Modified Inverse LSB Method for Highly Secure Message Hiding

Mohammed Abuzalata¹; Ziad Alqadi²; Jamil Al-Azzeh³; Qazem Jaber⁴

¹Computer Engineering Department, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: abuzalata@bau.edu.jo
²Computer Engineering Department, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: Natalia_maw@yahoo.com
³Computer Engineering Department, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: azzehjamil@gmail.com
⁴Computer Engineering Department, Al Balqa’a Applied University, Amman, 11134, Jordan
E-mail: qazemjaber@gmail.com

Abstract: The data may be very important and very confidential and when sent in an insecure environment may be stolen or being snatched on them, which eliminates the importance and confidentiality, so the need for data hiding becomes a very important issue to protect the data and prevent unauthorized party seeing or reading it. In this paper we will investigate a methodology to increase the security level of LSB method of data hiding. The process of data hiding and data extracted will tested in order to obtain an acceptable parameters, which allow us to adopt this methodology.

Keywords: LSB, steganography, hiding time, extracting time, reference, PSNR, MSE, Covering image, holding image, secret message.

1- Introduction

True color image can be represented by a 3D matrix [1-40], and mostly for a high resolution images the matrices are very huge and the can be easily used to hold secret message or to hold another image which carries the secret message.
Steganography is the process of hiding data into another data as shown in figure (1), in this study we will insert (hide) a secret message in a color image, then the holding color image will be hidden in another bigger color image.

![Figure (1): Data steganography](image)

Steganography is very useful and applicable process for the following reasons [3], [4], [5]:

a. Personal data are private and sometimes are very confidential.

b. Data are very sensitive

c. Confidential data and trade secrets

d. Misuse of data is not acceptable.

e. Data does not bear unintentional damage, or human error and accidental deletion.

f. Data should not be exposed monetary and blackmail purposes

g. Data does not deal with hiding traces of crime

Any data hiding technique shall express certain features such as:

- Capacity, which refers to the amount of data that can be hidden in cover medium [3].

- Security, the data hiding method should provide security such that only the intended user can gain access to it. In other words, it refers to the inability of un-authorized user to detect hidden information. This is very crucial to protect the confidentiality and sensitivity of information being sent [3], [4].

- Robustness, which refers to the amount of data that can be hidden without showing any negative effects and destroying hidden information [1]. In other words making it difficult to distinguish them with the naked eye
Perceptibility, the data hiding method should hide data in such a manner that the original covering data and the hidden data are perceptually indistinguishable [5].

True color image is a three dimensional matrix [1], [2], the first dimension is reserved for the red color; the second dimension is reserved for the green color, while the third one is reserved for the blue color.

High intensity color image usually has a huge size, thus it can be used as a good medium to cover and hold the secret message and to make it very difficult to distinguish them with the naked eye. So using a huge color image as a covering media will reduce mean square error (MSE) and will maximize peak signal to noise ratio (PSNR) [6], [7], avoiding destroying covering image.

2- Proposed Methodology

The proposed methodology is based on least significant bit (LSB) method of data hiding [8]. This method is very simple to implement and provides a high PSNR value and a low MSE value, but it is not secure because of the known procedures [9], [10].

LSB method sometimes added minor changes to covering image due to the following reasons [11], [12]:

a. If the least bit of the byte in the covering image equal the corresponding bit of the message byte then the byte of the covering image will not change.

b. If the least bit of the byte in the covering image equal 1 and the corresponding bit of the message byte equal 0 then the byte of the covering image will be reduced by 1 and still closed to the original value.

c. If the least bit of the byte in the covering image equal 0 and the corresponding bit of the message byte equal 1 then the byte of the covering image will be increased by 1 and still closed to the original value.

Each byte from data to be hidden requires 8 bytes from the covering image, so the maximum size of the hidden data must not exceed the size of the holding image divided by 8 [1].

Table (1) shows how to hide 'Ziad' in covering image bytes:

