ISSN 2278-3091
Volume 8, No.3, May - June 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse64832019.pdf
https://doi.org/10.30534/ijatcse/2019/64832019

Comparative Analysis of LSB, LSB2, PVD Methods of Data Steganography

Rashad J. Rasras[1], Mutaz Rasmi Abu Sara[2], Ziad A. AlQadi[3], Rushdi Abu zneit[4]
[1]Department of Computer Engineering, Al-Balqa' Applied University, Amman, Jordan, rashad.rasras@bau.edu.jo
[2] Department of Computer Science, Taibah University, Al-Medina, Saudi Arabia, mabusara@taibahu.edu.sa
[3]Department of Computer Engineering Al-Balqa' Applied University Amman, Jordan, natalia_maw@yahoo.com
[4]Department of Computer Engineering Al-Balqa' Applied University Amman, Jordanzneit_rush@hotmail.com

ABSTRACT

Steganography is the art of hiding data in another covering medium. This process is very important because it is used in various applications concerned deferent life fields. In this paper we will introduce 3 methods of data steganography: LSB, LSB2 and PVD. These methods will be implemented, tested and analyzed. The obtained experimental results will be compared in order to do some judgment regarding the quality, capacity, hiding time and extracting time**.**

Key words: Capacity, extracting time, hiding time, lookup table, LSB, LSB2, MSE, PSNR, PVD**.**

1. INTRODUCTION

True color image [1], [2], [3] contains three channels, the first one for the red color, the second one for the green color and the third one for the blue color, thus digital color image can be represented by a 3D matrix [4], [5]; each one of the three dimensions is reserved for one color [6], [7], [8].
Color image is a good and secure [9] media, thus it can be easily used as a covering file to hide a secret message; this process is called data steganography [10], [11], [12].
Data steganography is an important process and widely used for many reasons (some data are private, data can be confidential, the need for creating a tag in the image to identify image authority).Data steganography requires the minimum components as shown in figure (1):

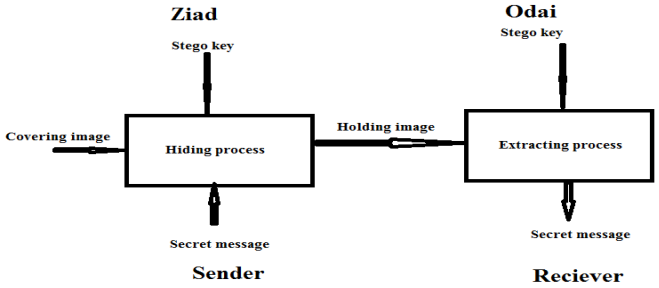


Figure 1: Process of steganography

- Covering image which is to be used to hold the secrete message.
- Secret message.
- Stego key, which is an optional, and it can be added to secure the process of hiding-extracting.

Data steganography can be performed several times, for example we can hide a secret message in a color image, then the holding image can be hidden in another, this will make it difficult to hack the secret message as shown in figure (2):

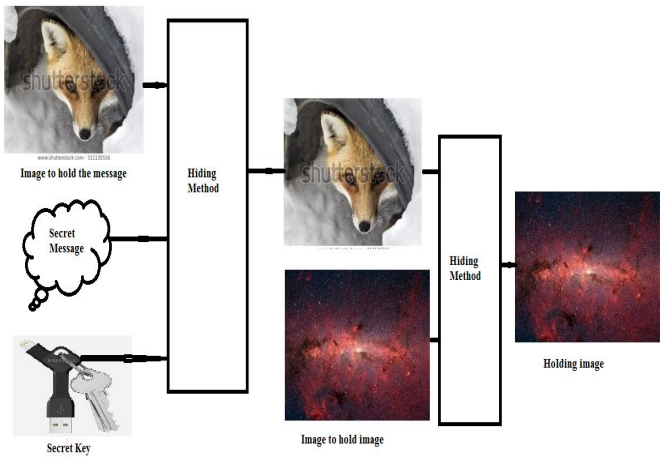


Figure 2: 2 levels of data steganography

Many methods are now using to apply data steganography, but any selected method must have the following features:

- Providing a high capacity: increasing the message length as long as possible
- Providing high efficiency: Minimizing the hiding and extracting time.
- Providing a maximum similarity between the covering and the holding images: minimizing the mean square error (MSE) and maximizing peak signal to noise ratio (PSNR) between the covering and the holding images [10], [14], [15].
- Providing some kind of security to prevent hacking process.

