Genetic Algorithm in Image Inserting with Modified Least Significant Bit Method to Find The Best MSE Value

S F Ramadhan\textsuperscript{1}, P Sihombing\textsuperscript{1} and Sutarman\textsuperscript{2}

\textsuperscript{1} Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Medan, Indonesia
\textsuperscript{2} Department of Mathematics and Natural Sciences, Universitas Sumatera Utara, Medan, Indonesia.

*Email: satriarmdhn@yahoo.com, poltak@usu.ac.id, sutarman@usu.ac.id

Abstract. Inserting data into digital media tends to change the large size of the media cover file, this is because the inserted pixel has a different size from the insertion value, i.e. with a larger or smaller value, for example a pixel value of 10 is inserted with larger or smaller data from the value 10. So that the pixel value is getting further from the value of 10, this causes the value of Mean Squared Error (MSE) of the media of the cover (cover) to become larger. In this study, the selection of pixels in the image is in accordance with the value of the insertion data as the insertion place with the Genetic algorithm (GA). Insertion of data on selected pixels using the GA algorithm is performed using the Modified Least Significant Bit (MLSB) algorithm at the position of the least significant bits. To find out the reliability of the proposed algorithm, the data insertion experiment was carried out by comparing the conventional LSB algorithm with the proposed algorithm where the results obtained a significant MSE value, i.e. the average MSE value of the LSB algorithm was 20.54 and the proposed algorithm was 7.12.

1. Introduction

Digital formats with high redundant bits can be used to hide secret messages, so the implications of replaced bits are unnoticeable. Steganography can be done trough Text, Image, Audio-Video and Protocols as Shown in Figure 1. Using Text file as a cover medium is quite small in size and can have small amount of message and audio-video is very large to transmit over the network. So the Image has medium size and sufficient enough to hide secret messages, so images are quite popular in steganography.

![Image with text](text.png)

**Figure 1.** Text and Image.
Images are presented as grid, where each individual point is called a 'pixel'. Pixels are displayed horizontally, row by row. In a color scheme, the number of bits is known as the „bit depth” and this basically refers to the number of bits assigned to each pixel. The smallest bit depth is represented by 8 bits. This, in any given pixel, the number of different shades of red, green, and blue can reach 256 ($2^8$) that adding up to more than 16 million ($256^3$) combinations that finally result in more than 16 million colors.

![Figure 2. Matrix representation of RGB image](image)

Genetic algorithm is an optimization and search technique based on the principles of genetic science and natural selection where the algorithm is inspired by Darwin’s evolutionary biology. In genetic algorithms, selection, mutation, and crossover are the main things. Genetic algorithms are different from conventional search techniques because they start with a set of initializations of random solutions called populations. The population consists of chromosomes called genes. The value of genes can be numeric, binary, symbol or character, depending on the problem to be solved.

2. Study of Literature

2.1. Modified Least Significant Bit (MLSB) Algorithm
Modified Least Significant Bit (MLSB) Algorithm is used to encode insertion data. This algorithm uses manipulation of several levels of data insertion bits using Control Symbol before encoding the data (Zaher, 2011). Modify the message with the MLSB algorithm where the message bits that should be 1 character have a value of 8 bits ASCII code will be modified to 5 bits. In this algorithm the characters and numbers are represented in 5 bits which will then be inserted into digital media using the Least Significant Bit (LSB) technique.

| Hex Representation | Operation           |
|--------------------|---------------------|
| 1 Bh               | Define Small Letter |
| 1 Ch               | Define Capital Letter|
| 1 Dh               | Define Space        |
| 1 Eh               | Define Number       |
| 1 Fh               | Define end of text  |

To describe this work we will use a 24-bits color image as over image. The hiding operation will be done on the basis of bit wise message hiding concept. Where the message's characters will be treated with their 8-bits ASCII codes (Kochhar, 2008), then the codes will be converted to 5-bits code using our algorithm, and then the 5-bits code will be hidden in the image using LSB method. In the cover image,
each pixel is treated accordingly with its Red, Green, Blue values. This will generate a very small difference, for human eye. In our algorithm we use just 5 bits to represent characters and numbers. To perform that we study characters ASCII representation to eliminate redundant bits.

