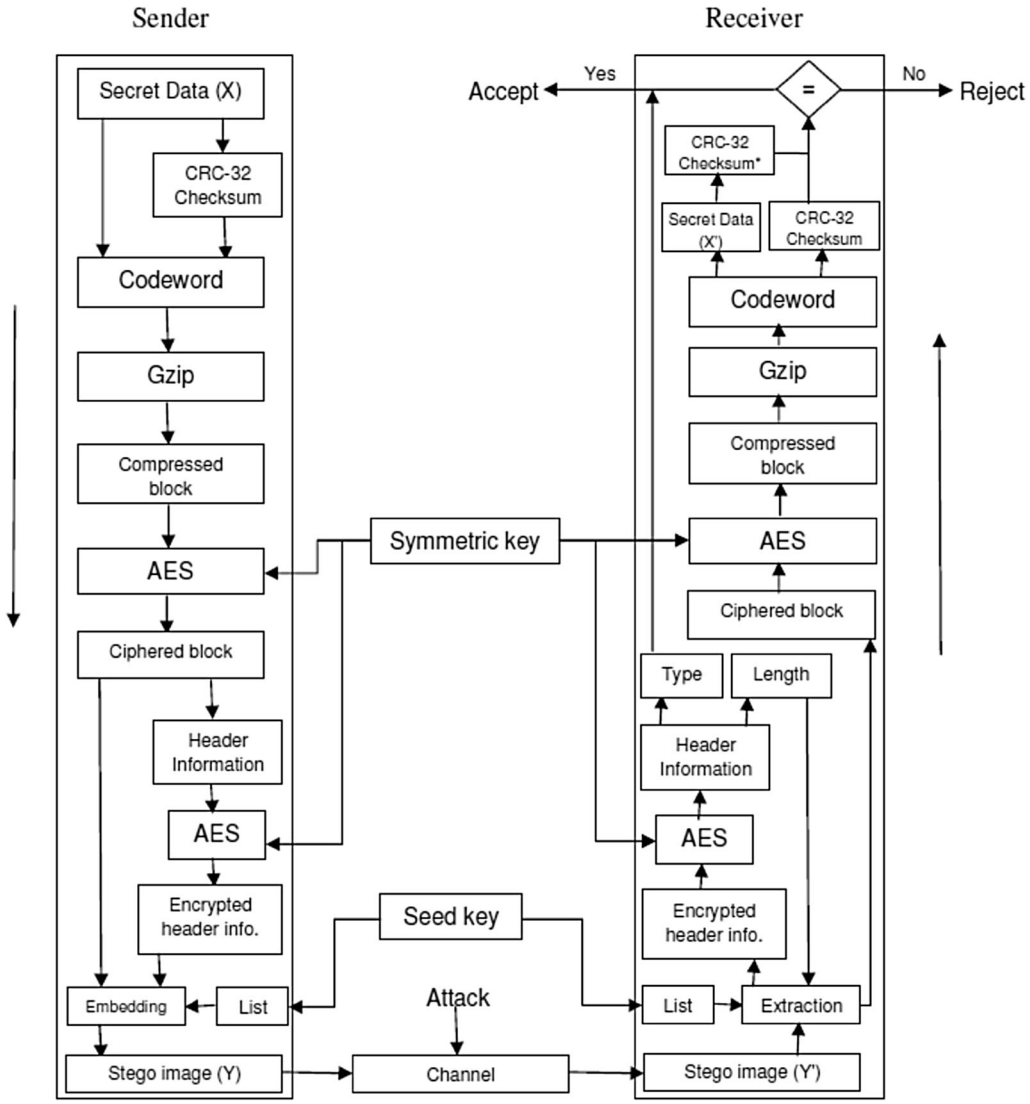


Figure 1. Proposed framework.

3.2 *Data compression*

The aim of integrating data compression to the proposed method is to increase the amount of payload that can be embedded in the cover image, since steganography requires sufficient amount of capacity for hidden communication unlike watermarking. Shortening the message size increases the payload capacity and also decreases the probability of discovering the existence of the message. Amongst many compression methods, Gzip (GNU zip) is chosen because not only it offers an acceptable capacity for lossless data compression and decompression, but also it is patent free and relatively easy to implement [18]. Compression is advised to be administered before the data encryption, as in the proposed method. Since the entropy of the data will increase after encryption, low data compression capacity will result.

The data compression procedure is very simple; the sender compresses the codeword, which is the combination of the secret data block and its CRC-32 checksum. On the other side, the receiver decompresses the received compressed data block and regenerates the original codeword.

3.3 *Data encryption*

In order to not get attention of an eavesdropper, hidden content needed to be unnoticeable both statistically and perceptually. For the sake of increasing data security, AES (Advanced Encryption Standard) encryption algorithm is implemented in the proposed method, just before embedding the message in the cover image as depicted in figure 1. AES is chosen, because it uses symmetric encryption, is versatile with many operation modes, is a block cipher (but

can work as a stream cipher as well), and is more secure than similar algorithms [19]. AES can operate with key/block length of 128, 192 and 256 bits long and their all possible combinations [20].

In the proposed method, a block size of 128 bits with a 128-bit-key is used. At the start of every session, the sender randomly generates the symmetric key and shares it with the receiver through one of symmetric key distribution methods. Furthermore, to ensure the production of the cipher text, which has the same length with the plain text length, we have used CTS operation mode of AES. CTS stands for Cipher Text Stealing mode, that handles any length of plain text and produces cipher text whose length matches the plain text length. The data encoding procedure is very plane; the sender encrypts the compressed data block using the randomly generated key and generates the ciphered data block. On the other side, the receiver decrypts the received ciphered data block using the same shared key, and then regenerates the original compressed data block.

3.4 *Header information*

Unless the receiver in the digital image steganography knows the precise length, the type and format of the embedded secret data will not be able to extract the embedded secret data properly. In order to overcome this problem, a new header information system is designed and implemented. This header information will enable the receiver to retrieve the embedded secret data properly.

In the proposed method, the sender is responsible for generating a 6-bytes length of header information from the ciphered data block. Figure 2 shows the construction of the desired header information block. The first two bytes of the header information are used to indicate the type of the original secret data. The secret data could be text, image file, multimedia, executable or any data file. Table 1 shows only a few examples of the 2-bytes length characters and their corresponding secret data type meaning. The last four bytes of the header information are reserved to specify the length of the ciphered data block in bytes. Four-bytes length number will be able to store the data length to Gigabytes which is big enough for every image steganography application. The sender will generate the header information by concatenating type of secret data and length in bytes. In order to avoid any information leak, this header information is also encrypted with the same generated key using AES with CTS operation mode. The resulting encrypted header information will be embedded into the cover image.

