International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.10, No. 1 (2017), pp.35-48
http://dx.doi.org/10.14257/ijsip.2017.10.1.05

Variant of LSB Steganography Algorithm for Hiding Information in RGB Images

Ashis Kumar Mandal[1*] and M N M Kahar[2]

[1]Department of Computer Science and Engineering, Hajee Mohammad Danesh Science and Technology University, Dinajpur, Bangladesh
[2]Faculty of Computer Systems & Software Engineering, University Malaysia Pahang, Pahang, Malaysia
[1]ashis@hstu.ac.bd,[1]ashis.62@gmail.com,[2]mnizam@ump.edu.my*

Abstract

Steganography has been emerged as a prominent technique for communicating secret information in a proper multimedia career. In image steganography, least significant bit (LSB) substitution method is popular but vulnerable to numerous security attacks. Hence, it is always imperative to make the steganographic technique more robust and secure. This paper presents a new approach to hiding text messages within red-green-blue (RGB) images based on LSB technique. Unlike the standard LSB technique, where message bits are directly embedded in the LSB positions, our proposed algorithm encodes these message bits before embedding in the LSB positions. This embedding and encoding process are done on the basis of most significant bit (MSB) values of RGB and the concept of odd and even parities of those pixels respectively. The main goal of the presented method is to improve the robustness and the image quality of the stego-image. The performance of stego-image quality is measured by mean square of error (MSE), peak signal-to-noise ratio (PSNR), structural similarity measure (SSIM), bits ratio (BR), and histogram analysis. Experimental results and comparative studies have revealed that the proposed technique is able to conceal message effectively, with there being little difference between the cover image and the stego-image. Results also show that our proposed approach can embed reasonable amount of information.

Keywords*: Image steganography, LSB, MSB, odd and even parity, RGB image*

1. Introduction

Information hiding research domain has attracted many researchers in producing the reliable technique to ensure the protection of digital contents from numerous security attacks, especially in the online domain. Such techniques seen in the literature include cryptography, watermarking and steganography [1-2]. One of the popular approaches is the steganography, which is the art and science of hiding information by embedding messages within cover media such as text, image, sound or video files [3]. For example, in image steganography, the cover medium is the image whereby information is embedded by altering some nonessential components of that image.

Usually, in image steganography, there are two ways of steganographic algorithms, transform domain and spatial domain. Steganography in image transform domain includes performing a series of changes to the cover image before hiding the information. Discrete cosine transformation (DCT), fourier transformation (FT) and discrete wavelet transformation (DWT) are some techniques used for this purpose [4]. On the other hand, steganography in the image spatial domain generally uses a direct least significant bit (LSB) replacement technique — encoding at the level of the LSBs [5].

[*] Corresponding Author

ISSN: 2005-4254 IJSIP

LSB image steganography is a popular approach due to its better imperceptibility and payload capacity [6]. Despite its simplicity, traditional LSB is vulnerable as the probability of detecting the secret message is high. Therefore, a series of methods have been proposed based on LSB substitution technique. Most of the existing approaches attempt to select a region or bit streams of a cover image and embed a message which is usually encoded based on cryptography or other mathematical functions, with a secret key employed for decoding the message [7-8]. Although performances of these methods are better compared with LSB substitution method, communication of secret key is challenging as well as in terms of overall efficiency, frequently those methods are not fully exploited.

We propose a modified version of LSB that can increase imperceptibility and robustness of LSB image steganography. Rather than storing the message bits directly in LSB positions, each message bit is encoded and placed in one of the 3 channels of RGB pixel. The selection of a particular channel is done on the basis of MSB bits of that RGB pixel, whereas even and odd parity values of that selected channel biase the encoding process of the message bit. Although the proposed algorithm can be applied to lossless image formats like BMP, GIF, PNG or TIF, we have only concentrated on 24-bit TIF color image format for experimental purpose.

The remainder of the paper is organized as follows: Section 2 describes works done in steganography, especially RGB image steganography, whereas Section 3 presents related theory – classical LSB algorithm and parity concept – needed for the current study. The description of the proposed model and encoding and decoding processes are presented in Section 4. In Section 5, experimental results and discussion are given. Finally, Section 6 includes the conclusion and future studies.

2. Related Works

Recently, the popularity of the LSB method has attracted many researchers in modifying or hybridizing the standard LSB technique in image steganography. Some surveys on image steganography have highlighted different approaches [4-9]. Saha, *et. al.,* [10] proposed a scheme for hiding messages in 24-bit bitmap image where most frequent pixel values and modification of those pixels are calculated to hide the information. It is observed that efficiency of the approach is highly attributed to the number of eligible pixels, with the moderate number of eligible pixels producing better results. On the other hand, Islam [11] used 24-bit bitmap image for embedding messages using stream builder and parity concept. One stream is built by aggregating LSBs of 8 bytes of pixels and then parity values of these pixels decide how the message will be embedded. This concept is effective, but the embedding capacity is low. Akhtar, *et. al.,* [12] used bit-inversion strategy in such a way that less number of LSBs of pixels was altered. This strategy enhances imperceptibility significantly but requires extra storage to record the inversions. Jung and Yoo [13] proposed combining interpolation and LSB substitution for data hiding. Interpolation method, a preprocessing of cover images for getting better capacity and quality, scales up and down the cover image, whereas LSB substitute method is then applied for embedding. Secret data can be retrieved from stego image without extra information.