According to our study if we look for ASCII representation we can note that, small letter character representation (hexadecimal) in the range between 61h to 7Ah, and capital letter character in the range between 41h to 5Ah. Finding relationship between them we can note that: For small letter the last 4 bits can be just either 6 or 7. In case of 6 the first 4 bits of characters is incremented from 1h to Fh to represents characters from 'a' to 'o' as shown in Table 1. Then the last four bits (with value = 6) are incremented to be 7 and the first 4 bits again incremented from 0h to Ah to represents characters from 'p' to 'z' as shown in Table 2. For capital letter the last 4 bits can be just either 4 or 5. In case of 4 the first 4 bits of characters is incremented from 1h to Fh to represents characters from 'A' to 'O'. Then the last four bits (with value = 4) are incremented to be 5 and the first 4 bits again incremented from 1h to Ah to represents characters from 'P' to 'Z' (see Table 3).

We can note that there is a repeated sequence between capital and small letters representation. In the case of small letter if we take the binary representation of last four bits, it will be "0110" or "0111". For capital letter the last four bits will be "0100" or "0101". As we can see the last three bits are identical so we can discard them. Accordingly we decide to use 5 bits to represent character where the first 4 bits representation will be the same as the first 4 bits representation in ASCII. The fifth bit can be 0 or 1. If 0 then we represent the first sequence from 01h to 0Fh, in case of small letter characters from 'a' to 'o'. Else if the fifth bit equal 1; we represent the second sequence from 0h to Ah, in case of small letter characters from 'p' to 'z'. But how we can distinguish between small letter, capital letter, numbers, and space. To perform that we use control symbol that can define the state of next character. Our control symbols are also 5-bits. In 5-bits representation when the fifth bit is one the first 4 bits will range from 01h to 0Ah. As we can note the range from Bh to Fh is not used. So we use the successive from Bh to Fh as control symbol as shown in Table 4.

Flowchart of our text hide algorithm shown in Figure 1. According to Table 4 and Figure 1, our algorithm deals with five cases:
1. Small letter:
   If we want to represent small letter first we must put the first 5-bits to be 1Bh then include our converted (5-bits) small letter text.
2. Capital letter:
   If we want to use capital letter first we must put the first 5-bits to be 1Ch then include our text.
3. Space:
   In the case of space we put the 5-bits to be 1Dh instead of 20h (ASCII representation of space) since space it is not in the range of ASCII characters.
4. Numbers:
   In case of numbers ASCII represents number from 0 to 9 as 30h to 39h. if we discard the last four bits we can represent numbers using just 4 bits. In our case the fifth bit will be 0. In our process to indicate that we deal with number then first we must use the control symbol 1Eh (first 5 bits) then write the number (each digit range from 0 to 9).
5. End of text:
   Finally to define the end of the text we use the control symbol 1Fh. In some cases we can ignore this control symbol. In this algorithm before each new text format (capital, small, or numbers) the first 5-bits must be control symbol then we include the converted text sequence (Zaher, M. A 2011).

2.2. Substitution Techniques and RGB Color Model
Substitution technique is a process of replacing LSB’s of cover image data by message data bits. When we are using 24-bit image then we can embed 8-bits of audio message into each pixel data. Replacements of bits take in place 3+3+2 bits in Red, Green and Blue component of a pixel respectively. For example given vector shows a pixel of 24 bit color image, using 3 bytes of memory. (10100111 10101001 11011000) When an 8 bit binary (10010011) inserted, resultant will be (10100100 10101100 11011011). The following diagram shows, the LSBs modified in proposed algorithm. 3 bits of each Red and Green and 2 bits of Blue are modified. The secret data would be put in these LSBs only.
2.3. Genetic Algorithm
In general, the process cycle of genetic algorithms starts with a number of ‘n’ chromosomes in a random genetic population, evaluating the fitness of each chromosome in the population. Then the selection process is carried out by selecting two parent chromosomes from the population based on their respective fitness, the better the fitness value, the greater the chance to be chosen. With crossover probability, both parents are crossed to produce offspring. If a crossover does not occur, the resulting offspring is the product of the parent doubling. After crossover, the next process is mutation with a predetermined mutation probability. Furthermore, new offspring is placed in the new population. The new offspring is used to repeat the next process. If the final conditions are as expected, the process will be stopped, and will be returned to the best solution in the population. The process cycle of the genetic algorithm can be seen as in Figure 3.