2. STUDIED METHODS

For comparative analysis we will focus in this paper on the most popular methods of data steganography, these methods include: LSB, LSB2 and PVD.

2.1 LSB method

Least significant bit method of data steganography (LSB) [14], [16] is a very simple method, it reserves 8 pixel from the covering image to hold one character(byte) from the secret message to be hidden as shown in figure (3), if we use a color image then each pixel in each color will hold one pixel from the secret message as shown in figure (4).

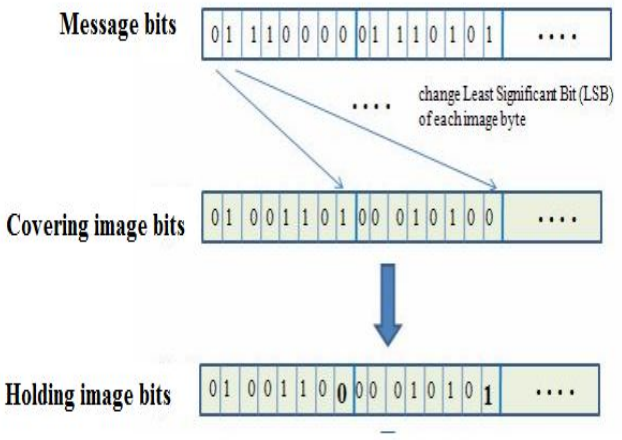


Figure 3: LSB data hiding

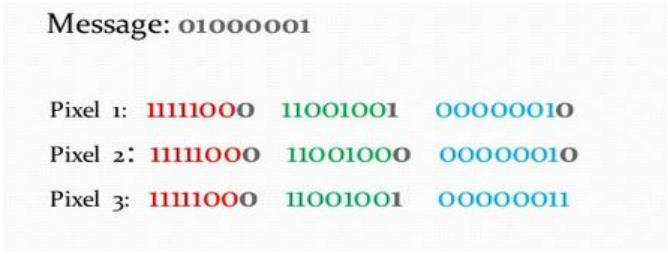


Figure 4: Hiding a message in image colors

This method will add a minor changes to the pixel (no change, or add 1 to the pixel value, or subtract 1 from the pixel value), this will lead to small MSE value and big PSNR value, thus the changes in the image will not be noticed by the human eyes.

The maximum capacity of this method is limited to the image size divided by 8.

The security level of LSB is very weak and it can be increased by adding a reference to be used as private key.

2.2 LSB2 method

This method is based on LSB method but it reserves 4 bytes from the covering image to hold one character from the secret message to be hidden. The least 2 bits from the covering byte are to be used to hide 2 bits of the character byte as shown in figure (5), so this method will double the maximum capacity, and here the maximum capacity of this method will be limited to the covering image size divided by 4.

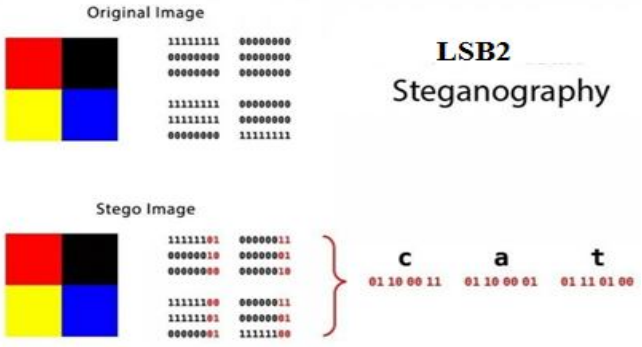


Figure 5: LSB2 hiding

This method adds also minor changes to the image by adding a value between -3 to +3, and thus it will keep the values of MSE and PSNR closed to those values in LSB method.

