Scientific Journal of Informatics
Vol. 8, No. 2, November 2021

p-ISSN 2407-7658 http://journal.unnes.ac.id/nju/index.php/sji e-ISSN 2460-0040

Increasing Message Capacity in Images Using Advanced Least Significant Bit and Image Scaling

Affan Fadlil[1*], Budi Prasetiyo[2], Alamsyah[3]
[1,2,3]Computer Science Department, Faculty of Mathematics and Natural Sciences,
Universitas Negeri Semarang, Indonesia

Abstract.
Purpose: Steganography is the science of writing hidden or hiding messages so that apart from the sender and the recipient, no one can know or realize that a message is hidden. This paper aims to analyze the method of advanced LSB to increase message capacity.
Methods/Study design/approach: The steganography technique advanced LSB algorithm develops pre-existing steganographic algorithms such as LSB by utilizing a range of media pixel values cover (images that are used as media to hide messages) with different insertion rules from LSB. Image scaling in digital image processing is known as resampling. Resampling is a mathematical technique used to produce a new image from the previous image with different pixel size, often called interpolation. Increasing the pixel size of the previous image is called upsampling and in this study we will only use twice the image magnification.
Result/Findings: The results of each test method using advanced LSB without image scaling and advanced LSB using image scaling were compared to obtain detailed comparison results of each method.
Novelty/Originality/Value: Advanced LSB and image scaling in this study can increase the message capacity three times compared to only using the advanced LSB method without image scaling. It depends on the image pixels used.

Keywords: Steganography, advanced LSB, image scaling, message capacity.
Received December 2020 / **Revised** April 2021 / **Accepted** November 2021

INTRODUCTION

Steganography is the art of hiding that communication occurs by hiding information in other information [3]. Steganography abuse often occurs recently, one of which is used to insert a certain message based on crime. Therefore, steganalysis is needed to control the misuse of steganography [4]. Two important things in steganography are the payload capacity and hiding information (imperceptibility) [6]. Use of advanced LSB and image scaling to increase the capacity of messages that can be inserted.

The algorithm for the advanced LSB steganography technique develops the existing steganography algorithms such as LSB by utilizing the range of pixel values for the cover media (images that are used as media to hide messages). The higher the pixel value, the more message bits we hide, and vice versa if the smaller the pixel value the fewer message bits we hide. Advanced LSB, as already mentioned, is a message hiding method that has the development of LSB where the hidden or inserted message in an image that has more storage or the capacity of the message is inserted. The quality of the stego image is better.

According to Nugraha [5] Steganography was carried out using the Redundant Pattern Encoding method and digital images as a container for hidden text carriers. The advantage of this method is that it is resistant to cropping and compression when processing the cover media. Text insertion in the redundant pattern encoding method is done by inserting the text on noise or parts that are not noticed or are not visually visible in the file. The LSB used has been modified according to the algorithm [2]. The message capacity inserted in the image is more than the message inserted in the ordinary LSB. This research aims to increase the capacity of messages that can be inserted into the image and compare between ordinary LSB and modified LSB. The results of this study are three points, namely error, stego image, and message retrieval. The resulting error ordinary LSB is high for error results, while the modified LSB has a low error. For the results

[*] Corresponding author.
Email addresses: affanfadlil525@students.unnes.ac.id (Fadlil), bprasetiyo@mail.unnes.ac.id (Prasetiyo), alamsyah@mail.unnes.ac.id (Alamsyah)

DOI: 10.15294/sji.v8i2.25356

of the stego image, the resulting image quality is usually not good, while the modified LSB has good quality. The ordinary LSB messages delivered are not good results for the message retrieval results, while the modified LSB messages have good results.

METHODS

In this study, the use of advanced LSB and image scaling to increase the capacity of messages that can be inserted. The flowchart used in this method is shown in Figure 1.