![Figure 3. Genetic Algorithm Cycle Chart.](image)

3. Finding and Discussions

3.1. The Algorithm
In this study the insertion of text data into a digital image file with the Modified Least Significant Bit (MLSB) algorithm. The problem to be solved is how to choose the image pixel as a place for inserting text data into the image file in order to obtain the best or the smallest MSE value. In this study Genetic algorithm is used to select the appropriate pixels in the image file where text data is inserted so that the stego image is obtained from the insertion. Steps The proposed algorithm for image insertion is as in Figure 4.

![Figure 4. Proposed Algorithm Chart.](image)
3.2. Data Used
The test was carried out using five digital images of different sizes where each of the images will be inserted with five different text files. The parameter to be calculated is the MSE value.

3.3. Results and Discussion
From these tests the results obtained as shown in Table 2 and Table 3.

Table 2. Results of LSB Steganography Insertion.

| No | File Cover | Size (byte) | Embed File | Num. | Char | Parameter |
|----|------------|------------|------------|------|------|-----------|
|    |            |            |            |      |      | MSE       |
|    |            |            |            |      |      | PSNR      |
| 1. | Pic01.jpg  | 110.020    | Message1.txt | 29   |      | 5.63      |
|    | Pic01.jpg  | 110.020    | Message2.txt | 73   |      | 15.02     |
|    | Pic01.jpg  | 110.020    | Message3.txt | 158  |      | 29.86     |
|    | Pic01.jpg  | 110.020    | Message4.txt | 185  |      | 34.94     |
|    | Pic01.jpg  | 110.020    | Message5.txt | 228  |      | 47.32     |
| 2. | Pic02.jpg  | 148.525    | Message1.txt | 29   |      | 0.33      |
|    | Pic02.jpg  | 148.525    | Message2.txt | 73   |      | 1.16      |
|    | Pic02.jpg  | 148.525    | Message3.txt | 158  |      | 1.93      |
|    | Pic02.jpg  | 148.525    | Message4.txt | 185  |      | 2.24      |
|    | Pic02.jpg  | 148.525    | Message5.txt | 228  |      | 4.47      |
| 3. | Pic03.jpg  | 194.710    | Message1.txt | 29   |      | 0.029     |
|    | Pic03.jpg  | 194.710    | Message2.txt | 73   |      | 1.41      |
|    | Pic03.jpg  | 194.710    | Message3.txt | 158  |      | 1.82      |
|    | Pic03.jpg  | 194.710    | Message4.txt | 185  |      | 2.33      |
|    | Pic03.jpg  | 194.710    | Message5.txt | 228  |      | 5.28      |
| 4. | Pic04.jpg  | 138.358    | Message1.txt | 29   |      | 2.16      |
|    | Pic04.jpg  | 138.358    | Message2.txt | 73   |      | 6.86      |
|    | Pic04.jpg  | 138.358    | Message3.txt | 158  |      | 11.46     |
|    | Pic04.jpg  | 138.358    | Message4.txt | 185  |      | 13.41     |
|    | Pic04.jpg  | 138.358    | Message5.txt | 228  |      | 22.64     |
| 5. | Pic05.jpg  | 223.932    | Message1.txt | 29   |      | 2.30      |
|    | Pic05.jpg  | 223.932    | Message2.txt | 73   |      | 7.68      |
|    | Pic05.jpg  | 223.932    | Message3.txt | 158  |      | 12.26     |
|    | Pic05.jpg  | 223.932    | Message4.txt | 185  |      | 14.34     |
|    | Pic05.jpg  | 223.932    | Message5.txt | 228  |      | 25.55     |
The results of testing the insertion of text data into an image file can be seen as in Table 3.