Juneja and Sandhu [14] proposed an edge-detection technique. Pixels of edges of the cover image are detected by advanced edge detection filter and messages are embedded in LSBs of the pixels using pseudo-random numbers. Likewise, Jain, *et. al.,* [15] also used edge detection technique to embed messages in cover images. Chakraborty, *et. al.,* [16] devised edge predictor named Modified Median Edge Detector (MMED) along with the adaptive strategy for selecting edge portion of a cover image. The binary payload is then embedded in the edge area of the cover image. Results show better embedding capacity. Wu, *et. al.,* [17] proposed a scheme using LSB replacement and pixel-value differencing method. This method embeds more secret data in edge areas than smooth areas in the cover image. Here former areas use the LSB replacement and latter arrears use pixel-value methods respectively. Tyagi, *et. al.,* [18] extended the approach where pixel-value differences along with pixel-value sum of two consecutive pixels are manipulated for hiding secret data. However, Mandal and Das [19] used pixel value differencing method for embedding the secret message in each of the components of a pixel in a color image.

Another method proposed by Das and Tuithung [20] was using Huffman encoding for embedding messages. Huffman encoding is performed over the secret message prior to embedding at the LSB of each of the pixel's intensities of a cover image. At the same time, Huffman encoded bit stream as well as Huffman table is also embedded into the cover image for standalone facilities of the stego-image. Sun [1] also employed Huffman table and Huffman encoding for embedding secret data into the edge of the cover image. At the same time, a correction technique called 2^k correction is introduced to minimize the discrepancy between the stego and cover image.

Many researchers have also expanded interest on embedding message not only in the 1st LSB position but also in higher LSB position, generally called K-bit LSB substitution method. Parvez and Gutub [21] proposed RGB intensity values of the pixel. In their approach, the lower color component contains more bits than the higher color component, with more than one LSB bit substitution occurred in the lower color component. Other K-bit substitution method in image steganography can be observed by Liao, *et. al.,* [22] and Nag, *et. al.,* [23] where 4 LSB bits of a cover image are replaced. Although embedding data in higher LSB layers is prone to less attack than those embedded in the lower layers, this enhances visual distortion in the stego-image. To minimize this noise, adaptive and optimized LSB methods have been used in some literature including [24-26].

3. Related Theory

3.1. Classical LSB Algorithm

In this technique, bits of the message are directly embedded into the LSB of the cover image in a deterministic sequence. This modification does not provide any impact on human perception due to the amplitude of the change being small. In terms of 24-bit RGB image, each pixel is derived from three primary colors: red, green and blue, and each primary color is represented by 8 bits. One can store 3 bits in each pixel by changing a bit of each of the red, green and blue color components. For example, inside 24-bit image we have three adjacent pixels (9 bytes), which are shown in Figure 1(a). Assume, we want to hide the letter 'a' (ASCII code of 'a' is 97, which is 01100001 in binary). Superimposing these 8 bits in sequence over the LSB of the 9 bytes above, we get the result as in Figure 1(b), (where bits in bold and underline indicate changes). In this way, message bits can be embedded in the cover image generating the stego-image from which message bit can be extracted. Figure 2, describes this overall process of LSB strategy.

11010101 (byte 1)	10001100 (byte 2)	00011000 (byte 3)		1101010**0** (byte 1)	1000110**1** (byte 2)	0001100**1** (byte 3)
11010111 (byte 4)	10001110 (byte 5)	00011011 (byte 6)	→	1101011**0** (byte 4)	1000111**0** (byte 5)	0001101**0** (byte 6)
11010101 (byte 7)	11010000 (byte 8)	00001011 (byte 9)		1101010**0** (byte 7)	1101000**1** (byte 8)	00001010 (byte 9)

(a) (b)

Figure 1. Example of Embedding 0/1 into Binary String

```
Select an  image and convert it into binary
Convert the secret  message into binary
while until all message bits are embedded
    Chose one pixel of an image and divide it into three channels: red, green and blue
    Select next three message bits sequence
    Replace LSB of each red, green and blue channel with these message bits
end while
Set the image to a new value and save it
```

Figure 2. Least Significant Bit (LSB) Strategy

3.2. Parity Concept

Usually, parity concept is intensively used in data communications for detecting errors [27]. It is the technique of sending a redundant bit to verify the integrity of the received data. In this paper, we simply define ‘parity’ as a number of 1s in a given binary stream. If a stream of bits with particular length L has an even number of binary 1, the parity of this stream is considered as even parity and we depict this as 1. On the other hand, when a stream of bits with particular length L has an odd number of binary 1, parity of this stream is considered as odd parity and we depict this as 0 (see Table 1). We can easily compute whether a stream is an even or odd parity by simply performing XNOR of all bits of a stream. For example, if $B = X_1 X_2 X_3 ... X_n$ are n length bits, then parity is $\rho = X_1 xnor X_2 xnor X_3 xnor ... X_n$. This ρ value is either 1 or 0, which represent even and odd parity respectively.