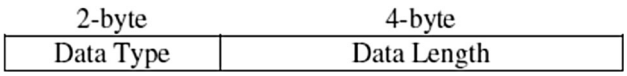


Figure 2. The header information (6-bytes).

Table 1. Secret data types and corresponding codes.

Code	Type
TT	Plain Text
IJ	JPEG Image File
IB	BMP Image File
IP	PNG Image File
IT	TIFF Image File
IG	GIF Image File
FT	Text File
FW	Word File
FP	PDF File
FA	Audio File
FV	Video File
FX	Executable File
XL	Excel File

On other side, the receiver will extract the encrypted header information, then decrypts it using AES with shared key in order to regenerate the original header information (type and length). After that, the receiver will isolate the last four bytes of the received header information and reads the length of the ciphered data block in bytes. The receiver will use this length to extract the whole ciphered data block from the stego image properly. Finally, the receiver will read the first two bytes of the received header information and store the type of the secret data. The receiver will use this type later to reconstruct the secret data to its original type.

3.5 *Pixel selection*

The pixel selection is one of the most important part of the image steganography. Its responsibility is to select a candidate pixel in the cover image in a specific order and embed the portion of the secret data in that pixel value. There are many techniques implemented for pixel selection. Namely, pseudo-random selection, optimal pixel adjustment [7], chaotic steganography [21], edge detection selection [14, 22], genetic steganography [23], etc. Pseudo-random pixel selection is the most common technique. Since even an attacker differentiate a stego image, it is expected that it will be hard to recover the embedding order or pattern of the secret data.

In the proposed method, a new pseudo-random pixel selection technique based on the Fisher-Yates Shuffle algorithm is implemented. The Fisher-Yates shuffle algorithm is attributed as an efficient and correct way of sorting arrays, as described by Donald Knuth and implemented in 1964 by Durstenfeld. It has an accurate, versatile and useful shuffling routine which randomizes array's element order. The advantage of the Fisher-Yates Shuffle algorithm is that, it produces an unbiased permutation [24].

The proposed new technique determines the dimensions of the cover image, multiplies the dimensions together to

provide the number of pixels available, and then uses the Fisher-Yates Shuffle algorithm to randomly permutate a list that includes values from 0 to (the number of pixels available-1) in a predictable and repeatable way by using the same random seed key value. This ensures that we do not overwrite secret data values in the cover image, and we can recover the secret data properly during the extraction phase. The advantages of our pseudo-random pixel selection technique over the others PRNG techniques are that, it is faster because the pixel locations are pre-computed, and is more secure because the secret data is embedded randomly across the entire image, as well as is almost unknown for the unintended receivers.

Firstly, the sender randomly generates a 32-bit-length seed key, and then uses it with the above described technique to obtain a randomly ordered list. Every time the sender picks up a number from the list consecutively, this number value has to be mapped to the pixel location of the cover image. The mapping procedure is very simple and convenient. The image is considered as a H x W matrix where H and W is the height and the width of that image, respectively. The pixels in the image have a specific order, which begins from the upper-left-corner pixel and continues from Top to Bottom and Left to Right manner, as shown in figure 3(a). Depending on this pixels' order, every pixel in the image has its own width and height coordinates. The upper-left-corner pixel has both width and height coordinates equal to zero. The sender computes these coordinates by using formula 1 and 2 for width and height coordinates, respectively, where i is the picked up number from the ordered list, and H is the height of the cover image. Then, the sender uses these coordinates to access the pixel location and embeds the secret data values in the cover image. Figure 3(b) shows an example of how to map pixel 2 in 3 × 3 image to its width and height coordinates.

$$X = i/H \tag{1}$$

$$Y = i \bmod H \tag{2}$$

On the other side, the receiver uses the same shared seed key with the above described technique to obtain a randomly ordered list. Every time the receiver picks up a number from the list consecutively, this number value has to be mapped to the pixel location of the stego image, using the same mapping procedure described in the sender side. Finally, the receiver accesses the pixel location and extracts the secret data values from the stego image.

3.6 *Data embedding*

LSB is considered to be the most common technique in the spatial domain image steganography, because rather than its simplicity, these LSB bits (specially 4-LSB) have lower amount of information than the 4-MSB. Figure 4 shows the percentage of the information that stored in each bit of one data byte. Regarding 1-byte of data, the 3-LSB bits hold less than 3% of the whole information that is stored in that byte. So changing the values of 3-LSB bits of image's data will make the image's alteration not perceptible for any human eyes, because the slight difference of colours.

Each pixel in colour image is specified by three values, one each for red, blue and green colour components. In the proposed method, we only deal with colour images which have at least a colour depth of 24-bits at each pixel. We embed 8 bits per pixel (8 bpp). This high embedding rate will lead us to increase the payload capacity within the colour image without sacrificing the imperceptibility. As shown in figure 5, one byte of the secret data is evenly distributed among the pixel's three-colour-components: red, green and blue. Regarding one byte of the secret data to

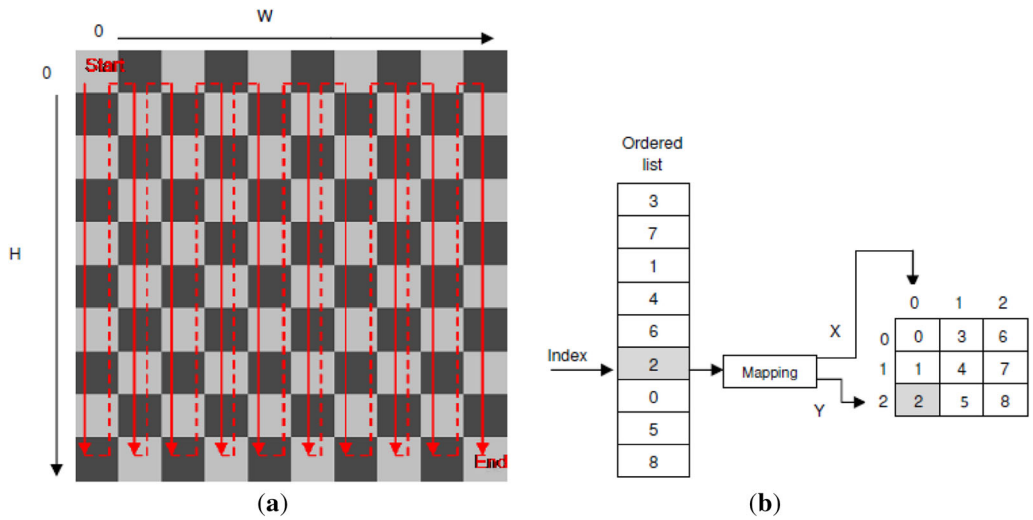


Figure 3. (**a**) Pixels order in the image. (**b**) Mapping pixel 2 to its coordinates in 3 × 3 image.