Example 1 and 2 illustrate how this method works:

Example 1:

Character ='A', decimal a1=65, binary=0100 0001; Covering bytes b= [217 200 120 190], complement of 3=252

Hiding Phase

Covering byte	Hiding process operations	Holding byte
b (1)=217	s(1)= uint8(bitor(bitand(b(1),525),bitshift(a1,-6)))=217	217
b (2)=200	a=bitand (a1, 48) =0 a=bitshift (a, 2) =0 s(2)=uint8(bitor(bitand(b(2),252),bitshift(a,-6)))=200	200
b (3)=120	a=bitand (a1, 12) = 0 a=bitshift (a, 4) = 0 s(3)=uint8(bitor(bitand(b(3),252),bitshift(a,-6))) = 120	120
b (4)=190	a=bitand (a1, 3) = 1 a=bitshift (a, 6) = 64 s(4)=uint8(bitor(bitand(b(4),252),bitshift(a,-6))) = 189	189

Extracting Process

Holding bytes s= [217 200 120 189],

Holding byte	Hiding process operations	Character weight
s (1)=217	d1=bitand(s (1), 3) =1 d1=bitshift (d1, 6) = 64	64
s (2)=200	d2=bitand(s (2), 3) =0 d2=bitshift (d2, 4) =0	0
s (3)=120	d3=bitand(s (3), 3) =0 d3=bitshift (d3, 2) =0	0
s (4)=189	d4=bitand(s (4), 3) =1	1
Sum	Extracted character	65

Example 2:

Character decimal a1=223, binary= 11011111

Covering bytes b= [217 200 120 190], complement of 3=252

Hiding Phase

Covering byte	Hiding process operations	Holding byte
b (1)=217	s(1)= uint8(bitor(bitand(b(1),525),bitshift(a1,-6)))= 219	219
b (2)=200	a=bitand (a1, 48) = 16 a=bitshift (a, 2) =64 s(2)=uint8(bitor(bitand(b(2),252),bitshift(a,-6)))=200	201
b (3)=120	a=bitand (a1, 12) = 12 a=bitshift (a, 4) = 192 s(3)=uint8(bitor(bitand(b(3),252),bitshift(a,-6))) = 123	123
b (4)=190	a=bitand (a1, 3) = 3 a=bitshift (a, 6) = 192 s(4)=uint8(bitor(bitand(b(4),252),bitshift(a,-6))) = 189	191

Extracting process

Holding bytes s= [217 200 120 189],

Holding byte	Hiding process operations	Character weight
s (1)=219	d1=bitand(s (1), 3) = 3 d1=bitshift (d1, 6) = 192	192
s (2)=201	d2=bitand(s (2), 3) = 1 d2=bitshift (d2, 4) =16	16
s (3)=123	d3=bitand(s (3), 3) =3 d3=bitshift (d3, 2) =12	12
s (4)=191	d4=bitand(s (4), 3) =3	3
Sum	Extracted character	223

2.3 PVD method

Pixel value deference (PVD) method uses 2 pixels from the covering image to hide a set of bits from the secret message to be hidden in image [17], [18]. This method of data hiding starts with an initialization phase, in which we divide the pixels range value (0:255) into non-overlapping partitions (these partitions are called lookup table, each partition has a lower and upper values, and each partition is associated with number of bits from the message to be hidden, this number is calculated by using formula (1):

$$t = \log2(upper_i - lower_i + 1) \qquad (1)$$

Two consecutive pixels in the i^{th} partition are denoted as P_i and P_{i+1}, respectively. The difference value, d_i, between two consecutive pixels is calculated by $d_i = |P_i - P_{i+1}|$. The absolute value of di denotes the variation present in each partition. A small value of di suggests the presence of a smooth region, whereas a larger value indicates the presence of the edge region. The d_i value can be quantized into several regions as shown in examples 1 and 2. The obtained bit sequence is converted into decimal value, t_d. The new difference value (d'_i) is obtained by $d'_i = t_d + lower_i$.