Table 1. Example of Parity Concept

Stream (L=8)	**Parity**
11010111	$\rho = 1$, even
11010101	$\rho = 0$, odd

4. Proposed Methodology

In this section, we present our proposed solution in the sequence of description of a model, encoding and decoding of algorithms and explanation of the encoding and decoding process.

4.1. Model Description

Our proposed method uses 24-bit RGB images as cover images. Like other 24-bit RGB images, each of these cover images is a collection of pixels and each pixel is represented by 3 bytes indicating the intensities of red, green, and blue channels in that pixel. In Figure 3(a), during the embedding process, each pixel of the cover image is converted into a byte array of red, green and blue channels (RGB) or blocks. Next, MSB value of these three bytes (each of red, green and blue channels) of this pixel works as an indicator that determine in which channel the message bit is kept. After selecting the proper channel, now the message bit is embedded in the selected channel using encoding. This encoding is done with parity checking concepts. If the parity value of the selected channel coincides with message bit, there is no change of LSB of selected channel. Otherwise, the LSB bit of the selected channel is toggled. In this way, the entire message is embedded into the cover image. Once the embedding process is completed, the image is transformed into a stego-image. During extracting process –shown in Figure 3(b), each pixel of this stego-image is converted into a byte array of RGB channels where MSB of the three bytes selects the desired channel containing message bits. Finally, the parity value of the selected channel indicates the desired message bit. In this way, the whole message is obtained by aggregating all message bits. Note that the following schematic diagram in Figure 3, depicts the entire concepts briefly, whereas pseudo-codes presented in following sub-section depict detailed explanation of the method.

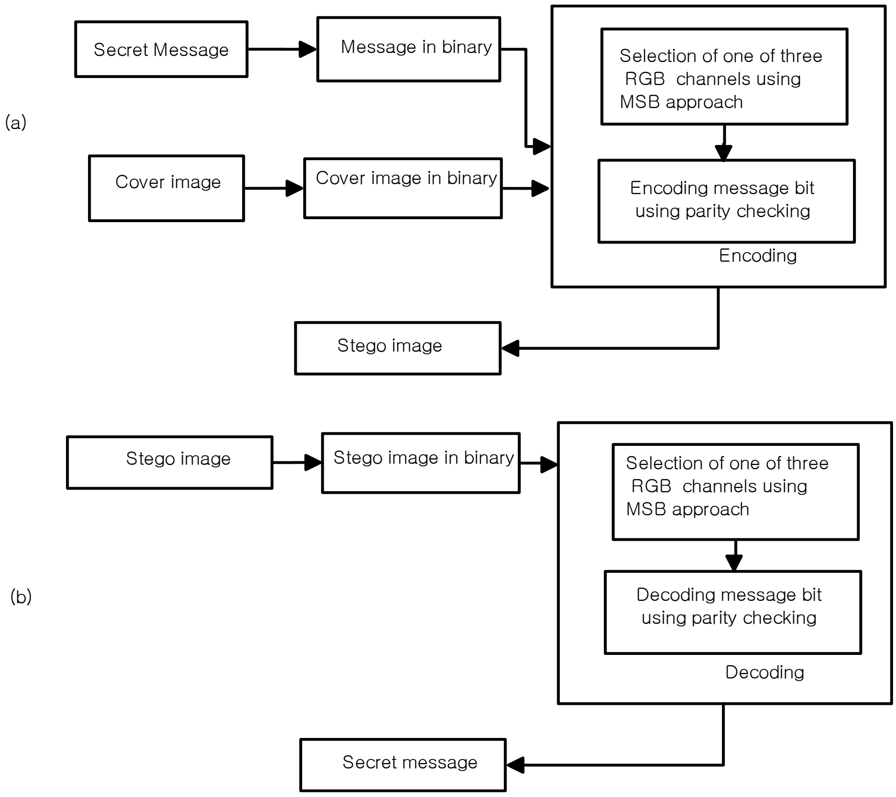

(a)
Secret Message
Message in binary
Cover image
Cover image in binary
Selection of one of three RGB channels using MSB approach
Encoding message bit using parity checking
Encoding
Stego image
(b)
Stego image
Stego image in binary
Selection of one of three RGB channels using MSB approach
Decoding message bit using parity checking
Decoding
Secret message

Figure 3. Overall Structure of the Proposed Technique (a) Encoding Process (b) Decoding Process

4.2. Encoding and Decoding Process Explanation

Following pseudo-codes depict how message bits are encoded in a cover image and decoded from a stego-image. We have considered following notations in our encoding and decoding process:

M: n-bit long embedding message
M_i: *1*-bit long message block
I_c: Cover image
P_i^r: 8-bit block of pixel from a cover image I_c (red channel); $P_i^r[j]$ indicates jth bit in the block
P_i^g: 8-bit block of pixel from a cover image I_c(green channel); P_i^g [j] indicates jth bit in the block
P_i^b: 8-bit block of pixel from a cover image I_c(blue channel); $P_i^b[j]$ indicates jth bit in the block
I_s: Stego-image
$P_i^{r'}$: 8-bit block of pixel from a stego-image I_s (red channel); $P_i^{r'}$ [j] indicates jth bit in the block
$P_i^{g'}$: 8-bit block of pixel from a stego-image I_s (green channel); $P_i^{g'}$ [j] indicates jth bit in the block
$P_i^{b'}$: 8-bit block of pixel from a stego-image I_s (blue channel); $P_i^{b'}$ [j] indicates jth bit in the block

```
Procedure GET-PARITY($P_i^x$)
  v←$P_i^x$ [j]
  for j←1 to 7  /*check the parity of a particular  channel;*/
    v←v xnor $P_i^x$ [j]    /*performing XNOR ;*/
  return v;
end Procedure
```