8	7	6	5	4	3	2	1
50	24	13	6.8	3.5	1.7	0.7	0.3
MSB							LSB

Figure 4. The percentage of information in each bit of one byte of data.

be embedded in the selected pixel, the 1st, 2nd and 3rd bits of that byte are embedded into the 3-LSB bits of the red component's byte. Then the 4th, 5th and 6th bits of that byte are embedded into the 3-LSB bits of the green component's byte. Finally, the 7th and 8th bits of that byte are embedded into the 2-LSB bits of the blue component's byte. This process repeats itself until all the secret data bytes are embedded successfully into the selected pixels.

4. The proposed algorithm

In this section, the detailed algorithms of both Embedding phase and Extraction phase will be step by step explained and presented.

4.1 *Embedding phase*

The embedding process is as follows:

Inputs: Secret data and cover image (BMP, PNG, TIFF).
Output: Stego image.
Procedure:
Step 1:

a. Convert secret data to bytes-array.
b. Compute CRC-32 checksum of the secret data bytes-array.
c. Concatenate the 4-bytes of the computed CRC-32 checksum with the secret data bytes-array together in one codeword data block.

Step 2: Compress the codeword data block using Gzip compression method to the compressed data block.
Step 3:

a. Generate randomly a 128-bit AES symmetric key.
b. Encrypt the compressed data block using AES cryptography algorithm to the ciphered data block.

Step 4:

a. Create the 6-bytes header information by storing the type of the original secret data into the first two bytes, and the length of the ciphered data block in bytes into the last four bytes.
b. Protect the header information by encrypting it using AES and the same 128-bit symmetric key used in step 3 to the 6-bytes of encrypted header information.

Step 5:

a. Determine W = width and H = height of the cover image.
b. Let Total = W*H.
c. Generate a list of integers with size equals to Total including numbers from 0 to (Toal − 1).
d. Generate a randomly 32-bit seed key.
e. Randomize the order of the elements of the list by using the Fisher-Yates Shuffle algorithm and the seed key.
f. Save the new randomly ordered list and put the index to the first element of it (index = 0).

Step 6:

a. Let j = 0.
b. Let i = list[index].
c. Map i to pixel's coordinates by computing X and Y using formula 1 and 2, respectively.
d. Access the selected pixel i in the cover image.
e. Embed the byte b = encrypted header information[j] into the channels of the selected pixel i.
f. let j = j+1 and index = index+1.
g. Repeat sub-steps 6(b) to 6(f) until j = 6.

Step 7:

a. Let length = length of the ciphered data block in bytes.
b. Let k = 0.
c. Let i = list[index].
d. Map i to pixel's coordinates by computing X and Y using formula 1 and 2 respectively.
e. Access the selected pixel i in the cover image.
f. Embed the byte b = ciphered data block[k] into the channels of the selected pixel i.
g. let k = k+1 and index = index+1.
h. Repeat sub-steps 7(c) to 7(g) until k = length.

Step 8: Create the stego image and transmit it to the receiver.

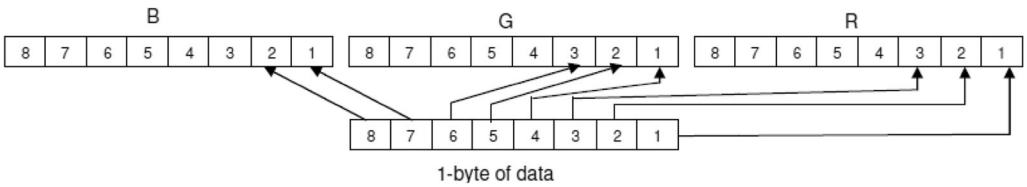
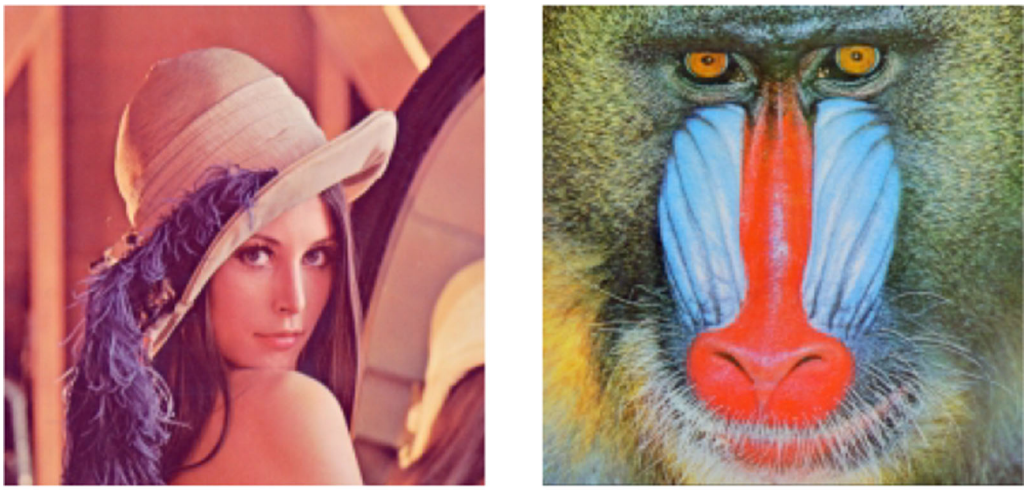


Figure 5. The process of LSB substitution in colour image.



(**a**) (**b**)

Figure 6. 512 × 512 colour cover images: (**a**) Lena, (**b**) Baboon.

4.2 *Extraction phase*

The extraction process is as follows:

Input: Stego image (BMP, PNG, TIFF).

Output: The original secret data if the receiver accepts the received data. Else the receiver rejects it.

Procedure: The extraction process is the reverse of the embedding process as shown in figure 1.

5. Performance analysis

The proposed method is compared with the Sequential-LSB and PRNG-LSB methods. Sequential-LSB is the simple LSB image steganography method, where secret data is embedding sequentially. In contrast, PRNG-LSB is the LSB image steganography method, where secret data is randomly embedding using simple pseudo-random number generator. In Sequential-LSB, PRNG-LSB and the proposed method, we embedded secret data with 8 bpp (bit per pixel) embedding rate, as described in sub-section 3.6. There is no limitation to use our proposed method except that the cover image must be 24-bits colour image at least. This means that, regardless of the type of the selected colour cover image (conventional, unconventional, synthetic, etc.) our proposed method can be applied on that image smoothly.