**Table 3.** Results of the Insertion of Proposed Algorithm Steganography (GA-MLSB).

| No | File Cover | Size (byte) | Embed File | Num. Char | Parameter | MSE     | PSNR  |
|----|------------|-------------|------------|-----------|-----------|---------|-------|
| 1. | Pic01.jpg  | 110.020     | Message1.txt | 29        | 0.000035  | 92.65   |
|    | Pic01.jpg  | 110.020     | Message2.txt | 73        | 0.000084  | 88.84   |
|    | Pic01.jpg  | 110.020     | Message3.txt | 158       | 0.000165  | 85.93   |
|    | Pic01.jpg  | 110.020     | Message4.txt | 185       | 0.000197  | 85.17   |
|    | Pic01.jpg  | 110.020     | Message5.txt | 228       | 0.000281  | 83.63   |
| 2. | Pic02.jpg  | 148.525     | Message1.txt | 29        | 0.000024  | 94.31   |
|    | Pic02.jpg  | 148.525     | Message2.txt | 73        | 0.000052  | 90.92   |
|    | Pic02.jpg  | 148.525     | Message3.txt | 158       | 0.000097  | 88.22   |
|    | Pic02.jpg  | 148.525     | Message4.txt | 185       | 0.000098  | 88.19   |
|    | Pic02.jpg  | 148.525     | Message5.txt | 228       | 0.000134  | 86.84   |
| 3. | Pic03.jpg  | 194.710     | Message1.txt | 29        | 0.000034  | 92.72   |
|    | Pic03.jpg  | 194.710     | Message2.txt | 73        | 0.000089  | 88.60   |
|    | Pic03.jpg  | 194.710     | Message3.txt | 158       | 0.000155  | 86.21   |
|    | Pic03.jpg  | 194.710     | Message4.txt | 185       | 0.000182  | 85.51   |
|    | Pic03.jpg  | 194.710     | Message5.txt | 228       | 0.000216  | 84.77   |
| 4. | Pic04.jpg  | 138.358     | Message1.txt | 29        | 0.000016  | 95.87   |
|    | Pic04.jpg  | 138.358     | Message2.txt | 73        | 0.000015  | 91.94   |
|    | Pic04.jpg  | 138.358     | Message3.txt | 158       | 0.000075  | 89.58   |
|    | Pic04.jpg  | 138.358     | Message4.txt | 185       | 0.000098  | 88.17   |
|    | Pic04.jpg  | 138.358     | Message5.txt | 228       | 0.000120  | 87.30   |
| 5. | Pic05.jpg  | 223.932     | Message1.txt | 29        | 0.000014  | 96.65   |
|    | Pic05.jpg  | 223.932     | Message2.txt | 73        | 0.000039  | 92.15   |
|    | Pic05.jpg  | 223.932     | Message3.txt | 158       | 0.000101  | 88.05   |
|    | Pic05.jpg  | 223.932     | Message4.txt | 185       | 0.000084  | 86.66   |
|    | Pic05.jpg  | 223.932     | Message5.txt | 228       | 0.000111  | 87.66   |

Based on the results obtained in table 2 and table 3, then MSE, PSNR comparison data can be formed each sample image in accordance with the embed inserted. the results of this comparison are shown in Table 4 Table 5, Table 6, Table 7 and Table 8 below.
Table 4. Comparison in the insertion of Pesan1.txt

| No | File Cover | MSE LSB | GA-MLSB |
|----|------------|--------|---------|
| 1  | Pic01.jpg  | 5.63   | 0.00035 |
| 2  | Pic02.jpg  | 0.33   | 0.00024 |
| 3  | Pic03.jpg  | 0.029  | 0.00034 |
| 4  | Pic04.jpg  | 2.16   | 0.00016 |
| 5  | Pic05.jpg  | 2.30   | 0.00014 |

From Table 4 a comparison chart can be made of MSE Value of LSB algorithm with GA-MLSB algorithm for Pesan1.txt insertion as in Figure 5.

Figure 5. Comparison Graph of MSE LSB Algorithm with GA-MLSB Pesan1.txt

Table 5. Comparison in Insertion of Pesan2.txt

| No | File Cover | MSE LSB | GA-MLSB |
|----|------------|--------|---------|
| 1  | Pic01.jpg  | 5.63   | 0.00084 |
| 2  | Pic02.jpg  | 1.16   | 0.00052 |
| 3  | Pic03.jpg  | 1.41   | 0.00089 |
| 4  | Pic04.jpg  | 6.86   | 0.00015 |
| 5  | Pic05.jpg  | 7.68   | 0.00039 |

From Table 5, above a graph is made comparing the MSE value of the LSB algorithm with the GA-MLSB algorithm for the insertion of Pesan2.txt as in Figure 6.