Figure 4. Procedure for Getting Parity

```
Procedure SELECTOR_RGB($P_i^r$, $P_i^g$,  $P_i^b$)
    $B_i$← $P_i^r$ [7]*$2^0$+ $P_i^g$ [7]*$2^1$+ $P_i^b$ [7]*$2^2$ /*MSB of three channel in decimal */
    if ($B_i$ mod 3)=0
    then return $P_i^x$←$P_i^r$ /*Red channel is selected */
    if ($B_i$ mod 3)=1
    then return $P_i^x$←$P_i^g$ /*Green channel is selected */
    if ($B_i$ mod 3)=2
    then return $P_i^x$←$P_i^b$ /*Blue channel is selected */
end Procedure
```

Figure 5. Selection of Color Channel Steps

```
Procedure  ENCODE-MESSAGE (M, $I_c$) /*Encoding process of message bit*/
for i←0 to n-1
    $P_i^x$ ← SELECTOR_RGB ($P_i^r$,  $P_i^g$,  $P_i^b$)
    if GET-PARITY ($P_i^x$)=$M_i$
    then $P_i^x$ [0] ← $P_i^x$ [0]   /*no change*/
    else if GET-PARITY ($P_i^x$)!=$M_i$
    then $P_i^x$ [0] ← ¬$P_i^x$ [0]  /*toggle*/
end Procedure
```

Figure 6. Encoding Process

```
Procedure DECODE-MESSAGE($I_s$)    /*Decoding process*/
  for i←0 to n-1
    $P_i^{x\prime}$← SELECTOR_RGB($P_i^{r\prime}$,  $P_i^{g\prime}$,  $P_i^{b\prime}$)
    if GET-PARITY ($P_i^{x\prime}$)=1
    then M[i] ←1
    else if GET-PARITY ($P_i^{x\prime}$)=0
    then M[i] ←0
  return M
end Procedure
```

Figure 7. Decoding Process

4.2.1. Encoding Step: For embedding a message in a cover image, Figure 6, shows the encoding process along with additional procedures described in Figure 4, and 5. According to the algorithms, in order to hide a message bit M_i into a particular cover image pixel P_i (=P_i^r or P_i^g or P_i^b) in I_c, MSB of P^i_r, P^i_g, and P_i^b channels are collected and performed a modulo operation on it by 3. This mod value generates any of the value 0, 1, 2 indicating P_i^r (red channel), P_i^g (green channel), and P_i^b (blue channel) respectively. In this way, a specific channel is selected for embedding. Afterward, parity value of the selected color channel P_i^x (=P_i^r or P_i^g or P_i^b) is compared with message bit M_i. This is important as it indicates how this message bit is encoded in that channel. For instance, if the parity value of P_i^x is even (p = 1) and message bit is 1, the LSB bit of this channel will be unchanged, but in the case of message bit being 0, single LSB of that channel P_i^x is toggled. All possible encoding actions can be described in Table 2. Above mentioned approach changes the LSB of each indicated RGB channels of I_c until the end of the embedding process of all message bits.

Table 2. Actions in Embedding a Message Bit in a Channel

Case	Parity of Channel	Message Bit	Action on LSB of channel
1	Even (1)	1	No
2	Even (1)	0	Toggle the bit
3	Odd (0)	1	Toggle the bit
4	Odd (0)	0	No

In order to illustrate encoding algorithms, we take message bits 1100 and four pixels of a cover image, with each pixel containing R, G and B channels. First message bit, which is 1 has to be embedded in one of the channels of a first pixel. MSB of these three channels is $101_2 = 5_{10}$ and after the modulus arithmetic (5 mod 3), the result is 2. Note that modulo 3 indicates the number of channels. Therefore, according to the algorithm, the blue channel will be selected for embedding messages. Now, this blue channel has an odd number of 1, so its parity is 0. As the parity value is different from the message bit, LSB position of the blue channel will be toggled indicating message bit has been embedded. In this way, the first message bit is embedded. Similarly, 2nd, 3rd, and 4th message bits are embedded in 2nd, 3rd, and 4th pixels. Table 3, illustrates this process. It is also observed that only two LSB of 4 pixels has been altered for embedding 4 message bits (underlined fonts).