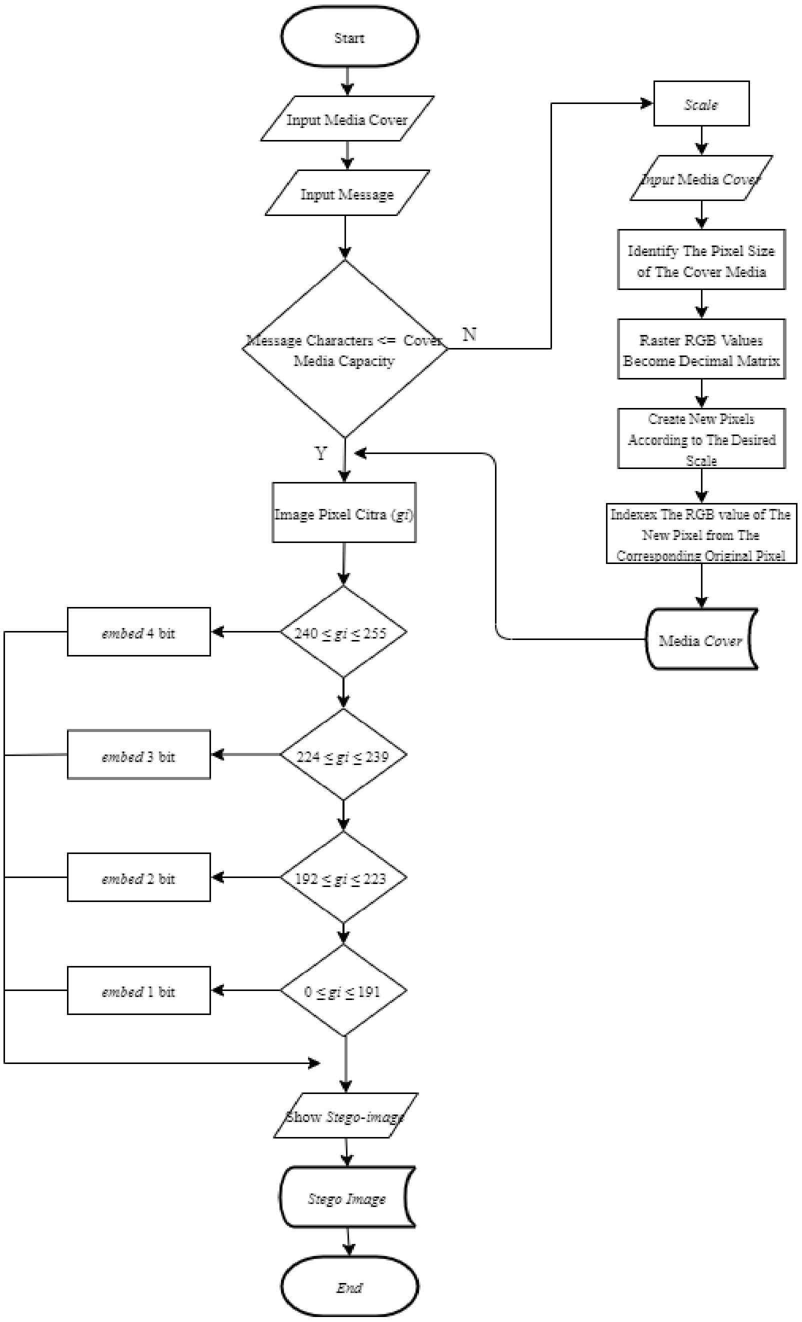

Start
Input Media Cover
Input Message
Message Characters <= Cover Media Capacity
N
Y
Scale
Input Media Cover
Identify The Pixel Size of The Cover Media
Raster RGB Values Become Decimal Matrix
Create New Pixels According to The Desired Scale
Indexex The RGB value of The New Pixel from The Corresponding Original Pixel
Media Cover
Image Pixel Citra (gi)
embed 4 bit
240 ≤ gi ≤ 255
embed 3 bit
224 ≤ gi ≤ 239
embed 2 bit
192 ≤ gi ≤ 223
embed 1 bit
0 ≤ gi ≤ 191
Show Stego-image
Stego Image
End

Figure 1. Flowchart of advanced LSB embedding and image scaling.

Advanced LSB

The advanced LSB method is a message hiding technique developed from the LSB where this method uses eight key bits and runs the XOR process with all bytes of the message to be inserted and also has rules in message insertion [2].