Figure 6. Comparison Graph of MSE LSB Algorithm with GA-M LSB Insertion of Pesan2.txt
Table 6. Comparison in Insertion of the Pesan3.txt

| No | File Cover | MSE         |       |
|----|------------|-------------|-------|
|    |            | LSB         | GA-MLSB |
| 1  | Pic01.jpg  | 29.86       | 0.000165 |
| 2  | Pic02.jpg  | 1.93        | 0.000097 |
| 3  | Pic03.jpg  | 1.82        | 0.000155 |
| 4  | Pic04.jpg  | 11.46       | 0.000075 |
| 5  | Pic05.jpg  | 12.26       | 0.000101 |

From Table 6 a comparison chart of MSE Value of LSB algorithm and GA-MLSB algorithm for insertion Pesan4.txt is shown in Figure 7.

![Figure 7. Comparison Graph of MSE LSB Algorithm with GA-M LSB Insertion of Pesan4.txt](image)

Table 7. Comparison in the insertion of message4.txt

| No | File Cover | MSE         |       |
|----|------------|-------------|-------|
|    |            | LSB         | GA-MLSB |
| 1  | Pic01.jpg  | 34.94       | 0.000197 |
| 2  | Pic02.jpg  | 2.24        | 0.000098 |
| 3  | Pic03.jpg  | 2.33        | 0.000182 |
| 4  | Pic04.jpg  | 13.41       | 0.000098 |
| 5  | Pic05.jpg  | 14.34       | 0.000084 |

From Table 7. above a graph is made comparing the MSE value of the LSB algorithm with the GA-MLSB algorithm for the insertion of Pesan4.txt as in Figure 8.

![Figure 8. Comparison Graph of MSE LSB Algorithm with GA-M LSB Insertion of Pesan4.txt](image)
Table 8. Comparison in the insertion of Pesan5.txt

| No | File Cover | MSE          |
|----|------------|--------------|
|    |            | LSB | GA-MLSB     |
| 1. | Pic01.jpg  | 47.32| 0.000281    |
| 2. | Pic02.jpg  | 4.47 | 0.000134    |
| 3. | Pic03.jpg  | 5.28 | 0.000216    |
| 4. | Pic04.jpg  | 22.64| 0.000120    |
| 5. | Pic05.jpg  | 25.55| 0.000111    |

From Table 8. above a graph is made comparing the MSE value of the LSB algorithm with the GA-MLSB algorithm for the insertion of Pesan5.txt as in Figure 9.

![Comparison Graph of MSE LSB Algorithm with GA-MLSB insertion of Pesan5.txt](image)

**Figure 9.** Comparison Graph of MSE LSB Algorithm with GA-MLSB Insertion of Pesan5.txt

4. Conclusion

In the graph above it can be seen that the GA-MLSB algorithm is superior in comparison to MSE parameters. This is because the genetic algorithm can select pixels that are suitable for insertion with the Genetic algorithm and the MLSB algorithm can compress the insertion binaries into 5 bits per character and then insert message bits at the Least Significant Bit position. Changing the bits in the Least Significant Bit position does not have a large effect on the embedded image so that the MSE value is not too large.

From the tables and graphs presented above conclude that the GA-MLSB algorithm selects image pixels that are suitable for insertion data and obtains messages into five bits per character so that the number of image pixels needed to contain the message becomes less. This is evidenced by the MSE value of the image of the insertion is smaller than the image of the result of the conventional LSB algorithm insertion.

References

[1] Aliwa, M. B., Tarek E., T, Fahmy & Nasr, M. E. N. 2016. Robust Digital Watermarking Based Falling-off-Boundary in Corners Board-MSB-6 Gray Scale Images. International Journal of Computer Science and Network Security, VOL.9 No.8, August 2016. Faculty of Computers & Information Sciences-Ain Shams University, Egypt.