Table 3. Embedding Message using Proposed Encoding Approach

Pixel	Channel	Channels' Values in Binary (Cover Image)	MSB of Three Channels	Selected Channel [MSB mod 3]	Parity of the Selected Channel	Message Bit	Channels' Values in Binary (stego-image)
1nd	R	10001010	$101_2 = 5_{10}$	2 B	0	1	10001010
	G	00011100					00011100
	B	11001110					1100111$\underline{1}$
2rd	R	11110000	$110_2 = 6_{10}$	0 R	1	1	11110000
	G	10001111					10001111
	B	01001110					01001110
3th	R	01001010	$010_2 =2_{10}$	2 B	1	0	01001010
	G	10001101					10001101
	B	00110000					0011000$\underline{1}$
4th	R	00001111	$001_2 = 1_{10}$	1 G	0	0	00001111
	G	00001101					00001101
	B	10000000					10000000

4.2.2. Decoding Steps: Decoding process starts with the decoding algorithm presented in Figure 7, and additional procedures shown in Figure 4, and Figure 5. In order to extract the secret message bit M_i from the stego-image pixel $P_i'=(P_i^{r\prime}, P_i^{g\prime}, P_i^{b\prime})$ in I_s, particular channel $P_i^{x\prime}$ (from the MSB of $P_i^{r\prime}$, $P_i^{g\prime}$ and $P_i^{b\prime}$) is selected by performing the modulo operation of MSB bits with 3. This selected channel contains the desired message bit, which can be obtained by the parity value of this channel. Even (p = 1) parity value of $P_i^{x\prime}$ indicates the message bit is 1, whereas odd (p = 0) parity value of $P_i^{x\prime}$ indicates the message bit is 0. The iteration will continue until all message bits are extracted. It is observed that the extracted message is the same as the embedded message.

To explain the decoding process with an example, we take the four stego-image pixels from previous encoding steps (See Table 4). MSB of the first pixel is $101_2 = 5_{10}$ and after a module it with 3 produces 2. Therefore, first message bit can be found from blue component. As the parity value of the blue component is even (1), the message bit is 1. Similarly, other message bits can be extracted from the rest of the pixels. Table 4, describes this decoding process. Moreover, it is observed from Table 3, and Table 4, that same message bits are obtained before encoding and after decoding process.

Table 4. Extracting Message from a Stego-image Portion using Proposed Decoding Approach

Pixel	Channel	Channels' Values in Binary (stego- image)	MSB of three Channels	Selected Channel [MSB mod 3]	Parity of the Selected Channel	Message Bit
1nd	R	10001010	101_2 $= 5_{10}$	2 B	1	1
	G	00011100				
	B	11001111				
2rd	R	11110000	110_2 $= 6_{10}$	0 R	1	1
	G	10001111				
	B	01001110				
3th	R	01001010	$010_2=$ 2_{10a}	2 B	0	0
	G	10001101				
	B	00110001				
4th	R	00001111	001_2 $= 1_{10}$	1 G	0	0
	G	00001101				
	B	10000000				

5. Result and Discussion

5.1. Experimental Setting and Assessment of Image Quality

Several experiments were performed to evaluate the efficiency of the proposed scheme. 18 colors (24 bits) images with size 512x 512 in TFT formats were used as cover media. These images are taken from the USC-SIPI Image Database (available at http://sipi.usc.edu/database/). For the case of Inserting a secret message, a series of pseudo-random numbers were generated and the bit streams of these numbers were embedded into the cover images. The experiment is simulated using MATLAB 10 as well as Java SE on Windows 7. Four different methods including MSE (mean square of error), PSNR (peak signal-to-noise ratio), SSIM (structural similarity measure), and bits ratio (BR) have been incorporated for evaluating the efficiency of the proposed method. Equation (1) - (5) shows these comparison metrics.

MSE is cumulative squared error between cover and its corresponding stego-image. MSE can be defined as:

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

$$MSE_{avg} = \frac{MSE_R + MSE_G + MSE_B}{3} \quad (2)$$

Where m and n signify the height and width of the images respectively. $I(i, j)$ and $K(i, j)$ represent pixel values of the cover and the stego-images respectively. MSE_R, MSE_G, and MSE_B indicate mean square errors in red, green, and blue channels respectively.

PSNR is a statistical measurement used for digital image or video quality assessment [28]. The PSNR is estimated in decibel (dB) and is defined as:

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE_{avg}}\right) \tag{3}$$

SSIM is also used for assessing the quality between original and distorted image. This metric shows better results than PSNR and MSE, which are inconsistent with human perception. SSIM is defined as [29] :

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{4}$$

Where μ_x is the mean of X, μ_y is the mean of Y, σ_x^2 is the variance of X, σ_y^2 is the variance of Y, and σ_{xy} is covariance of XY. C_1 and C_2 are constants with $C_1 = (k_1L)^2$, and $C_2 = (k_2L)^2$. L is also the dynamic range of the pixel-values. k_1 and k_2 have default values as 0.01 and 0.03 respectively.

BR is a measurement which indicates exact number of bits changed within the image when imbedding the secret bits, *i.e.*, generating the stego-image. It can be measured in percentage.

$$BR = \frac{no.\ of\ \ replaced\ bits}{no.\ of\ \text{cover}\ \ image\ bits} \times 100\% \tag{5}$$

After embedding full capacity secret messages in eighteen different cover images, the result of the stego-image for each cover image was produced and then MSE_avg, PSNR, SSIM, BR values of the stego-images were calculated. Table 5, shows the comparison result of proposed algorithm between cover and stego image of eighteen 512*512 (24 bit color) size. Besides, embedding capacity of each image is 262000 bits. It is observed that for all eighteen stego-images, the mean of MSE_avg values were between 0.034680 and 0.243265±0.055779, PSNR values were between 54.27 dB and 62.73dB±2.289, SSIM values were between 0.999459and 0.999932±0.000109, BR values were between 0.2869and 0.6237±0.0988.