The insertion of messages for the pixel range with the maximum number of message bits that we can hide is described in the rules, namely:

1. Let the pixel value be *gi*.
2. If the pixel value is 240 ≤ *gi* ≤ 255 then insert 4 bits of the secret message into 3 bit LSB on the pixel cover object. This can be done by observing the first 4 bits of MSB. If all MSB values are 1, then the remaining 4 LSB bits can be used for inserting data.
3. If the pixel value is 224 ≤ *gi* ≤ 239 then insert 3 bits of the secret message into the 3 bit LSB on the pixel cover object.
4. If the pixel value is 192 ≤ *gi* ≤ 223, insert 2 bits of the secret message into 2 LSB bits on the cover object pixel.
5. If the pixel value is 0 ≤ *gi* ≤ 191 then insert 1 bit of the secret message into 1 bit LSB on the pixel cover object.

To retrieve a secret message, it can be retrieved by examining the first four bits of the MSB and using the insertion steps to retrieve the secret data bits. So the first four bits are worth 1, then 4 bits will be taken from the back as a secret message. So the match for 3 bits, 2 bits, and 1 bit [2].

Image Scaling

Using image scaling uses the upscaling method, which is part of the nearest-neighbor where the previous image is enlarged by 2x the magnification of the original image.

Embedding

The message embedding for the pixel range with the maximum number of message bits that we can hide is described in the rules written in the advanced LSB method.
The embedding process is shown in Figure 2.

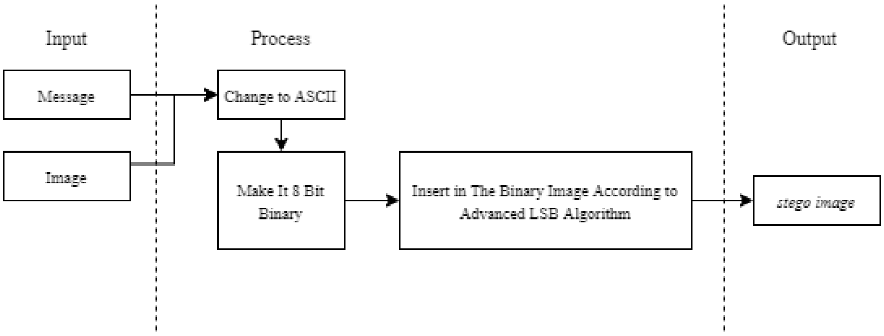


Figure 2 Embedding Process

Extraction

To retrieve a secret message, it can be retrieved by examining the first four bits of the MSB and using the insertion steps to retrieve the secret data bits. So the first four bits are worth 1, then 4 bits will be taken from the back as a secret message. That is the match for 3 bits, 2 bits, and 1 bit. The extraction process is shown in Figure 3.

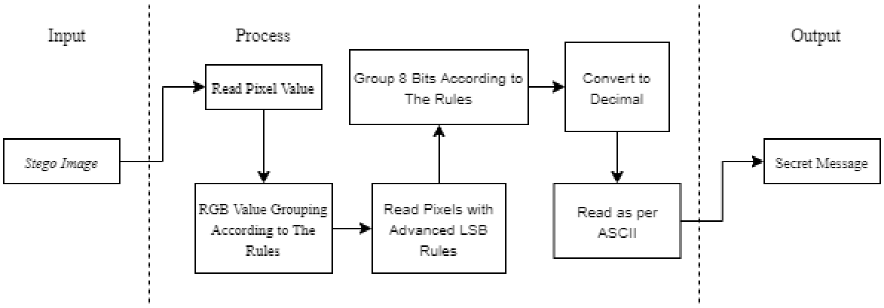

Input
Process
Output
Stego Image
Read Pixel Value
RGB Value Grouping According to The Rules
Read Pixels with Advanced LSB Rules
Group 8 Bits According to The Rules
Convert to Decimal
Read as per ASCII
Secret Message

Figure 3. Extraction Process

MSE and PSNR

Mean square error (MSE) refers to the squared error between the cover image and the stego image. MSE helps measure distortion in images. The formula is given by equation 1.

$$MSE = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}[I(i,j) - K(i,j)]^2 \qquad (1)$$

I (i, j) and *K (i, j)* are the pixel values of the original image and the stego image, respectively. M and N represent the number of rows and columns in the input image.

