Proposed Implementation Method to Improve LSB Efficiency

Ziad Alqadi\textsuperscript{1}; Bilal Zahran\textsuperscript{2}; Qazem Jaber\textsuperscript{3}; Belal Ayyoub\textsuperscript{4}; Jamil Al-Azzeh\textsuperscript{5}; Ahmad Sharadqh\textsuperscript{6}

\textsuperscript{1}Computer Engineering Department, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: Natalia_maw@yahoo.com
\textsuperscript{2}Computer Engineering Department, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: zahranb@yahoo.com
\textsuperscript{3}Department of Mechatronics Engineering, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: qazemjaber@gmail.com
\textsuperscript{4}Computer Engineering Department, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: belal_ayyoub@hotmail.com
\textsuperscript{5}Computer Engineering Department, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: azzehjamil@gmail.com
\textsuperscript{6}Computer Engineering Department, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: azzehjamil@gmail.com

Abstract: Data steganography is an important process, which is needed in various human life applications, so the need for efficient method of data hiding is vital issue. In this paper we will introduce a new method of LSB implementation, the proposed method will decrease the hiding process time to make this process more efficient, doing this the proposed method must keep other parameter such MSE and PSNR closed to that obtained by a popular standard method of LSB implementation.

Keywords: Steganography, hiding time, MSE, PSNR, speedup, capacity.

1- Introduction

Steganography as shown in figure (1) is the process of hiding data into another data, the data where to hide another data is called a holding data [1-60]. Steganography is widely used in deferent life applications in order to protect personal, private and confidential data from seeing by an authorized or third party [3], so data hiding process is one of the most popular methods used to secure the data from being hacked by any other party, that has no relation with data sent or received via un secure environment such as the internet [4], [5].
Figure (1): Data steganography process

Digital color image [6], [7] are usually presented by a 3D matrix, one dimension for one of the three colors (Red, green, and blue); the 3D matrix has a huge size capable to hold a big message in size.

The selected method for data steganography must perform the hiding process to insert the secret message into the covering image without destroying the image, and must perform the extracting process to get the original secret message.

The selected method of data steganography must have the following features:

- **Capacity**: This feature means the amount of data (message length in bytes) which can be hidden in the covering image without destroying the image, here it is reasonable to use a high resolution color image (with huge size) to hold a long secret message [8], [9].

- **Minimizing the negative effects of embedding a message in color image by minimizing a mean square error (MSE) [10] between the original covering image and the holding image, or maximizing peak signal to noise ratio (PSNR) [10], [13], [14] between both images, the covering and holding.**

- **Accuracy**, which means that the extracted message is the same as hidden message [11], [12].

- **Efficiency**: here we have to look for minimizing both of the hiding time and extracting time.

2- **Least significant bit method of steganography**

Least significant bit (LSB) is the most popular method of data hiding-extracting because of many reasons like [9], [12]:

- Very simple to be implemented.

- Provides low value for MSE and high value for PSNR, making it difficult to notice the differences between the covering image and the holding image by human eyes.
- Very accurate by extracting the message without any loss of information.
- Provides low times for hiding and extracting processes.
- It suits the process of hiding short messages and very long messages,
- The capacity of this method is equal the image size divided by 8, so using big images will allow us to hide huge messages [2].

LSB method as shown in figure (2) reserves 8 bytes from the covering image to hide single character from the message, so we have to get the binary version of the character byte, and take each bit of this character to be inserted in the corresponding least significant bit of the image byte as shown in figure (2).

![Figure (2): LSB hiding and extracting](image)

### 3- First method of LSB implementation

This method of LSB implementation is the most traded and used by the users and researchers, but this method of implementation has a high hiding time specially, when it used for hiding short a medium messages.

The hiding process of this method can be implemented applying the following steps:

1) Get the covering image, define the image size.
2) Get the secret message, define the secret message length.
3) Get the decimal values of the message.
4) Get the transpose binary version of the message.
5) Reshape the transpose binary version to one column array.
6) For each bit in the reshaped array do the following:
   i. Get the LSB of the corresponding byte of the covering image by applying modulus 2 of the image byte.
ii. If the LSB bit equal the bit of the reshaped array of the message then the holding byte must equal the covering byte, else add the bit to covering byte to get the covering byte.

The extracting process can be implemented applying the following steps:

1) Get the message length.

2) For each character to be extracted reserve a set of 8 bytes from a holding image.

3) Get the LSB from each byte in the set.

4) Multiply each bit by the corresponding weight \([128, 64, 32, 16, 8, 4, 2, 1]\).

5) Summate the results of step 4 to get the decimal value of the character.

4- Proposed method of LSB implementation

This method uses LSB concepts and can be implemented applying the following steps:

1) Get the covering image and reshape it to one row array.