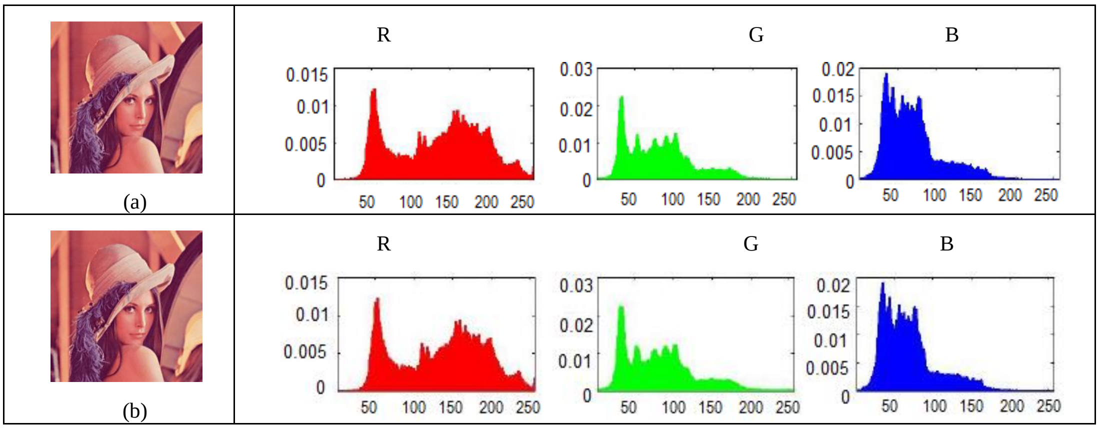
Table 5. Performance Result of Proposed Algorithm

Color image of Size (512*512)	MSE_avg	PSNR	SSIM	BR %
Lena	0.03468	62.73	0.999869	0.2869
Baboon	0.075003	59.38	0.999852	0.4160
Peppers	0.088937	58.64	0.999782	0.4622
Sailboat	0.178271	55.62	0.999811	0.5943
Tiffany	0.123618	57.21	0.999459	0.5042
Splash	0.150689	56.35	0.999898	0.5273
Airplane	0.046782	61.43	0.999932	0.3200
San Francisco	0.048762	61.25	0.999915	0.3395

Oakland	0.058895	60.43	0.999924	0.3531
Woodland Hills	0.067621	59.83	0.999731	0.3823
Foster City	0.171427	55.79	0.999795	0.5769
Earth from space	0.133684	56.87	0.999801	0.5200
Downtown	0.073296	59.48	0.999878	0.4059
Shelter Island	0.061529	60.24	0.999910	0.3639
Point Loma	0.086513	58.76	0.999883	0.4497
North Island NAS	0.106189	57.87	0.999812	0.4679
Golden Gate	0.243265	54.27	0.999789	0.6237
Miramar NAS	0.064429	60.04	0.999899	0.3781
Min*	0.034680	54.27	0.999459	0.2869
Max*	0.243265	62.73	0.999932	0.6237
SD*	0.055779	2.289	0.000109	0.0988

*Min=minimum, Max=maximum, SD=standard deviation

Figure 8(a), shows the cover and corresponding stego image of Lena with color histogram plots for all three channels. It is observed that there are little differences in the histograms of cover and stego image of Lena. For other images, similar characteristics are obtained.



(a)

(b)

Figure 8. Cover and Stego Images of Lena: (a) Cover Image and RGB Histograms (b) Stego Image and RGB Histogram

From above experimental results, it is apparent that each stego-image is almost analogous to corresponding cover image and shows better imperceptibility. That is, deterioration of the quality of images due to the embedding of the secret messages cannot be distinguished. Our proposed method is robust and secure in the sense that the algorithm alters a small number of bits during embedding a large message and makes the text message difficult to identify. Besides, if stego-analyzer detects our stego-images anyway, it is not easy to reveal the message bits due to the fact that proposed approach does not put the message directly at LSB positions.

5.2. Comparison with the State-of-the-Arts

In order to get the viability of the proposed system, we compared it with several previous works done on LSB image steganography. For comparison purpose, the same cover images with the same resolution (512*512) were considered.

Table 6. Comparison of the Proposed Approach with the Literature

Steganographic Methods	Cover Images (512*512)	Payload Capacity (Bits)	PSNR of Stego-Images in(dB)
Mandal and Das [19]	Lena	145,787	42.26
Lin, *et. al.,* [30]	Lena	20,032	53.78
Wu, *et. al.,* [17]	Airplane	409,752	40.13
Islam [11]	Lena	98,285	60.35
Abduallah, *et. al.,* [31]	Lena	609,129	32.87
Our proposed method	Lena	262,000	62.73
Our Proposed Method	Airplane	262,000	61.43

From the Table 6, it is observed that our proposed approach is able to produce better capacity and PSNR value compared to approaches introduced by Mandal and Das [19], Lin, *et. al.,* [30] and Islam [11]. Moreover, although methods proposed by Wu, *et. al.,* [17] and Abduallah, *et. al.,* [31] have higher hiding capacity than our approach, in terms of PSNR value our approach outweighs them. In general, our method is able to produce better results compared to others in terms of PSNR values. However, our approach did not always produce higher payload values but it does produce a competitive results. The explanation can be expressed in following ways:

In steganography, it is always desirable to increase the steganographic capacity and enhance imperceptibility (PSNR in our case). However, steganographic capacity and imperceptibility conflict with each other [10-32]. Therefore, it is not possible to simultaneously maximize the imperceptibility and capacity of steganography systems [33]. Usually, those methods producing better payload capacity might not be good at in producing well PSNR values and vice-versa. Characteristics observed in Table 6 support this statement as well. Therefore, researchers have to always trade-off between two purposes. As our focus was given to increase imperceptibility, we have to trade-off with payload capacity. The capacity, however, is not inferior.