Sequential-LSB, PRNG-LSB and the proposed method are implemented by using C#.NET framework. Images of 512 × 512 Lena and Baboon, which are illustrated in figures 6(a), (b), respectively, are used in colour mode for testing. We selected Baboon and Lena images, because they are widely used in the literature and it would be easy for readers to compare the results. The secret data which is used in the implementation is 100% arbitrary text with different size from 1 to 256 Kbytes.

5.1 *Image quality metrics*

The image quality metrics are used to determine the quality of stego image and similarity with cover image. Eight of the most well-known image quality metrics are used: Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), Normalized Cross-Correlation (NK), Average Difference (AD), Structural Content (SC), Maximum Difference (MD), Laplacian Mean Square Error (LMSE) and Normalized Absolute Error (NAE). Table 2 shows each of the image quality metrics and their corresponding formula, where M is the width of the image, N is the height of the image, $x_{j,k}$ is the j^{th} k^{th} pixel in the stego image and $x'_{j,k}$ is the j^{th} k^{th} pixel in the cover image [25].

The results of Sequential-LSB, PRNG-LSB and the proposed method are presented in tables 3, 4 and 5, respectively. Different size of payloads are embedded in sample of 512 × 512 Lena image. The greater the value of PSNR, the lower degree of distortion presents for stego image. The results indicate that the proposed method has higher PSNR values in all test cases. It means that in all test cases, the proposed method gives lower MSE values since it decreases the number of pixels that are altered.

However, increasing the payload amount causes a significant fall in PSNR value. Furthermore, the proposed method increases the possible amount of secret data that could be embedded into same cover image, because it uses Gzip compression algorithm to decrease the size of payload before embedding it. Regarding all other metric values, MSE, NK, AD, SC, LMSE and NAE indicate that the proposed method has performed better than others. Figures 7 and 8 show MSE and PSNR values, respectively, for each of the tested cases.

Table 2. The used image quality metrics.

Metrics	Calculation Formula
Mean Square Error	$\mathrm{MSE} = \frac{1}{\mathrm{MN}}\sum_{j=1}^{M}\sum_{k=1}^{N}\left(x_{j,k} - x'_{j,k}\right)^2$
Peak Signal to Noise Ratio	$\mathrm{PSNR} = 10\log\frac{(2^n-1)^2}{\mathrm{MSE}} = 10\log\frac{255^2}{\mathrm{MSE}}$
Normalized Cross-Correlation	$\mathrm{NK} = \sum_{j=1}^{M}\sum_{k=1}^{N} x_{j,k}.x'_{j,k} / \sum_{j=1}^{M}\sum_{k=1}^{N} x_{j,k}^2$
Average Difference	$\mathrm{AD} = \sum_{j=1}^{M}\sum_{k=1}^{N}\left(x_{j,k} - x'_{j,k}\right)/\mathrm{MN}$
Structural Content	$\mathrm{SC} = \sum_{j=1}^{M}\sum_{k=1}^{N} x_{j,k}^2 / \sum_{j=1}^{M}\sum_{k=1}^{N} x'^2_{j,k}$
Maximum Difference	$\mathrm{MD} = \mathrm{Max}\left(\left\|x_{j,k} - x\prime_{j,k}\right\|\right)$
Laplacian Mean Square Error	$\mathrm{LMSE} = \sum_{j=1}^{M}\sum_{k=1}^{N}\left[O\left(x_{j,k}\right) - O\left(x'_{j,k}\right)\right] / \sum_{j=1}^{M}\sum_{k=1}^{N}\left[O\left(x_{j,k}\right)\right]^2.$ $O\left(x_{j,k}\right) = x_{j+1,k} + x_{j-1,k} + x_{j,k+1} + x_{j,k-1} - 4x_{j,k}$
Normalized Absolute Error	$\mathrm{NAE} = \sum_{j=1}^{M}\sum_{k=1}^{N}\left\|x_{j,k} - x'_{j,k}\right\| / \sum_{j=1}^{M}\sum_{k=1}^{N}\left\|x_{j,k}\right\|$

Table 3. Results of sequential-LSB.

Payload Size (Kbytes)	Embedded Data (Bytes)	MSE	PSNR (dB)	NK	AD	SC	MD	LMSE ($\times 10^{-6}$)	NAE
1	1024	0.0311	63.2094	1	− 0.0006	1	7	0.588	0.0001
2	2048	0.062	60.206	1	− 0.0014	1	7	0.6453	0.0001
4	4096	0.1259	57.1313	1	− 0.0029	1	7	0.7134	0.0003
8	8192	0.2534	54.092	1	− 0.0021	1	7	0.6633	0.0005
16	16384	0.501	51.1321	1.0001	− 0.0104	0.9999	7	0.6166	0.0011
32	32768	1.0004	48.129	1.0001	− 0.022	0.9998	7	1.1311	0.0021
64	65536	2.0011	45.1182	1.0002	− 0.0426	0.9995	7	0.5073	0.0043
128	131072	3.996	42.1145	1.0005	− 0.0846	0.9988	7	1.1311	0.0086
256	–	–	–	–	–	–	–	–	–

Table 4. Results of PRNG-LSB.

Payload Size (Kbytes)	Embedded Data (Bytes)	MSE	PSNR (dB)	NK	AD	SC	MD	LMSE ($\times 10^{-6}$)	NAE
1	1024	0.0322	63.0579	1	− 0.0004	1	7	0.0269	0.0001
2	2048	0.0611	60.2721	1	− 0.0014	1	7	0.0179	0.0001
4	4096	0.1251	57.1588	1	− 0.0024	1	7	0.0574	0.0003
8	8192	0.252	54.1173	1	− 0.0045	0.9999	7	− 0.0556	0.0005
16	16384	0.5011	51.1314	1.0001	− 0.0098	0.9999	7	0.0896	0.0011
32	32768	1.0029	48.1184	1.0001	− 0.0226	0.9997	7	0.2259	0.0022
64	65536	1.9913	45.1393	1.0002	− 0.0436	0.9994	7	0.7511	0.0043
128	131072	3.9778	42.1343	1.0005	− 0.0852	0.9989	7	0.8515	0.0085
256	–	–	–	–	–	–	–	–	–

5.2 *Compression ratio*

To calculate maximum payload capacity of the proposed method, equation (3) can be formulated easily since 1 byte of data is hidden in one colour pixel. Therefore, maximum capacity is directly proportional to the cover image's multiplication of width and height minus 6 bytes, which are used for embedding the encrypted header information:

$$\text{Max Capacity} = (\text{Image width} * \text{Image height}) - 6\,(\text{Bytes}) \quad (3)$$

Table 5. Results of the proposed method.