The modified pixel values are computed based on the condition shown in formula 2:

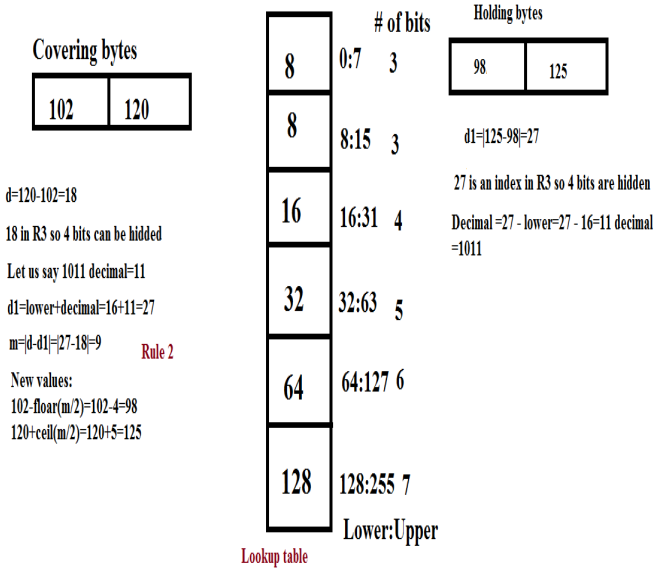
$$\text{Holding 2 bytes } (P_i', P_{i+1}') = \begin{cases} (P_i + \lceil \frac{m}{2} \rceil, P_{i+1} - \lfloor \frac{m}{2} \rfloor), \text{ if } P_i \geq P_{i+1} \text{ and } d_i' > d_i & \text{Rule 1} \\ (P_i - \lfloor \frac{m}{2} \rfloor, P_{i+1} + \lceil \frac{m}{2} \rceil), \text{ if } P_i < P_{i+1} \text{ and } d_i' > d_i & \text{Rule 2} \\ (P_i - \lceil \frac{m}{2} \rceil, P_{i+1} + \lfloor \frac{m}{2} \rfloor), \text{ if } P_i \geq P_{i+1} \text{ and } d_i' \leq d_i & \text{Rule 3} \\ (P_i + \lceil \frac{m}{2} \rceil, P_{i+1} - \lfloor \frac{m}{2} \rfloor), \text{ if } P_i < P_{I+1} \text{ and } d_i' \leq d_i & \text{Rule 4} \end{cases} \qquad (2)$$

Where $m=|d'_i - d_i|$.

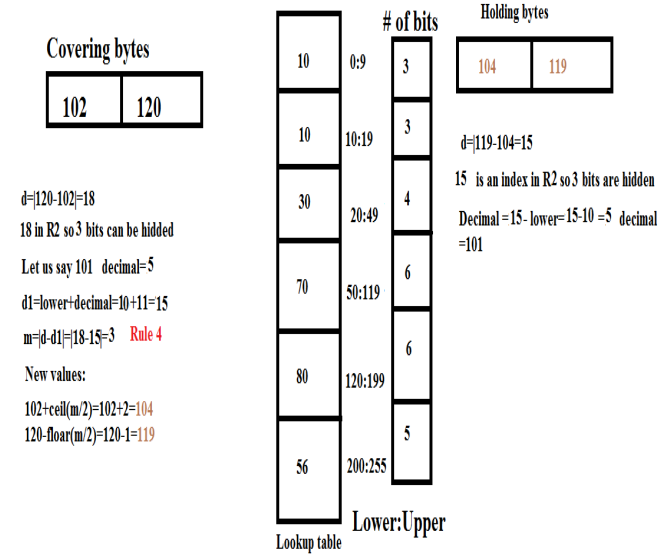
The capacity of this method is depend on the average bits in lookup table and it is always bigger than 2 bits from a message into one byte from the covering image.