6. Conclusions

In this paper, we have devised a novel approach to hiding messages in lossless RGB images. It presents an improved LSB image steganography method where each encoded message bit is embedded in one of the three RGB channels (indicator/selector) on the basis of the MSBs of channels, with encoding being obtained based on the parity values of that selected channel. The experimental results demonstrate that our method has primarily shown significant improvements in terms of imperceptibility and robustness. Although the capacity is not very high, higher payload capacity has to be sacrificed for higher imperceptibility. The main contribution of this study is that decent number of secret message bits is encoded into LSB positions effectively by altering comparatively a few numbers of cover image bits and without direct involvement of any stego-keys. Our scheme is straightforward in generating quality stego-image, and feasible for other steganographic fields such as audio/video steganography. In the future, we could extend our study by adding more LSB (K^{th} LSB bit) bits. This is likely to improve the capacity of image steganography. Besides, employing other transformation equation in the selection of proper channel might strengthen the security of the stego-image.

References

[1] S. Sun, “A Novel Edge Based Image Steganography with 2k Correction and Huffman Encoding”, Information Processing Letters., vol. 116, no. 2, **(2016)**, pp. 93-99.

[2] E. Zielińska, W. Mazurczyk and K. Szczypiorski, “Trends in Steganography", Commun. ACM., vol. 57, **(2014)**, pp. 86-95.

[3] A. K. Mandal, M. Kaosar, M. O. Islam and M. D. Hossain, “An Approach for Enhancing Message Security in Audio Steganography”, Proceedings of the 16th International Conference on Computer and Information Technology (ICCIT), Khulna, Bangladesh, **(2014)** April 20-23.

[4] M. S. Subhedar and V. H. Mankar, “Current Status and Key Issues in Image Steganography: A survey”, Computer Science Review., vol. 13–14, no. 11, **(2014)**, pp. 95-113.

[5] Y. J. Chanu, T. Tuithung and K. M. Singh, “A Short Survey on Image Steganography and Steganalysis Techniques”, Proceedings of the 3rd National Conference on Emerging Trends and Applications in Computer Science (NCETACS), India, **(2012)** April 30-31.

[6] N. Hamid, A. Yahya, R. B. Ahmad and O. M. Al-Qershi, “Image Steganography Techniques: An Overview”, International Journal of Computer Science and Security (IJCSS)., vol. 6, no.1, **(2012)**, pp. 168-187.

[7] X. Niu, M. Ma, R. Tang and Z. Yin, “Image Steganography Via Fully Exploiting Modification Direction”, International Journal of Security and Its Applications., vol. 9, no. 20, **(2015)**, pp. 243-256.

[8] A. Cheddad, J. Condell, K. Curran and P. Mc Kevitt, “Digital Image Steganography: Survey and Analysis of Current Methods”, Signal Processing., vol. 90, no. 3, **(2010)**, pp. 727-752.

[9] S. Ashwin, J. Ramesh, S. A. Kumar and K. Gunavathi, “Novel and Secure Encoding and Hiding Techniques using Image Steganography: A Survey”, Proceedings of the International Conference on Emerging Trends in Electrical Engineering and Energy Management (ICETEEEM), India, **(2012)** December 13-15.

[10] A. Saha, S. Halder and S. Kollya, “Image Steganography using 24-bit Bitmap Images”, Proceedings of the 14th International Conference on Computer and Information Technology (ICCIT), Bangladesh, **(2011)** Noveber 22-24.

[11] M. O. Islam, “A High Embedding Capacity Image Steganography using Stream Builder and Parity Checker”, Proceedings of the 15th International Conference on Computer and Information Technology (ICCIT), Dhaka, Bangladesh, **(2012)** Noveber 22-24.

[12] N. Akhtar, S. Khan and P. Johri, “An Improved Inverted LSB Image Steganography”, Proceedings of the International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), **(2014)** February 7-8.

[13] K. H. Jung and K. Y. Yoo, “Steganographic Method Based on Interpolation and LSB Substitution of Digital Images”, Multimedia Tools and Applications., vol. 74, **(2014)**, pp. 2143-2155.

[14] M. Juneja and P. S. Sandhu, “An Improved LSB Based Steganography Technique for RGB Color Images”, International Journal of Computer and Communication Engineering., vol. 2, **(2013)**, pp. 513-517.

[15] N. Jain, S. Meshram and S. Dubey, “Image Steganography Using LSB and Edge Detection Technique”, International Journal of Soft Computing and Engineering (IJSCE)., vol. 223, **(2012)**, pp 1-12.