Payload Size (Kbytes)	Embedded Data (Bytes)	MSE	PSNR (dB)	NK	AD	SC	MD	LMSE ($\times 10^{-6}$)	NAE
1	912	0.0267	63.8619	1	0.0002	1	7	0.0054	0.0001
2	1744	0.054	60.8053	1	0	1	7	− 0.0287	0.0001
4	3440	0.1034	57.985	1	0.0006	1	7	− 0.0484	0.0002
8	6848	0.2064	54.9833	1	0.0008	1	7	0.0197	0.0004
16	13648	0.411	51.9925	1	0.0007	1	7	− 0.0108	0.0009
32	27248	0.8169	49.0092	1	0.0009	1	7	0.1183	0.0018
64	54464	1.6335	45.9995	1	0.0011	0.9999	7	0.2044	0.0035
128	108864	3.2673	42.9889	1	0.0003	0.9999	7	0.0932	0.007
256	217664	6.4575	40.0834	1	− 0.0013	0.9997	7	0.0932	0.014

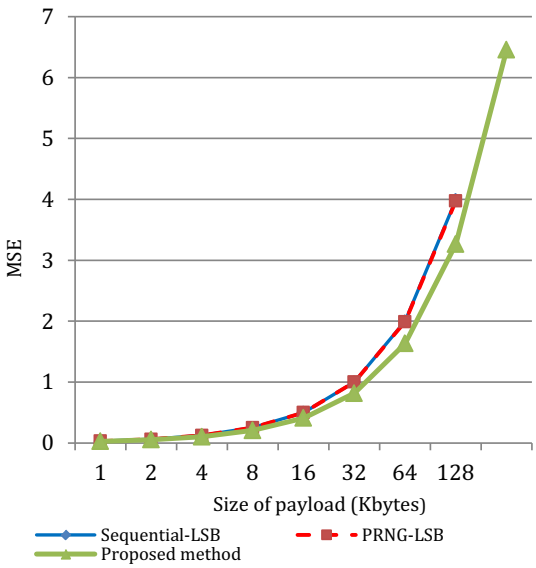


Figure 7. Size of payload (Kbytes) vs. MSE values.

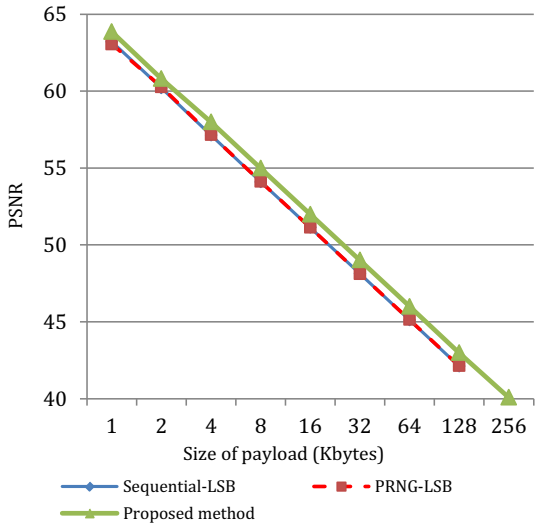


Figure 8. Size of payload (Kbytes) vs. PSNR values.

In addition to that, since Gzip algorithm is used to compress the data, expecting an increase in the maximum size of payload is reasonable. The compression ratio is known to be highly dependent on the entropy of the secret data; in other words, the redundancy in the secret message. The worst case scenario could occur when having highest entropy or no redundancy for secret data, which may result in a compression ratio of 1. As a result, it is not an easy task to calculate the maximum size of the payload, since it is relative to the type of secret message. In tests, uniformly distributed random text secret messages with different size are produced and used throughout tests. The growth in the rate of compression is investigated and illustrated in figure 9.

Table 6 indicates the maximum size of payload that can be embedded into colour cover image when all pixels of the cover image, for embedding the secret data and the encrypted header information, are considered. It is easy to notice that the maximum size of payload is increased when the data is compressed, as in the proposed method.

6. Security analysis

In this section, we will analyse the proposed method against three of famous statistical and visual attacks to ensure its immunity against these attacks, and have a more precise evaluation of our method in terms of security.

6.1 *Histogram analysis*

It is considered a statistical attack since the histogram of an image shows a graph of the number of pixels at each different intensity value found in that image. This attack allows human eye to distinguish the difference between the cover and stego images, if there is a message embedded in channels. For a 24-bit colour image, 256 different

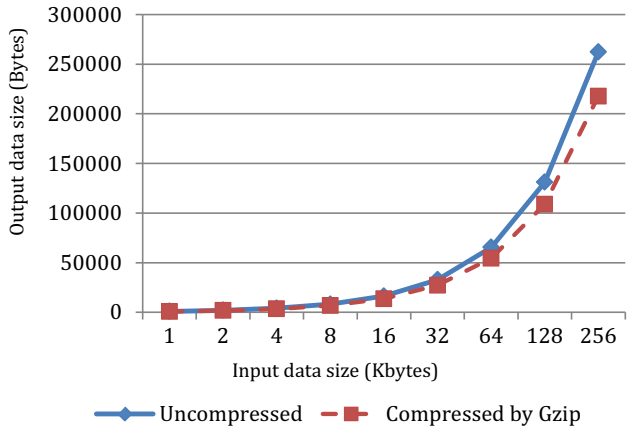


Figure 9. Input data (Kbytes) vs. Output data (bytes).

intensities for each of the 3 channels (red, green, blue) are possible. Therefore, a histogram for each channel can be drawn separately, or an average histogram of all channels can be produced. Table 7 presents the red-channel, green-channel, blue-channel and the average histograms of Sequential-LSB, PRNG-LSB, and the proposed method when we embedded 128 Kbytes of payload in the sample picture of Baboon by the size of 512 × 512.

From table 7 it can be noticed by human eye that the red, green and blue channels' histograms of the Sequential-LSB and PRNG-LSB methods are different from those of the original cover image. In contrast, the red, green and blue channels' histograms of the proposed method are almost the same as those of the original cover image due to data compression (which decreases the size of embedded data). However, the average histograms of Sequential-LSB, PRNG-LSB and the proposed method are still the same as the one of the original cover image.