[2] Arya, A. 2017. Security Enhancement Using Triple DES Algorithm. International Journal of Computer Sciences and Mobile Computing, Vol. 6, Issue 4, April 2017.
[3] Chugh, G. 2013. Image Steganography Techniques: A Review Article. Acta Technica Corviniensis – Bulletin of Engineering Tome VI, Fascicule 3, July 2013.
[4] Ciptayani, P. I., Mahmudy, W. F. & Widodo, A. W. 2015. Penerapan Algoritma Genetika Untuk Kompresi Citra Fraktal. Jurnal Program Studi Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Brawijaya.
[5] Dilawar, A. 2014. Image Steganography: For Hiding Audio Messages Within Grayscale Images Using LSB, DCT And AES Algorithm. Thesis. International Islamic University, Sector H-10: Islamabad.
[6] Gonzalez, R.C., Woods, R.E. 2008. Digital Image Processing. 3rd Edition. Pearson Education, Inc: Upper Saddle River, New Jersey.
[7] Halim, S. A. & Sani, M. F. A. 2015. Embedding Using Spread Spectrum Image Steganography with GF(2^m). Proceedings of the 6th IMT-GT Conference on Mathematics, Statistics and its Applications (ICMSTA2015) Universiti Tunku Abdul Rahman, Kuala Lumpur, Malaysia.
[8] Husain, M.P. & Rafat, K.F. 2016. Enhanced Audio LSB Steganography for Secure Communication. International Journal of Advanced Computer Science and Aplications, Vol. 7, No. 1: 340-347.
[9] Jatmoko, C. Handoko, B., Sari, C. A. & Setiadi, I. M. 2018. Uji Performa Penyisipan Pesan Dengan Metode Lsb Dan Msb Pada Citra Digital Untuk Keamanan Komunikasi. Jurnal Dinamika Rekayasa Vol. 14 No. 1 2018.
[10] Jiansheng, M., Li Sukang, L. & Tan Xiaomei, T. 2017. A Digital Watermarking Algorithm Based On DCT and DWT. Proceedings of the 2017 International Symposium on Web Information Systems and Applications (WISA’09) Nanchang, P. R. China, May 22-24, 2017.
[11] Megalingam, R. K., Nair, M. M, Srikumar, R., Balasubramanian, V. K. & Sarma, VSV. 2015. A Comparative Study on Performance of Novel, Robust Spatial Domain Digital Image Watermarking with DCT Based Watermarking. International Journal of Computer Theory and Engineering, Vol. 2, No. 4, August, 2015.
[12] Rachmansyah, G., Mahmudy, W. F. & Perdana, R. S. 2016. Optimasi Model Segmentasi Citra Metode Fuzzy Divergence Pada Citra Luka Kronis Menggunakan Algoritma Genetika. Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) Vol. 3, No. 1, Maret 2016, Universitas Brawijaya, Malang.
[13] Sivanandam, S., N. & Deepa, S., N. 2008. Introduction to Genetic Algorithms. Springer-Verlag: Berlin.
[14] Singh, S. & Siddiqui, T.J. 2012. A Security Enhanced Robust Steganography Algorithm for Data Hiding. International Journal of Computer Science Issues. Volume 9, Issue 3, No. 1, May 2012: 131-139.
[15] Sugiono, et al. 2018. Watermarking Pada File Audio PCM WAVE Dengan Metode Echo Data Hiding. Konferensi Nasional Sistem dan Informatika 2008; Bali, Nopember 15, 2018.
[16] Sukrisno & Utami, E. 2017. Implementasi Steganografi Teknik Eof Dengan Gabungan Enkripsi Rijndael, Shift Cipher Dan Fungsi Hash MD5. Seminar Nasional Teknologi 2007 Yogyakarta, 24 November 2007.
[17] Umbarkar, A., J. & Sheth, P., D. 2015. Crossover Operators in Genetic Algorithms: A Review. ICTACT Journal on Soft Computing 6(1): 1083 – 1092.
[18] Winarso, Sri. M.E. Digital Watermarking Menggunakan Metode Discrete Cosine Transform. Jurnal Teknologi Informasi-AITI. Februari 2017.
[19] Zaher, M.A. 2011. Modified Least Significant Bit (MLSB). Computer and Information Science. Vol. 4, No. 1, Januari 2011: 60-67
[20] Zhang, D., Mabu, S. & Hirasawa, K. 2010. Noise Reduction using Genetic Algorithm Based PCNN Method. Proceedings of The IEEE International Conference on Systems, Man and Cybernetics, pp. 2627 – 2633.