Peak signal to noise ratio (PSNR) is defined as the ratio of the peak or maximum signal to noise associated with the original image and the stego image. PSNR is measured in decibels (dB). The PSNR formula in equation 2.

$$PSNR = 10log_{10}\left(\frac{MAX^2}{MSE}\right) = 20log_{10}\left(\frac{MAX}{\sqrt{MSE}}\right) \qquad (2)$$

MAX is the largest possible value in an image, MSE is the average square error between pixels in two images [1].

RESULT AND DISCUSSION

This section discusses the results of research using the advanced LSB method without image scaling and the advanced LSB method with image scaling. Implementation of the method using Matlab 2014a. The research results that have been obtained include the insertion without image scaling, insertion with image scaling.

Insertion without Image Scaling

There are two processes at this stage, namely embedding or inserting and extracting or returning messages that have been inserted.

a. *Embedding*

The advanced LSB method for the embedding process uses a color image measuring 64x64, and a text message of 50 characters, namely "Lorem ipsum dolor sit amet, cras amet consectetur.". The 64x64 image is shown in Figure 4.

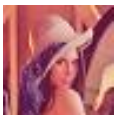


Figure 4. Image of Lena 64x64

Table 1. Text message binary value

Number	Pesan	ASCII	Nilai biner
1	L	76	01001100
2	o	111	01101111
3	r	114	01110010
4	e	101	01100101
5	m	109	01101101
:	:	:	:
51	0	null	00000000

Based on the data on the pixel value of the image in Figure 4, it can be obtained that the RGB pixel value data are grouped before the message is inserted. Inserting a message in a 64x64 pixel image using advanced LSB without image scaling produces a binary value. This is obtained from the decimal pixel value above with the first line rule that is first executed in Table 2, and the decimal to binary conversion results can be seen in Table 3.

Table 2. Pixel values are grouped.

Row	*Red*	*Green*	*Blue*
1	226, 225, 222, …, 213	135, 133, 131, …, 122	123, 117, 107, …, 111
2	225, 225, 226, …, 74	130, 130, 129, …, 8	107, 107, 107, …, 58
3	225, 227, 226, …, 92	129, 131, 130, …, 22	108, 106, 107, …, 62
4	226, 229, 226, …, 102	130, 133, 132, …, 26	109, 107, 110, …, 63
5	227, 228, 230, …, 98	132, 135, 140, …, 24	110, 109, 116, …, 62
…	**…**	**…**	**…**
64	89, 97, 232, …, 161	23, 29, 203, …, 64	57, 71, 184, …, 78

Table 3. Binary values after grouping.

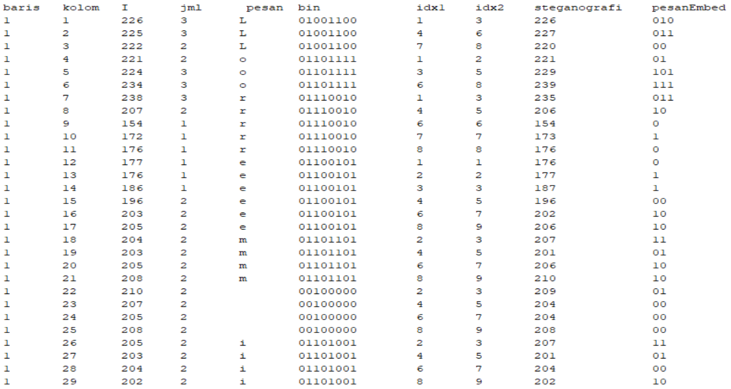
Row	*Red*	*Green*	*Blue*
1	11100010, 11100001, 11011110, …, 11010101	10000111, 10000101, 10000011, …, 01111010	01111011, 01110101, 01101011, …, 01101111
2	11100001, 11100001, 11100010, …, 01001010	10000010, 10000010, 10000001, …, 00001000	01101011, 01101011, 01101011, …, 00111010
3	11100001, 11100011, 11100010, …, 01011100	10000001, 10000011, 10000010, …, 00010110	01101100, 01101010, 01101011, …00111110
4	11100010, 11100101, 11100010, …, 01100110	10000010, 10000101, 10000100, …, 00011010	01101101, 01101011, 01101110, …, 00111111
5	11100011, 11100100, 11100110, …, 01100010	10000100, 10000111, 10001100, …, 00011000	01101110, 01101101, 01110100, …, 00111110
…	…	…	…
64	01011001, 01100001, 11101000, …, 10100001	00010111, 00011101, 11001011, …, 01000000	00111001, 01000111, 10111000, …, 010001110