[16] S. Chakraborty, A. S. Jalal and C. Bhatnagar, “LSB Based Non Blind Predictive Edge Adaptive Image Steganography”, Multimedia Tools and Applications., **(2016)**, pp. 1-15.

[17] H. C. Wu, N. I. Wu, C.-S. Tsai and M.-S. Hwang, “Image Steganographic Scheme Based on Pixel-value Differencing and LSB Replacement Methods," IEEE Proceedings-Vision, Image and Signal Processing, vol. 152, **(2005)**, pp. 611-615.

[18] A. Tyagi, R. Roy and S. Changder, “High Capacity Image Steganography Based on Pixel Value Differencing and Pixel Value Sum”, Proceedings of the Second International Conference on Advances in Computing and Communication Engineering (ICACCE), **(2015)** May 12-14.

[19] J. Mandal and D. Das, “Colour Image Steganography Based on Pixel Value Differencing in Spatial Domain”, Proceedings of the International Conference on Electronic Engineering and Computer Science, **(2013)** November 12-13.

[20] R. Das and T. Tuithung, “A Novel Steganography Method for Image Based on Huffman Encoding”, Proceedings of the 3rd National Conference on Emerging Trends and Applications in Computer Science (NCETACS), **(2012)** March 30-31.

[21] M. T. Parvez and A. A. Gutub, “RGB Intensity Based Variable-bits Image Steganography”, Proceedings of the Asia-Pacific Services Computing Conference, **(2008)** December 9-12.

[22] X. Liao, Q. Wen and J. Zhang, “A Steganographic Method for Digital Images with Four-pixel Differencing and Modified LSB Substitution”, Journal of Visual Communication and Image Representation, vol. 22, no. 1, **(2011)**, pp. 1-8.

[23] A. Nag, S. Ghosh, S. Biswas, D. Sarkar and P. P. Sarkar, “An Image Steganography Technique using X-box Mapping”, Proceedings of the International Conference on Advances in Engineering, Science and Management (ICAESM), **(2012)** Mach 30-31.

[24] N. N. El-Emam, “New Data-hiding Algorithm Based on Adaptive Neural Networks with Modified Particle Swarm Optimization”, Computers & Security, vol. 55, no. 11, **(2015)**, pp. 21-45.

[25] K. Muhammad, J. Ahmad, N. U. Rehman, Z. Jan and M. Sajjad, "CISSKA-LSB: Color Image Steganography using Stego Key-directed Adaptive LSB Substitution Method", Multimedia Tools and Applications, **(2016)**, pp. 1-30.

[26] H. R. Kanan and B. Nazeri, "A Novel Image Steganography Scheme with High Embedding Capacity and Tunable Visual Image Quality Based on a Genetic Algorithm", Expert Systems with Applications, vol. 41, **(2014)**, pp. 6123-6130.

[27] B. A. Forouzan, "Data Communications Networking", McGraw-Hill Science, USA, **(2006)**.

[28] M. Devi and N. Sharma, "Improved Detection of Least Significant Bit Steganography Algorithms in Color and Gray Scale Images", Recent Advances in Engineering and Computational Sciences (RAECS), **(2014)**, pp. 1-5.

[29] R. Karakış, İ. Güler, İ. Çapraz and E. Bilir, "A Novel Fuzzy Logic-based Image Steganography Method to Ensure Medical Data Security", Computers in Biology and Medicine, vol. 67, no. 1, **(2015)**, pp. 172-183.

[30] T. I. Lin, Y. T. Chang and W. N. Lie, "A Framework of Enhancing Image Steganography with Picture Quality Optimization and Anti-steganalysis Based on Simulated Annealing Algorithms", IEEE Transactions on Multimedia, vol. 12, no. 1, **(2010)**, pp. 345-357.

[31] W. M. Abduallah, A. M. S. Rahma and A. S. K. Pathan, "Mix Column Transform Based on Irreducible Polynomial Mathematics for Color Image Steganography: A Novel Approach", Computers & Electrical Engineering, vol. 40, no. 5, **(2014)**, pp. 1390-1404.

[32] H. Wang and S. Wang, "Cyber Warfare: Steganography vs. Steganalysis", Communications of the ACM., vol. 47, **(2004)**, pp. 76-82.

[33] S. Venkatraman, A. Ajith and M. Paprzycki, "Significance of Steganography on Data Security", in Information Technology: Coding and Computing., vol. 2, **(2004)**, pp. 347-351.

Authors

Ashis Kumar Mandal, is an Assistant Professor in the Department of Computer Science and Engineering at Hajee Mohammad Danesh Science and Technology University (HSTU), Bangladesh. He obtained his B.Sc.(Eng.) in Computer Science and Engineering from Shahjala University of Science and Technology (SUST), Sylhet, Bangladesh and his Master in Computer Science from University Malaysia Pahang, Malaysia. His current research topics include: data and network security, heuristics and meta heuristics search, and data mining

M N M Kahar, received PhD degree in Computer Science from University of Nottingham, United Kingdom, in 2011. Since then, he has been with University Malaysia Pahang (UMP), Malaysia, where he is currently a Senior Lecturer in Faculty of Computer Systems and Software Engineering (FSKKP). His research interests include solving real world optimization problems, such as scheduling, timetabling, routing problem using meta-heuristics approaches.