Table (1): LSB example

<table>
<thead>
<tr>
<th>Message= Ziad = 90 105 97 100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary= 01011010 01101001 01100001 01100100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Red pixel</th>
<th>Decimal</th>
<th>Covering binary</th>
<th>Holding binary</th>
<th>Decimal</th>
<th>Red pixel</th>
<th>Decimal</th>
<th>Covering binary</th>
<th>Holding binary</th>
<th>Decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>249</td>
<td>1111001</td>
<td>1111000</td>
<td>248</td>
<td>17</td>
<td>249</td>
<td>1111001</td>
<td>1111000</td>
<td>248</td>
</tr>
<tr>
<td>2</td>
<td>249</td>
<td>1111001</td>
<td>1111001</td>
<td>249</td>
<td>18</td>
<td>249</td>
<td>1111001</td>
<td>1111001</td>
<td>249</td>
</tr>
<tr>
<td>3</td>
<td>249</td>
<td>1111001</td>
<td>1111000</td>
<td>248</td>
<td>19</td>
<td>249</td>
<td>1111001</td>
<td>1111001</td>
<td>249</td>
</tr>
<tr>
<td>4</td>
<td>249</td>
<td>1111001</td>
<td>1111001</td>
<td>249</td>
<td>20</td>
<td>249</td>
<td>1111001</td>
<td>1111000</td>
<td>248</td>
</tr>
<tr>
<td>5</td>
<td>249</td>
<td>1111001</td>
<td>1111001</td>
<td>249</td>
<td>21</td>
<td>249</td>
<td>1111001</td>
<td>1111000</td>
<td>248</td>
</tr>
<tr>
<td>6</td>
<td>249</td>
<td>1111001</td>
<td>1111000</td>
<td>248</td>
<td>22</td>
<td>249</td>
<td>1111001</td>
<td>1111000</td>
<td>248</td>
</tr>
</tbody>
</table>
The proposed methodology can be applied implementing the following phases:

1) Hiding message in color image (a1).

   This phase can be implemented applying the following steps:
   
   A. Get the original covering image.
   
   B. Get the message to be hidden.
   
   C. Convert each of the image and the message into one column arrays.
   
   D. Select a reference (position) in the image array where to start hiding (save this reference let us say ref1).
   
   E. Apply LSB method starting from ref1 and back toward the beginning to hide each byte of the message.
   
   F. Get the decimal values of the image.
   
   G. Save message length
   
   H. Save image a1 size.
   
   I. Reshape the resulting image back to 3D matrix and save the holding image (a1).

2) Hiding a1 in another bigger color image.

   This phase can be implemented applying the following steps:
   
   A. Get the new bigger covering image.
   
   B. Get the image a1.
   
   C. Convert each of the two images into one column arrays.
   
   D. Select a reference (position) in the image array where to start hiding a1 (save this reference let us say ref2).
   
   E. Apply LSB method starting from ref2 and back toward the beginning to hide each byte of the image a1.
F. Get the decimal values of the holding image (a2).

G. Reshape the resulting image back to 3D matrix and save the holding image (a2).

3) Extracting the image a1 from image a2.

This phase can be implemented applying the following steps:

A. Load image a2.

B. Load reference ref2.

C. Load the size of image a1.

D. Convert image a2 to one column array and convert the values to binary.

E. For each value in a1 take the LSB of 8 bytes.

F. Repeat the previous step for a number of bytes equal a1 size.

G. Reshape a1 to the original size.

4) Extracting message from image a1.

This phase can be implemented applying the following steps:

H. Load image a1.

I. Load reference ref1.

J. Load the size of message.

K. Convert image a1 to one column array and convert the values to binary.

L. For each value in message take the LSB of 8 bytes.

M. Repeat the previous step for a number of bytes equal message size.

The reference contains (k1, k2, k3), where k1 is the row where to start, k2 is the column number, and k3 is the color number, this reference must be converted to location using the following formula:

\[ \text{ref} = (k3 - 1) \times n1 \times n2 + (k1 - 1) \times n2 + k2 \]

Where: n1, n2, and n3 are image matrix dimensions.

3- Experimental Implementation

In this part we will experimentally investigate the above mentioned phases, in order to get some parameters which can be used to judge the advantages of the proposed methodology. Figures (2) and (3) show the original image and a holding image after hiding a message of 100 characters length, and here we can see that the holding image is very closed to original covering image and the changes cannot be noticed by human eyes.
Now we will perform some experiments to measure some parameters of the process of steganography.

Experiment 1:

Hiding message (100 character length) in various image.

The results of this experiment are shown in table (2):
Table (2) : Experiment 1 results