After obtaining the binary value of the 64x64 image pixel, insert a 50 character text message converted to a binary value according to the ASCII code as in Table 1 after using advanced LSB without using image scaling as the rules for insertion.

Table 4. Binary value of 64x64 stego image pixels after grouping

Row	*Red*	*Green*	*Blue*
1	11100**010**, 11100**011**, 110111**00**, …, 110101**01**	1000011**1**, 1000010**1**, 1000001**1**, …, 0111101**1**	0111101**0**, 0111010**1**, 0110101**0**, …, 0110111**0**
2	11100**110**, 11100**110**, 11100**010**, …, 0100101**0**	1000001**1**, 1000001**0**, 1000000**1**, …, 00001000	01101011, 01101011, 01101011, …, 00111010
3	11100001, 11100011, 11100010, …, 01011100	10000001, 10000011, 10000010, …, 00010110	01101100, 01101010, 01101011, …00111110
4	11100010, 11100101, 11100010, …, 01100110	10000010, 10000101, 10000100, …, 00011010	01101101, 01101011, 01101110, …, 00111111
5	11100011, 11100100, 11100110, …, 01100010	10000100, 10000111, 10001100, …, 00011000	01101110, 01101101, 01110100, …, 00111110
:	:	:	:
64	01011001, 01100001, 11101000, …, 10100001	00010111, 00011101, 11001011, …, 01000000	00111001, 01000111, 10111000, …, 010001110

Bold indicates the change in binary value after inserting the binary message.
The following is the embedding process using an application that has been created as shown in Figure 5.



```
baris  kolom  I    jml  pesan  bin       idx1  idx2  steganografi  pesanEmbed
1      1      226  3    L      01001100  1     3     226           010
1      2      225  3    L      01001100  4     6     227           011
1      3      222  2    L      01001100  7     8     220           00
1      4      221  2    o      01101111  1     2     221           01
1      5      224  3    o      01101111  3     5     229           101
1      6      234  3    o      01101111  6     8     239           111
1      7      238  3    r      01110010  1     3     235           011
1      8      207  2    r      01110010  4     5     206           10
1      9      154  1    r      01110010  6     6     154           0
1      10     172  1    r      01110010  7     7     173           1
1      11     176  1    r      01110010  8     8     176           0
1      12     177  1    e      01100101  1     1     176           0
1      13     176  1    e      01100101  2     2     177           1
1      14     186  1    e      01100101  3     3     187           1
1      15     196  2    e      01100101  4     5     196           00
1      16     203  2    e      01100101  6     7     202           10
1      17     205  2    e      01100101  8     9     206           10
1      18     204  2    m      01101101  2     3     207           11
1      19     203  2    m      01101101  4     5     201           01
1      20     205  2    m      01101101  6     7     206           10
1      21     208  2    m      01101101  8     9     210           10
1      22     210  2           00100000  2     3     209           01
1      23     207  2           00100000  4     5     204           00
1      24     205  2           00100000  6     7     204           00
1      25     208  2           00100000  8     9     208           00
1      26     205  2    i      01101001  2     3     207           11
1      27     203  2    i      01101001  4     5     201           01
1      28     204  2    i      01101001  6     7     204           00
1      29     202  2    i      01101001  8     9     202           10
```

Figure 5. The process of embedding message binaries into binary images

b. Message Extraction

The extraction process is a part of recovery testing where the resulting stego image is returned to a complete message. The extraction process uses 64x64 images and 50 characters of embedded messages. The extraction process makes use of the insertion rule as the key to getting the message content from the stego image.