2) For each character (a) reserve an array of 8 bytes from the covering image (P).

3) Set the multiplier cc to 128.

4) Set I to 1.

5) Get the covering byte (S) by applying the following steps:

   i. Get a1 by applying anding (a) with (cc).

   ii. Bit shift left (a1) \((i-1)\) times (a2).

   iii. Shift right a2 7 times to get a3.

   iv. Bit anding P (I) with 254 to get a4.

   v. Get the holding byte S (I) by oring a3 and a4.

   vi. Increment I

   vii. Divide cc by 2.

   viii. If I less than 8 go to step 5

6) Repeat step 5 for each character in the message.

The extracting phase can be implemented applying the following steps:

For each character apply the following steps:

1) Get the set of holding 8 bytes in the image (S).
2) Set the decimal value (DV) of the character to zero.

3) Set the multiplier (VV) to 128.

4) Set I to 1.

5) Find the modulus 2 of S (I) (e1).

6) Multiply e1 by VV. (e2)

7) Add e2 to DV.

8) Divide VV by 2.

9) Increment I.

10) If I less than 8 go to step 5

11) Store DV as a byte of the message.

The following example shows how this method works:

Hiding process

Character "B" decimal a=66

cc=128

Covering bytes p= [200, 191, 189, 76, 89, 240, 80, 101]

<table>
<thead>
<tr>
<th>i</th>
<th>cc</th>
<th>a1=bitand(a,cc)</th>
<th>a1=uint8(bitshift(a1,i-1))</th>
<th>a1=(bitshift(a1,-7))</th>
<th>s(i)=uint8(bitand(p(i),254))</th>
<th>s(i)=uint8(bitor(s(i),a1))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>128</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>200</td>
<td>200</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>64</td>
<td>128</td>
<td>1</td>
<td>190</td>
<td>191</td>
</tr>
<tr>
<td>3</td>
<td>32</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>188</td>
<td>188</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>76</td>
<td>76</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>88</td>
<td>88</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>240</td>
<td>240</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
<td>128</td>
<td>1</td>
<td>80</td>
<td>81</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

Extracting process
Holding bytes $s= [200 \ 191 \ 188 \ 76 \ 88 \ 240 \ 81 \ 100]$

<table>
<thead>
<tr>
<th>i</th>
<th>vv</th>
<th>a1=mod(s(i),2)</th>
<th>a1=a1*vv</th>
<th>ccc=ccc+a1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>128</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>1</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>3</td>
<td>32</td>
<td>0</td>
<td>0</td>
<td>64</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>64</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>66</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>66</td>
</tr>
</tbody>
</table>

5- Implementation and experimental results:

The first and the proposed methods of LSB were implemented using matlab codes, figure (3) shows the original image which was prepared to hold a secret message with length $= 500$ characters, figure (4) shows the holding image after applying the first method, while figure (5) shows the holding image after applying the proposed method:

Figure (3): Original covering image
From figures (3), (4) and (5) we can see that the holding images using both methods are closed together, the color histograms of the three images are much closed and it is very difficult to notice any changes by human eyes, which tells us that both methods satisfy the requirements of data steganography.

The following 2 experiments were implemented to discuss some parameters of the both methods:

**Experiment 1: Evaluating methods performance (efficiency)**

In this experiment we took a huge color image to be used as a holding image (size=2500608 bytes), we categorized the secret message into three groups:

- Short messages with length range 100 to 1000 characters.
- Medium messages with length range 1600 to 6400 characters.
- Huge messages with length above 12800 characters.

The experimental results of this experiment are shown in table (1)

Table (1): Results of experiment 1

<table>
<thead>
<tr>
<th>Message length(byte)</th>
<th>Hiding time(seconds):First method</th>
<th>Hiding time(seconds):Proposed method</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.1350</td>
<td>0.001000</td>
<td>135.0000</td>
</tr>
<tr>
<td>200</td>
<td>0.1410</td>
<td>0.003000</td>
<td>47.0000</td>
</tr>
<tr>
<td>400</td>
<td>0.1280</td>
<td>0.004000</td>
<td>32.0000</td>
</tr>
<tr>
<td>800</td>
<td>0.1300</td>
<td>0.008000</td>
<td>16.2500</td>
</tr>
<tr>
<td>1600</td>
<td>0.1380</td>
<td>0.013000</td>
<td>10.6154</td>
</tr>
<tr>
<td>3200</td>
<td>0.1390</td>
<td>0.028000</td>
<td>4.9643</td>
</tr>
<tr>
<td>6400</td>
<td>0.1420</td>
<td>0.050000</td>
<td>2.8400</td>
</tr>
<tr>
<td>12800</td>
<td>0.1500</td>
<td>0.097000</td>
<td>1.5464</td>
</tr>
<tr>
<td>25600</td>
<td>0.2165</td>
<td>0.198000</td>
<td>1.0934</td>
</tr>
<tr>
<td>51200</td>
<td>0.6860</td>
<td>0.459000</td>
<td>1.4946</td>
</tr>
<tr>
<td>102400</td>
<td>1.2390</td>
<td>0.776000</td>
<td>1.5966</td>
</tr>
</tbody>
</table>