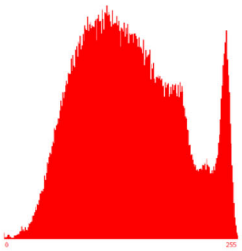
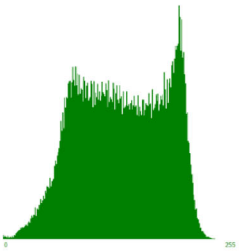
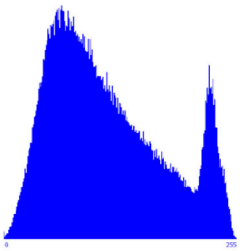
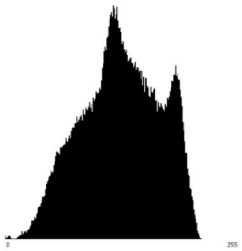
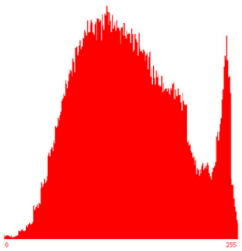
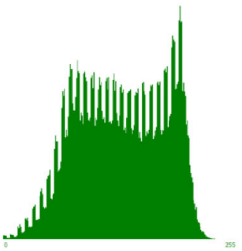
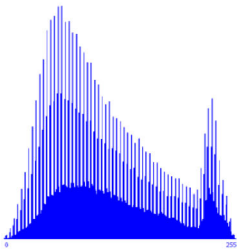
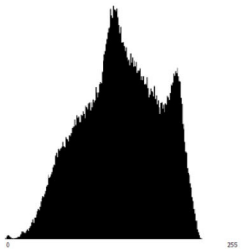
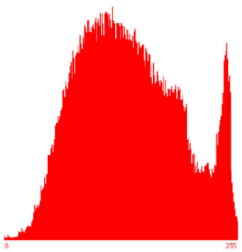
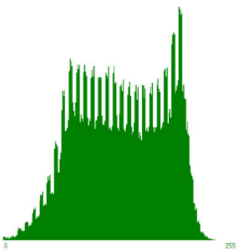
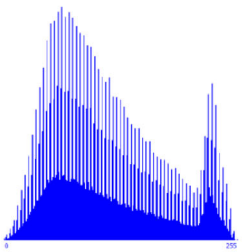
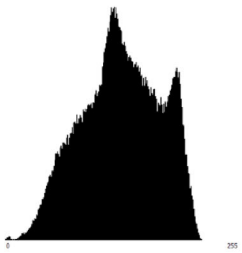
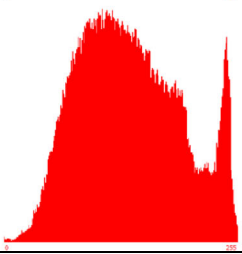
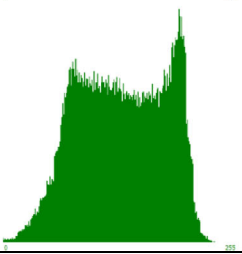
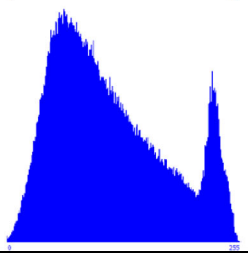
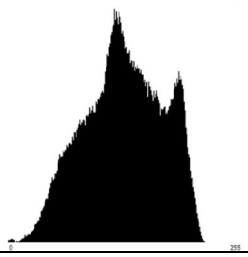
6.2 *Enhanced LSB analysis*

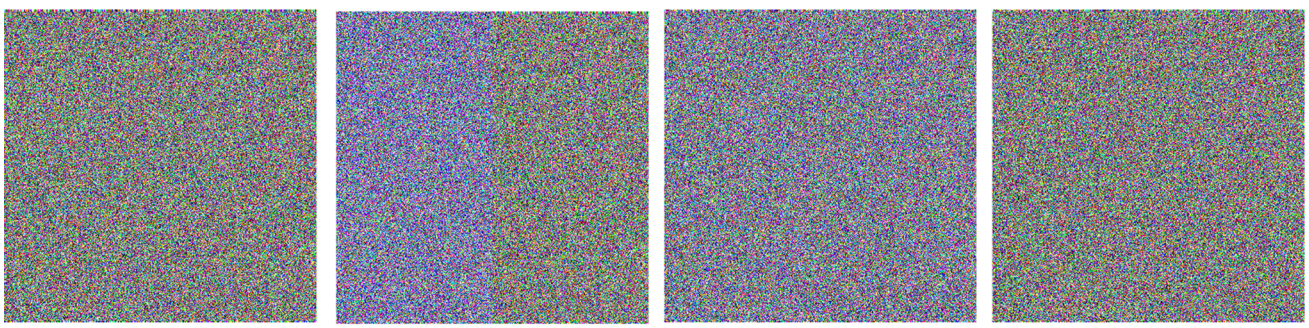
Since the image steganographic methods which are based on LSB only alters the least significant bits, these changes are not noticeable in regards to image quality in most cases. The fundamental philosophy of the Enhanced LSB attack, which is a visual analysis on a stego image, is to eliminate 7 high level bits of each channel of the pixels, and concentrate on the last LSB. Resulting channel's byte is going to be 0 or 1. Then, all 1s are converted to maximum value of 255 and all 0s are left as 0, which is a kind of enhancement basically. This analysis aims at emerging a visual pattern which can be checked by human eye. Figure 10 shows the results of Sequential-LSB, PRNG-LSB and the proposed method, when we embedded 128 Kbytes of payload in sample picture of Lena by the size of 512 × 512.

Table 6. Maximum payload sizes for different image dimensions.

Image dimension (pixels)	Sequential-LSB			PRNG-LSB			Proposed method		
	256 × 256	512 × 512	1024 × 1024	256 × 256	512 × 512	1024 × 1024	256 × 256	512 × 512	1024 × 1024
Maximum size of secret data (Bytes)	65530	262138	1048570	65530	262138	1048570	78848	315392	1262592

Table 7. Histograms of cover and stego images.

Cover/Stego	Red	Green	Blue	Average
Cover image				
Sequential-LSB				
PRNG-LSB				
Proposed method				



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

Figure 10. The results of: (**a**) Original cover image. (**b**) Sequential-LSB with 50% of hidden data. (**c**) PRNG-LSB with 50% of hidden data. (**d**) Proposed method with 50% of hidden data.