Insertion with Image Scaling

There are three processes at this stage, namely image scaling, embedding, and extraction or restoring messages that have been inserted.

a. *Image Scaling*

The results of image scaling can be seen in Figure 6 with the naked eye for quality, and the actual results are not visible.

Figure 6. Image scaling Lena 64x64 image to 128x128.

b. *Embedding*

The use of the advanced LSB method for the embedding process uses scaled images, and a text message of 50 characters, namely "Lorem ipsum dolor sit amet, cras amet consectetur.". 128x128 image is shown in Figure 7.

Figure 7. 128x128 pixel image

Based on the data on the pixel value of the image in Figure 7, it can be obtained that the RGB pixel value data is grouped before the message is inserted. Inserting a message in an image measuring 64x64 pixels using advanced LSB with an image that has been scaled to 128x128 produces a binary value. Obtained from the decimal pixel value with the first line rule executed first in Table 5 and the decimal to binary conversion results can be seen in Table 6.

Table 5. Pixel values are grouped

Row	***Red***	***Green***	***Blue***
1	226, 226, 225, …, 213	135, 135, 133, …, 122	123, 123, 117, …, 111
2	225, 225, 225, …, 74	130, 130, 130, …, 8	107, 107, 107, …, 58
3	225, 225, 227, …, 92	129, 129, 131, …, 22	108, 108, 106, …, 62
4	226, 226, 229, …, 102	130, 130, 133, …, 26	109, 109, 107, …, 63
5	227, 227, 228, …, 98	132, 132, 135, …, 24	110, 110, 109, …, 62
:	:	:	
128	89, 89, 97, …, 161	23, 23, 29, …, 64	57, 57, 71, …, 78

Table 6. Binary value of pixels after grouping

Row	***Red***	***Green***	***Blue***
1	11100010, 11100010, 11100001, …, 11010101	10000111,10000111, 10000101, …, 01111010	01111011, 01111011, 01110101, …, 01101111
2	11100001, 11100001, 11100001, …, 01001010	10000010, 10000010, 10000010, …, 00001000	01101011, 01101011, 01101011, …, 00111010
3	11100001, 11100001, 11100011, …, 01011100	10000001, 10000001, 10000011, …, 00010110	01101100, 01101100, 01101010, …00111110
4	11100010, 11100010, 11100101, …, 01100110	10000010, 10000010, 10000101, …, 00011010	01101101, 01101101, 01101011, …, 00111111
5	11100011, 11100011, 11100100, …, 01100010	10000100, 10000100, 10000111, …, 00011000	01101110, 01101110, 01101101, …, 00111110
:	:	:	:
128	01011001, 01011001, 01100001, …, 10100001	00010111, 00010111, 00011101, …, 01000000	00111001, 00111001 01000111, …, 010001110

After obtaining the binary value of the 128x128 Lena image pixel, then insert a 50 character text message which has been converted to a binary value according to the ASCII code as in Table 1 after using advanced LSB without using image scaling as the rules for insertion.

Table 7. The binary value of Lena's 128x128 stego image pixels after grouping

Row	***Red***	***Green***	***Blue***
1	11100**010**, 11100**011**, 11011**000**, …, 110101**01**	1000011**1**, 1000010**1**, 1000001**1**, …, 0111101**1**	0111101**0**, 0111010**1**, 0110101**0**, …, 0110111**0**
2	11100110, 11100110, 11100010, …, 01001010	10000011, 10000010, 10000001, …, 00001000	01101011, 01101011, 01101011, …, 00111010
3	11100001, 11100011, 11100010, …, 01011100	10000001, 10000011, 10000010, …, 00010110	01101100, 01101010, 01101011, …00111110
4	11100010, 11100101, 11100010, …, 01100110	10000010, 10000101, 10000100, …, 00011010	01101101, 01101011, 01101110, …, 00111111
5	11100011, 11100100, 11100110, …, 01100010	10000100, 10000111, 10001100, …, 00011000	01101110, 01101101, 01110100, …, 00111110
:	:	:	:
128	01011001, 01100001, 11101000, …, 10100001	00010111, 00011101, 11001011, …, 01000000	00111001, 01000111, 10111000, …, 010001110

Bold indicates the change in binary value after inserting the binary message.