For table (1) we can see that the proposed method is more efficient than method 1. The speedup of using the proposed method (hiding time of method1 divided by hiding time of the proposed method) is range from 135 to 1.5 for various in length secret messages, which means that the speed of hiding process for the proposed method is much higher than the speed of the first method as we can see in figure (6).
Experiment 2: Evaluating methods MSE and PSNR

In this experiment we took a huge color image to be used as a holding image (size=2500608 bytes), we categorized the secret message into three groups:

- Short messages with length range 100 to 1000 characters.
- Medium messages with length range 1600 to 6400 characters.
- Huge messages with length above 12800 characters.

The experimental results of this experiment are shown in table (2)

Table (2): Results of experiment 2

<table>
<thead>
<tr>
<th>Message length(byte)</th>
<th>First method</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>PSNR</td>
</tr>
<tr>
<td>100</td>
<td>0.00015316</td>
<td>198.6654</td>
</tr>
<tr>
<td>200</td>
<td>0.00030633</td>
<td>191.7339</td>
</tr>
<tr>
<td>400</td>
<td>0.00061265</td>
<td>184.8024</td>
</tr>
<tr>
<td>800</td>
<td>0.0012</td>
<td>177.8710</td>
</tr>
<tr>
<td>1600</td>
<td>0.0025</td>
<td>170.9395</td>
</tr>
</tbody>
</table>
From table (2) we can raise the following facts:

- PSNR for the proposed method is much closed to PSNR for method 1 (see figure (7)).
- MSE for the proposed method is much closed to MSE for method 1 (see figure (8)).
- MSE for both methods is very low, and PSNR for both methods is high, which means that the holding image is much close to the original covering image, and because of this it will be very difficult to notice the differences between the 2 images by human eyes.

<table>
<thead>
<tr>
<th>Message size (byte)</th>
<th>First method</th>
<th>Second method</th>
</tr>
</thead>
<tbody>
<tr>
<td>3200</td>
<td>0.0049</td>
<td>164.0080</td>
</tr>
<tr>
<td>6400</td>
<td>0.0098</td>
<td>157.0765</td>
</tr>
<tr>
<td>12800</td>
<td>0.0196</td>
<td>150.1451</td>
</tr>
<tr>
<td>25600</td>
<td>0.0392</td>
<td>143.2136</td>
</tr>
<tr>
<td>51200</td>
<td>0.0784</td>
<td>136.2821</td>
</tr>
<tr>
<td>102400</td>
<td>0.1568</td>
<td>129.3506</td>
</tr>
</tbody>
</table>

Figure (7): PSNR comparisons
Conclusion

A method of LSB implementation was proposed. The proposed method was implemented and tested; the obtained experimental results were analyzed.

The analysis of experimental results showed that the proposed method is more efficient by decreasing the hiding time, and the proposed method has a noticeable speedup when hiding various secret messages.

The proposed method gave a low MSE, and high PSNR, the values of these parameters are acceptable.

References

8. N. Akhtar, P. Johri, S. Khan, “Enhancing the security and quality of LSB based image steganography”, 5th International Conference on Computational Intelligence and Communication Networks, Mathura, India,


Jamil Al-Azzeh, Bilal Zahran and Ziad Alqad, Salt and Pepper Noise: Effects and Removal International Journal on Electrical Engineering and Informatics 2(4),


Jamil AL-Azzeh, Qualitative risk analysis of software development ; Asian Journal of Information Technology July 2018.


Jamil AL-Azzeh, Analysis of Self-Similar Traffic Models in Computer Networks; International Review on Modelling and Simulations; October 2017 Volume 10, Number 5.

Jamil AL-Azzeh: Statistical Analysis of Methods Used to Enhanced Color Image Histogram; XX International
Scientific and Technical Conference; Russia May 24-26, 2017.


42. Jamil AL-Azzeh, Computer simulation of vibration robot created for the wall movement; Research Journal of Applied Sciences.; 2014 , Issue: 9, Page No.: 597-602 ,


© 2019, IJCSMC All Rights Reserved 318
54. Jamil Al-Azzeh#, Bilal Zahran# , Ziad Alqadi#, Belal Ayyoub#, Muhammed Mesleh ; A Novel Based on Image Blocking Method to Encrypt-Decrypt Color; INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION VOL 3 (2019) NO 1