It can be seen from figure 10(a) that the cover image consists of no recognizable pattern, but some arbitrary pixels. However, after embedding the secret message using Sequential-LSB method, some artifacts become visible regarding the layer zero. When only 50% of the pixels in the cover image are used to embed the secret data, a vertical

strip pattern has appeared (figure 10(b)). In figure 10(c) some differences became also noticeable with bare eyes, when only 50% of hidden data is embedded using the PRNG-LSB method. Furthermore, when the proposed method is exploited, the LSB zero layer looks entirely innocent due to pseudo-random pixel selection technique. This technique is based on the Fisher-Yates Shuffle algorithm, which distributes the secret message randomly and efficiently to the entire stego image. The result of the proposed method is depicted in figure 10(d).

6.3 *Chi-square analysis*

One of the fundamental tests is the Chi-square statistical analysis, which aims to reveal whether the suspected image carries (or not) any hidden message [26]. This attack is based on the distribution probability of zeros and ones over the image. Figure 11 shows the results of Sequential-LSB, PRNG-LSB and the proposed method. We embedded 128 Kbytes (50% of total pixels) and 256 Kbytes (100% of total pixels) of payload in sample picture of Baboon by the size of 512 × 512.

Figures 11(a), (b) depict the result of the Chi-square analysis with 128 and 256 kilobytes of secret data embedded by Sequential-LSB method respectively. The existence of embedded secret data can be pinpointed with a sharp decrease to zero in the probability trend. Figures 11(c), (d) present the result of the Chi-square analysis with 128 and 256 kilobytes of secret data embedded by PRNG-LSB method respectively. The PRNG-LSB method gives better result than the Sequential-LSB method, and it is still easy to detect the existence of the secret data. Figures 11(e), (f) depict the result of the Chi-square analysis with 128 and 256 kilobytes of secret data embedded by the proposed method respectively. Even with 50% and 100% of hidden data, the Chi-square diagram almost fails to detect any embedded data for the test images. The results indicate that the proposed method is resistant to the Chi-square statistical analysis.

7. Proposed method vs. image steganography objectives

This section intents to evaluate the proposed method against the objectives of image steganography; namely: imperceptibility, capacity, robustness and security [27–30].

- Imperceptibility: As presented in sub-section 5.1 the proposed method produces a high quality stego image, and embedding the secret data does not distort the cover image to a visually unacceptable level. This is due to both effective and lossless data compression algorithm (Gzip) and the randomly embedding of the secret data into the cover image, using the pseudo-random pixel selection technique based on the Fisher-Yates Shuffle algorithm.
- Capacity: As presented in sub-section 5.2 the proposed method proved its ability to increase the payload

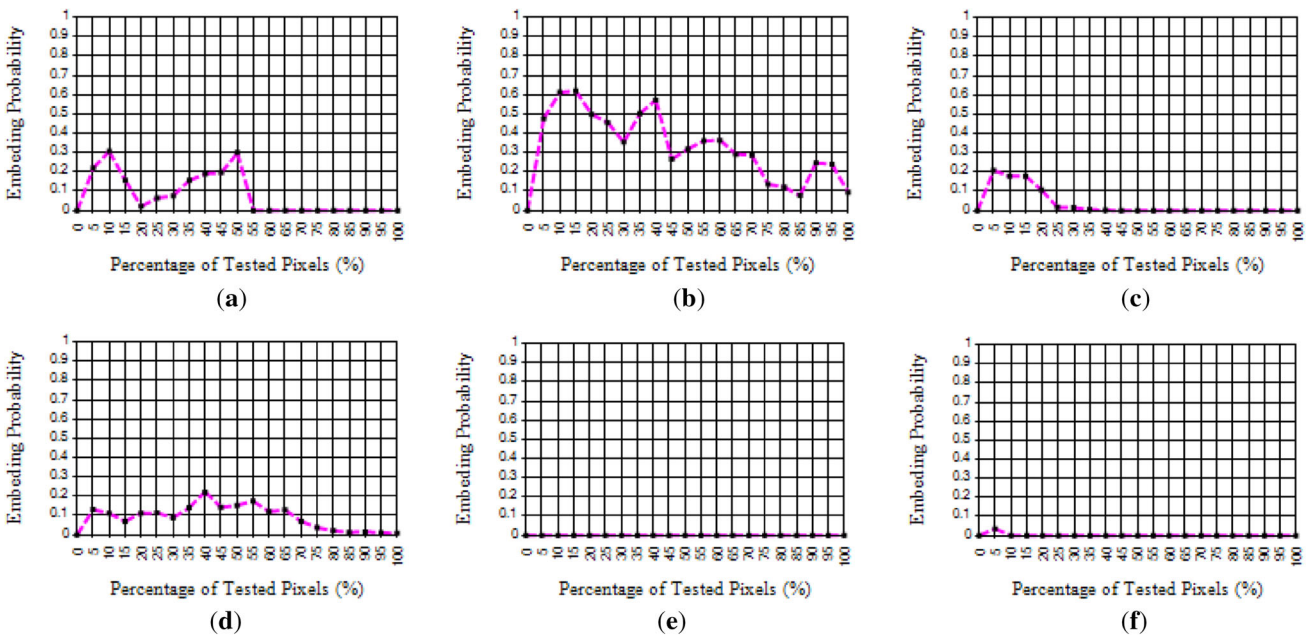


Figure 11. The results of: (**a**) Sequential-LSB with 50% of hidden data, (**b**) sequential-LSB with 100% of hidden data, (**c**) PRNG-LSB with 50% of hidden data, (**d**) PRNG-LSB with 100% of hidden data, (**e**) proposed method with 50% of hidden data, (**f**) proposed method with 100% of hidden data.

capacity that can be hidden within the cover image. This is due to both Gzip compression algorithm and embedding 1-byte of secret data per pixel (8 bpp) without sacrificing the imperceptibility.

- Robustness: In terms of integrity, the proposed method enhances it and has the ability to detect either intentional or unintentional alterations to the stego image. In case the secret data is modified during transmission, the receiver is able to check the integrity using CRC-32 checksum and realize whether the secret data is fake or altered.
- Security: As shown in section 6, the proposed method has the ability to be immune against some of famous and well-known statistical and visual attacks. This is due to the cooperation of combined mechanisms; AES with 128-bit-length symmetric key, pseudo-random pixel selection technique based on the Fisher-Yates Shuffle algorithm (with 32-bit-length seed key), and Gzip compression algorithm.

8. Conclusion

In this paper, we provided a series of enhancements, and argued that the proposed method fixed the weakness of Simple LSB image steganography method. The proposed method combines six fundamental improvements, specifically: CRC-32 checksum, Gzip compression, AES encryption, Header information, Pseudo-random pixel selection technique based on the Fisher-Yates Shuffle algorithm and 8 bpp embedding algorithm.