<table>
<thead>
<tr>
<th>Image size(byte)</th>
<th>Hiding time(seconds)</th>
<th>Extraction time(seconds)</th>
<th>PSNR</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>284x160x3=136320</td>
<td>0.0240</td>
<td>0.2040</td>
<td>170.4450</td>
<td>0.0026</td>
</tr>
<tr>
<td>284x260x3=221520</td>
<td>0.0290</td>
<td>0.2050</td>
<td>170.4450</td>
<td>0.0026</td>
</tr>
<tr>
<td>320x207x3=198720</td>
<td>0.0290</td>
<td>0.2090</td>
<td>171.4452</td>
<td>0.0023</td>
</tr>
<tr>
<td>314x280x3=263760</td>
<td>0.0295</td>
<td>0.2120</td>
<td>171.4921</td>
<td>0.0023</td>
</tr>
<tr>
<td>450x351x3=473850</td>
<td>0.0270</td>
<td>0.2110</td>
<td>171.4595</td>
<td>0.0023</td>
</tr>
<tr>
<td>477x268x3=383508</td>
<td>0.0260</td>
<td>0.2300</td>
<td>171.4810</td>
<td>0.0023</td>
</tr>
<tr>
<td>550x367x3=605550</td>
<td>0.0260</td>
<td>0.2140</td>
<td>169.2277</td>
<td>0.0023</td>
</tr>
<tr>
<td>560x315x3=529200</td>
<td>0.0280</td>
<td>0.2790</td>
<td>171.4810</td>
<td>0.0023</td>
</tr>
<tr>
<td>590x310x3=548700</td>
<td>0.0240</td>
<td>0.2320</td>
<td>171.2701</td>
<td>0.0023</td>
</tr>
<tr>
<td>600x340x3=612000</td>
<td>0.0270</td>
<td>0.2130</td>
<td>171.4735</td>
<td>0.0023</td>
</tr>
<tr>
<td>660x330x3=653400</td>
<td>0.0280</td>
<td>0.2330</td>
<td>171.5131</td>
<td>0.0023</td>
</tr>
<tr>
<td>1271x2048x3=7809024</td>
<td>0.0280</td>
<td>0.2520</td>
<td>171.4899</td>
<td>0.0023</td>
</tr>
<tr>
<td>4500x3000x3=40500000</td>
<td>0.0280</td>
<td>0.2150</td>
<td>171.4661</td>
<td>0.0023</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>0.0272</strong></td>
<td><strong>0.2238</strong></td>
<td><strong>171.1299</strong></td>
<td><strong>0.0024</strong></td>
</tr>
</tbody>
</table>

Here for the covering images with size range 133 Kbytes to 39551 Kbytes the average hiding time was equal 0.0272 seconds, the average extraction time was equal 0.2238 seconds and the average PSNR was equal 171.1299 which are good hiding parameters.

Experiment 2:

Hiding holding image with 560x315x3 bytes in other bigger images:

Figures (4), (5), and (6) show the images: Image which holds the message, original covering image, image which holds image.

Figure (4): Image holding the message
Figure (5): Original covering image

Figure (6): Image which holds another image
Table (3) shows the results of this experiment:

Table (3): Experiment 2 results

<table>
<thead>
<tr>
<th>Image size(byte)</th>
<th>Hiding time(seconds)</th>
<th>Extraction time(seconds)</th>
<th>PSNR</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>4500x3000x3=40500000</td>
<td>0.3410</td>
<td>0.2330</td>
<td>120.5734</td>
<td>0.3773</td>
</tr>
<tr>
<td>1500x1102x3=4959000</td>
<td>0.3200</td>
<td>0.2470</td>
<td>120.9807</td>
<td>0.3622</td>
</tr>
<tr>
<td>1300x957x3=3732300</td>
<td>0.3370</td>
<td>0.2300</td>
<td>121.1922</td>
<td>0.3546</td>
</tr>
<tr>
<td>1271x2048x3=7809024</td>
<td>0.3310</td>
<td>0.2140</td>
<td>120.5538</td>
<td>0.3780</td>
</tr>
<tr>
<td>1280x720x3=2764800</td>
<td>0.3270</td>
<td>0.2100</td>
<td>120.3233</td>
<td>0.3868</td>
</tr>
<tr>
<td>Average</td>
<td><strong>0.3312</strong></td>
<td><strong>0.2268</strong></td>
<td><strong>120.7247</strong></td>
<td><strong>0.3718</strong></td>
</tr>
</tbody>
</table>

From table (3) we can see that the average hiding time was equal 0.3312 seconds, the average extraction time was equal 0.2268 seconds and the average PSNR was equal 120.7247 which are good hiding and extracting parameters.

From tables (3) and (4) we can see that the total average hiding time equal 0.3584 seconds and the total extraction time equal 0.4506 with a high average of PSNR values for the phase of hiding a message into an image, and hiding an image into another image.

Conclusion

A methodology based on LSB method was proposed, implemented and tested.

The proposed methodology increases the security level of LSB method by using secret references ref1 and ref2 as a private keys. The experimental results showed that the obtained values for hiding time, extracting time, PSNR and MSE are acceptable and optimal.

References

10. Emam, M. M., Aly, A. A., & Omara, F. A. An Improved Image Steganography Method Based on LSB Technique with

© 2019, IJCSMC All Rights Reserved


35. AL-Azzeh Jamil, Review of Methods of Distributed Synchronization of Parallel Processes in Matrix VLSI Systems,