Based on Table 7, there are changes according to the advanced LSB method where a text message that can be inserted into a Lena 64x64 image using image scaling only requires 1 line of pixels to accommodate a message of 50 characters.

c. Extraction

The resulting stego image can be returned to the original message.

CONCLUSION

The final results of each test method using advanced LSB without image scaling and advanced LSB using image scaling were compared to obtain detailed comparison results of each method. For the comparison of the two methods used, it can be concluded that image scaling can increase the maximum message insertion capacity, which has a ratio of 3: 1. Based on the comparison, it is found that the MSE value in the advanced LSB method with image scaling values has decreased, meaning that the lower the MSE value, the better the image quality. The opposite of the MSE value, the PSNR value obtained results has increased so that the resulting stego image will also be better. Using the advanced LSB method with image scaling or without image scaling results in a stego image that is successfully extracted again.

REFERENCES

[1] M. Bala, "Secure Data Transmission techniques using AES cryptography along with Image Steganographic analysis.," *Int. J. Recent Res. Asp.*, vol. 4, no. 4, pp. 21–24, 2017,

[2] H. B. Kekre, D. Mishra, R. Khanna, S. Khanna, and A. Hussaini, "Comparison between the basic LSB replacement technique and increased capacity of information hiding in LSB's method for images," *Int. J. Comput. Appl.*, vol. 45, no. 1, pp. 33-38, 2012.

[3] T. Morkel, M. S. Olivier, and J. H. . Eloff, "an Overview of Image Steganography," in *Proceedings of the ISSA 2005 New Knowledge,* Africa, pp. 51–107, July 2005.

[4] S. B. Nasution, B. Hidayat, and I. N. A. Ramatryana, "Steganalisis citra digital berbasis discrete cosine transform dengan menggunakan metode k-Nearest neighbor," *Pros. SENIATI*, p. B57.1-6, 2017, Accessed: Nov. 29, 2021.

[5] Nugraha, E. F. (2016). *Meningkatkan Kapasistas Pesan yang disisipkan dengan Metode Redundant Pattern Encoding*. accessed, 24, 2010-2011.

[6] D. R. I. M. Setiadi, H. A. Santoso, E. H. Rachmawanto, and C. A. Sari, "An improved message capacity and security using divide and modulus function in spatial domain steganography," *2018 Int. Conf. Inf. Commun. Technol. ICOIACT 2018*, vol. 2018-January, pp. 186–190, Apr. 2018

[7] I. U. W. Mulyono, A. Susanto, M. R. F. Febrian, and G. A. Rosyida, "A combination of hill cipher and LSB for image security," *Sci. J. Inform.*, vol. 7, no. 1, pp/ 153-163, 2020.

[8] R. J. Anderson and F. A. P. Petitcolas, "On the limits of steganography," *IEEE j. sel. areas commun.*, vol. 16, no. 4, pp. 474–481, 1998.

[9] 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, pp. 727–752, 2010.

[10] S. Gupta, G. Gujral, and N. Aggarwal, "Enhanced least significant bit algorithm for image steganography," *IJCEM International Journal of Computational Engineering & Management*, vol. 15, no. 4, pp. 40–42, 2012.

[11] U. Sara, M. Akter, and M. S. Uddin, "Image Quality Assessment through FSIM, SSIM, MSE and PSNR—A Comparative Study," *J. Comput. Commun.*, vol. 07, no. 03, pp. 8–18, 2019.

[12] J. Hieronymus and J. Hieronymus, "ASCII Phonetic Symbols for the World's Languages: Worldbet," *AT&T BELL Lab. Tech. MEMO*, vol. 23, 1993, Accessed: Nov. 29, 2021.

[13] C. Atika Sari, E. H. Rachmawanto, and C. A. Haryanto, "Cryptography triple data encryption standard (3DES) for digital image security," *Sci. J. Inform.*, vol. 5, no. 2, pp. 105–117, 2018.

[14] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun, "Deep Image: Scaling up image recognition," *arXiv [cs.CV]*, 2015.

[15] N. Jiang and L. Wang, "Quantum image scaling using nearest neighbor interpolation," *Quantum Inf. Process.*, vol. 14, no. 5, pp. 1559–1571, 2015.