The process starts with computing the CRC-32 checksum of the secret data and combining both of them together in one codeword. Next, in order to improve the payload capacity, the Gzip compression algorithm is used to reduce the size of the codeword. Afterward, the proposed method generates a 6-bytes-length header information and both the codeword and the header information are encrypted with AES using a shared 128 bits key. Finally, the generated bytes stream of the encrypted header information and the ciphered data block are embedded into the cover image in the positions defined by the proposed pseudo-random pixel selection technique based on the Fisher-Yates Shuffle algorithm with a shared 32 bits seed key. The proposed method uses an 8 bit-per-pixel embedding algorithm to increase the payload capacity within the cover image.

After seeing the performance and security assessments of the proposed method, one can say that the proposed method not only satisfies the necessary and sufficient objectives of the image steganography, but also introduces a new embedding methodology and integrity check combination successfully.

References

[1] Trivedi M C Sharma S and Yadav V K 2016 Analysis of several image steganography techniques in spatial domain: a survey. In*; Second International Conference on Information and Communication Technology for Competitive Strategies (ICTCS '16)*. ACM. Article 84

[2] Petitcolas F A P Anderson R J and Kuhn M G 1999 Information hiding-a survey. *Proc. IEEE.* 87(7): 1062–1078

[3] Mazurczyk W and Caviglione L 2015 Steganography in modern smartphones and mitigation techniques. *IEEE Commun. Surv. Tutor.* 17(1): 334–357

[4] Singla D and Juneja M 2014 An analysis of edge based image steganography techniques in spatial domain. In: *Recent Advances in Engineering and Computer Sciences (RAECS)* 1–5

[5] Akhtar N 2016 An LSB substitution with bit inversion steganography method. In: *Proceedings of 3rd International Conference on Advanced Computing, Networking and Informatics* 43: 515–521

[6] Chen Y, Han Z, Li S, Lu C and Yao X H 2010 An adaptive steganography algorithm based on block sensitivity vectors using HVS features.In: *3rd International Congress in Image and Signal Processing*. 1151–1155

[7] Chan C-K and Cheng L-M 2004 Hiding data in images by simple LSB substitution. *Pattern Recognit.* 37: 469–474

[8] Wu D-A and Tsai W-H 2003 A steganographic method for images by pixel-value differencing. *Pattern Recognit. Lett.* 24(9–10):1613–1626

[9] Wu H-C, Wu N I, Tsai C S and Hwang M S 2005 Image steganographic scheme based on pixel-value differencing and LSB replacement methods. IEE Proceedings Vision, *Image and Signal Processing* 152: 611–615

[10] Anand J V and Dharaneetharan G D 2011 New approach in steganography by integrating different LSB algorithms and applying randomization concept to enhance security. Proceedings of the 2011 In: *International Conference on Communication, Computing* 474–476

[11] Kukapalli V R, Rao T B and Reddy B S 2014 Image Steganograpghy by Enhanced Pixel Indicator Method Using Most Significant Bit (MSB) Compare. *International Journal of Computer Trends and Technology* 15(3): 97-101

[12] Dighe D and Gand Kapale N D 2013 Random Insertion Using Data Parity Steganography Technique. *Int. J. Eng. Sci. Innov Technol (IJESIT)* 2(2): 364–368

[13] Bashardoost M, Sulong G B and Gerami P 2013 Enhanced LSB image steganography method by using knight tour algorithm, vigenere encryption and LZW Compression. *Int. J. Comput. Sci. Iss. (IJCSI)* 10(2) 1: 221–227

[14] Dadgostar H A and Fsari F 2016 Image steganography based on interval-valued intuitionistic fuzzy edge detection and modified LSB. *Journal of Information Security and Applications (JISA).* 30: 94–104

[15] Fridrich J, Goljan M and Du R 2001 Detecting LSB steganography in colour and gray-scale images. *IEEE Trans. Multimed.* 8(4): 22–28

[16] Peterson W W and Brown D T 1961 Cyclic Codes for Error Detection. *Proc. IRE* 49 (1): 228–235

[17] Wikipedia Cyclic redundancy check. https://en.wikipedia.org/wiki/Cyclic_redundancy_check. Accessed on Jan 2018

[18] Gailly J-L and Adler M Gzip Home Page. http://www.gzip.org/. Accessed on Jan 2018

[19] Sahmoud S, Elmasry W and Abudalfa S 2013 Enhancement the security of AES against modern attacks by using Variable key block cipher. *Int. Arab J. e-Technol. (IAJeT)*. 3(1): 17–26

[20] Daemen J and Rijmen V 2001 Rijndael: the advanced encryption standard. *Dr. Dobb's J.* 26(03): 137–139

[21] Lou D-C and Sung C-H 2004 A steganographic scheme for secure communications based on the chaos and Euler Theorem. *IEEE Trans. Multimed*. 6(3): 501–509

[22] Luo W, Huang F and Huang J 2010 Edge adaptive image steganography based on LSB matching revisited. *IEEE Trans. Inf. Forensics Secur* 5(2): 201–214

[23] Wang S, Yang B and Niu X 2010 A secure steganography method based on genetic algorithm. *J. Inf. Hiding Multimed. Signal Process* 1(1): 28–35

[24] Fisher R A and Yates F 1948 *Statistical tables for biological, agricultural and medical research*. London: Oliver & Boyd, pp. 26–27

[25] Sasivarnan C, Jagan A, Kaur J, Jyoti D and Rao D S 2011 Image quality assessment techniques in spatial domain. *Int. J. Comput. Sci. Technol (IJCST)* 2(3): 177–184

[26] Stanley C A *Pairs of Values and the Chi-Squared Attack* https://orion.math.iastate.edu/dept/thesisarchive/MSCC/CStanleyMSSS05.pdf. Accessed on Jan 2018.

[27] Cheddad A, Condell J, Curran K and McKevitt P 2010 Digital image steganography: Survey and analysis of current methods. *Signal Process.* 90(3): 727–752

[28] Li B, He J, Huang J and Shi Y Q 2011 A Survey on Image Steganography and Steganalysis. *J. Inf. Hiding Multimed. Signal Process.* 2(2):142–172

[29] Jain M and Lenka S K 2016 A review of digital image steganography using LSB and LSB array. *Int. J. Appl. Eng. Res.* 11(3):1820–1824

[30] Al-Hawi T, Al-Qutayri M and Barada H 2003 A testbed for evaluating security and robustness of steganography techniques. In: *IEEE 46th Midwest Symposium in Circuits and Systems*. 3: 1583–1586