Example 3 and 4 show how this method works:

Example 1:



Example 2:



3. IMPLEMENTATION AND EXPERIMENTAL RESULTS

In this part we will implement some practical experiments to analyze the above mentioned method parameters, all experiment were done in matlab environment.

a. Experiment 1: LSB implementation

In this experiment we took a huge color image with size equal 814x1024x3 = 2500608 byte(pixel); and the following message was hidden in this message using each one of the above mentioned method, the message was repeated then the same procedures was performed:

The message:

```
Albalqa applied uiversity
Faculty of engineering technology
Computer and networks engineering department
```

The results using LSB method are shown in table 1

Table 1 Experiment 1 Part 1 Results

Message length(byte)	Hiding time(seconds)	Extraction time(seconds)	MSE	PSNR
104	0.2540	0.2960	0.00016116	198.1563
208	0.2550	0.2990	0.00032232	191.2249
416	0.2560	0.3020	0.00064464	184.2934
832	0.2580	0.3026	0.0013	177.3619
1664	0.2585	0.3104	0.0026	170.4305
3328	0.2780	0.3130	0.0052	163.4990
6656	0.2840	0.3190	0.0103	156.5675
13312	0.2870	0.3270	0.0206	149.6360
26624	0.2890	0.3370	0.0413	142.7046
53248	0.3220	0.3720	0.0792	136.1860
106496	0.4230	0.4480	0.0792	136.1860

Figure (6) and (7) show original and holding images:

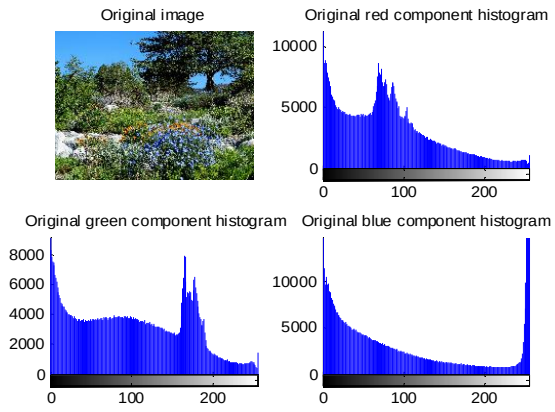


Figure 6: Original image with size 814x1024x3

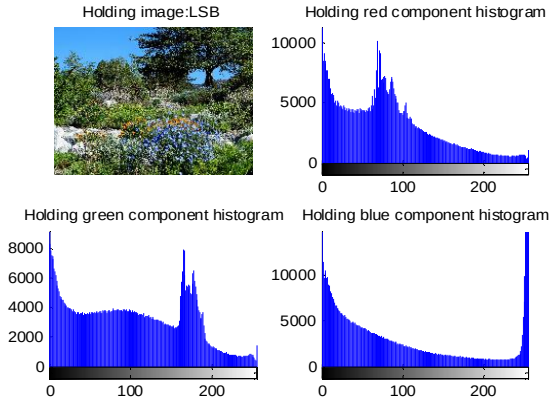


Figure 7: Image holding 3328 characters (LSB method)

We repeated the previous part of this experiment but the covering image was with a size 183x275x3=150975 pixels, the results of this part are shown in table (2)

Table 2 Experiment 1 Part 2 Results

Message length(byte)	Hiding time(seconds)	Extraction time(seconds)	MSE	PSNR
104	0.0270	0.0685	0.0027	170.0846
208	0.0280	0.0690	0.0053	163.1531
416	0.0290	0.0700	0.0107	156.2217
832	0.0300	0.0720	0.0214	149.2902
1664	0.0300	0.0720	0.0427	142.3587
3328	0.2120	0.0750	0.0713	137.2343
6656	0.2300	0.0790	0.0713	137.2343
13312	0.2420	0.0820	0.0713	137.2343
26624	x	x	x	x
53248	x	x	x	x
106496	x	x	x	x

Figures (8) and (9) show original and holding images:

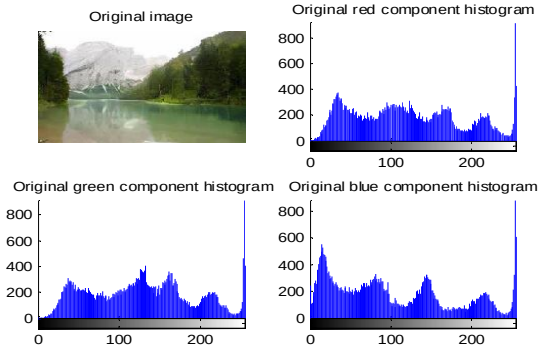


Figure 8: Original image with size 183x275x3

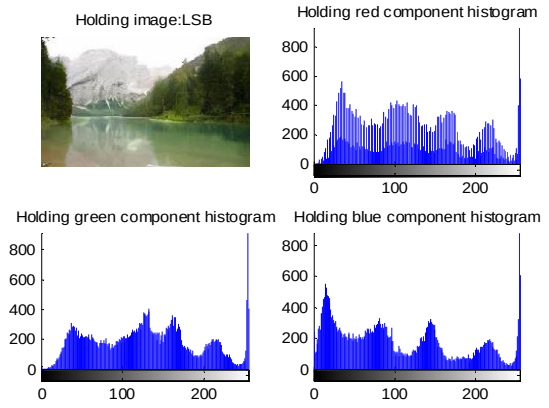


Figure 9: Image holding 3328 characters (LSB method)

b. Experiment 2: LSB2 implementation

Part 1: Using covering image with size equal 814x1024x3 = 2500608 byte (pixel)

The results of this experiment are shown in table (3):

Table 3 Experiment 2 Part 1 Results

Message length(byte)	Hiding time(seconds)	Extraction time(seconds)	MSE	PSNR
104	0.002000	0.000030	0.00040670	188.8996
208	0.003000	0.001000	0.00076341	182.6024
416	0.005000	0.001040	0.0015	175.8880
832	0.008000	0.004000	0.0029	169.1252
1664	0.016000	0.010000	0.0059	162.2347
3328	0.032000	0.031000	0.0117	155.3292
6656	0.064000	0.107000	0.0235	148.3375
13312	0.124000	0.356000	0.0470	141.3920
26624	0.249000	0.709000	0.0938	134.4929
53248	0.494000	1.575000	0.1876	127.5608
106496	0.996000	4.860000	0.3776	120.5655

Figure (10) shows the holding image:

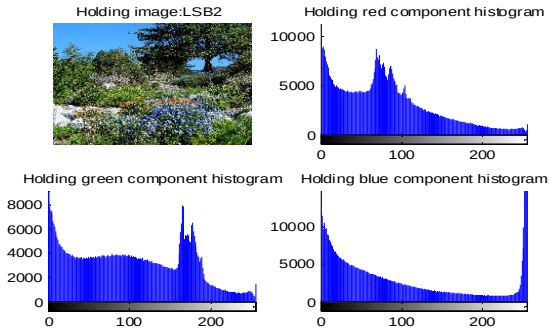


Figure 10: Image with size 814x1024x3 holding 3328 characters (LSB2 method)

Part 2: Using covering image with size equal 183x275x3=150975 byte (pixel)

The results of this experiment are shown in table (4):

Table 4 Experiment 2 Part 2 Results

Message length(byte)	Hiding time(seconds)	Extraction time(seconds)	MSE	PSNR
104	0.001000	0.000020	0.0063	161.5093
208	0.002000	0.000026	0.0124	154.7531
416	0.004000	0.001000	0.0242	148.0409
832	0.008000	0.003000	0.0485	141.0794
1664	0.017000	0.010000	0.0956	134.3047
3328	0.031000	0.032000	0.1895	127.4599
6656	0.062000	0.107000	0.3762	120.6026
13312	0.128000	0.363000	0.7541	113.6482
26624	0.252000	0.721000	1.4988	106.7789
53248	x	x	x	x
106496	x	x	x	x

Figure (11) shows the holding image:

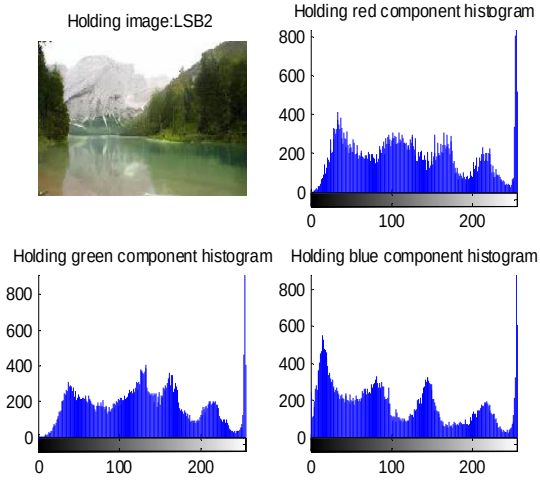


Figure 11: Image with size 183x275x3 holding 3328 characters (LSB2 method)

c. Experiment 3: PVD implementation

Part 1: Using covering image with size equal 814x1024x3 = 2500608 byte (pixel)

The results of this experiment are shown in table (5):

Table 5 Experiment 3 Part 1 Results

Message length(byte)	**Hiding time**(seconds) Including capacity computing time	**Extraction time**(seconds)	MSE	PSNR
104	10.205000	0.210000	0.00081860	181.9044
208	10.246000	0.226000	0.0031	168.6456
416	10.350000	0.259000	0.0096	157.2676
832	10.407000	0.342000	0.0184	150.7791
1664	10.832000	0.500000	0.0358	144.1156
3328	10.921000	0.898000	0.0808	135.9873
6656	10.991000	1.318000	0.1702	128.5349
13312	11.407000	2.438000	0.3454	121.4558
26624	12.449000	5.480000	0.7106	114.2423
53248	14.608000	17.340000	1.7685	105.1240
106496	18.450000	73.333000	4.6992	95.3514

Figure 12: shows the holding image using this method:

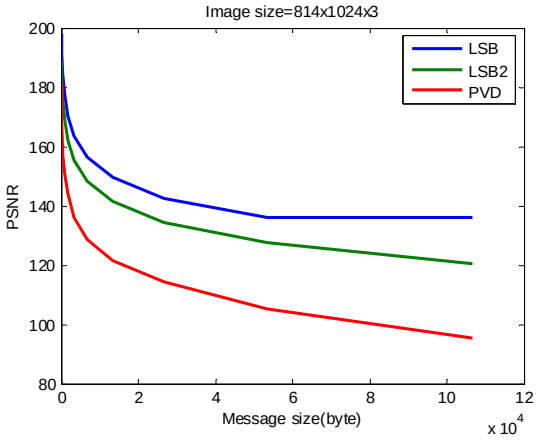


Figure 12: PSNR with covering image size= 814x1024x3

Part 2: Computing PVD capacity

Here we will compute the capacity of PVD method using the image with 814x1024x3 = 2500608 byte (pixel) and various lookup tables; the results of this experiment are shown in table (6):

Table 6 PVD Capacity for Various Lookup Tables

Lookup table partitions length						Capacity bit	Capacity byte(character)
8	8	16	32	64	128	1415347	176918
6	12	18	24	60	36	1192645	149080
12	18	18	24	48	156	1304901	163112
16	32	64	64	64	32	1610641	201330
16	16	16	16	128	64	1399984	174998
16	16	16	64	128	16	1488262	186032
32	32	64	64	32	32	1678605	209825

From the results shown in tables (1), (3) and (5) we can draw a comparison plot for the PSNR values for various methods of data steganography, from this plot (see figure (13)) we can see that the best quality of holding image can be obtained by LSB method, figure (14) shows the comparison between hiding times for the three method, from this method we can see that LSB2 hiding time is closed to LSB hiding time, but the hiding time for PVD method is much higher because it includes the time needed to calculate the capacity of PVD method

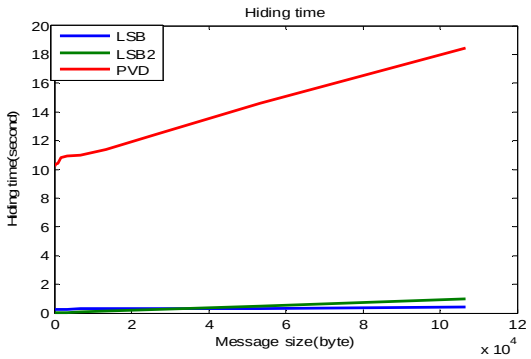


Figure 13: Hiding time with covering image size= 814x1024x3

4. CONCLUSION

Deferent methods of data steganography were implemented, tested and the obtained experimental results showed us the following facts:

- LSB provides a better results (better PSNR and hiding time);
- The three studied methods can be used easily used to hide short and huge messages keeping the quality of the holding image in a high level(acceptable PSNR);
- The Capacity of LSB2 method is double the capacity of LSB.
- The capacity of PVD methods depends on the covering image size and the selected lookup table.
- Partitions in the lookup table must be multiple of 2 in order to increase PVD capacity, because by using multiple of 2 we don't lose the fractions when we calculate the log2.
- The selected lookup table in PVD method can be used as key providing this method of some level of security.

REFERENCES

[1] Ziad A. Alqadi, Majed O. Al-Dwairi, Amjad A. Abu Jazar and Rushdi Abu Zneit, 2010, **Optimized True-RGB color Image Processing,** *World Applied Sciences Journal* 8 (10): 1175-1182, ISSN 1818-4952.

[2] Waheeb, A. and Ziad AlQadi, 2009. **Gray image reconstruction**, *Eur. J. Sci. Res.*, 27: 167-173.

[3] Akram A. Moustafa and Ziad A. Alqadi, **Color Image Reconstruction Using A New R'G'I Model**, *Journal of Computer Science* 5 (4): 250-254, 2009 ISSN 1549-3636. https://doi.org/10.3844/jcs.2009.250.254

[4] Musbah J. Aqel, Ziad ALQadi, Ammar Ahmed Abdullah, **RGB Color Image Encryption-Decryption Using Image Segmentation and Matrix Multiplication, International** *Journal of Engineering & Technology*, 7 (3.13) (2018) 104-107. https://doi.org/10.14419/ijet.v7i3.13.16334

[5] Bilal Zahran, Ziad Alqadi, Jihad Nader, Ashraf Abu Ein, **A COMPARISON BETWEEN PARALLEL AND SEGMENTATION METHODS USED FOR IMAGE ENCRYPTION-DECRYPTION** *International Journal of Computer Science & Information Technology (IJCSIT)* Vol 8, No 5, October 2016.

[6] Khaled Matrouk, Abdullah Al- Hasanat, Haitham Alasha'ary, Ziad Al-Qadi, Hasan Al-Shalabi, **Analysis of Matrix Multiplication Computational Methods,** *European Journal of Scientific Research*, ISSN 1450-216X / 1450-202X Vol.121 No.3, 2014, pp.258-266.

[7] Ziad A.A. Alqadi, Musbah Aqel, and Ibrahiem M. M. El Emary, **Performance Analysis and Evaluation of Parallel Matrix Multiplication Algorithms**, *World Applied Sciences Journal* 5 (2): 211-214, 2008.

[8] Z Alqadi, A Abu-Jazzar, **Analysis of program methods used in optimizing matrix multiplication**, *Journal of Engineering*, 2005.

[9] Musbah J. Aqel , Ziad A. Alqadi, Ibraheim M. El Emary ,**Analysis of Stream Cipher Security Algorithm**, *Journal of Information and Computing Science* Vol. 2, No. 4, 2007, pp. 288-298.

[10] J. Al-Azzeh, B. Zahran, Z. Alqadi, B. Ayyoub, M. Abu-Zaher**, A Novel zero-error method to create a secret tag for an image**, *Journal of Theoretical and Applied Information Technology*, Vol . 96. No. 13, pp. 4081-4091, 2018.

[11] Prof. Ziad A.A. Alqadi, Prof. Mohammed K. Abu Zalata, Ghazi M. Qaryouti, **Comparative Analysis of Color Image Steganography**, *JCSMC*, Vol.5, Issue. 11, November 2016, pg.37–43.

[12] M. Jose, "**Hiding Image in Image Using LSB Insertion Method with Improved Security and Quality**", *International Journal of Science and Research*, Vol. 3, No. 9, pp. 2281-2284, 2014.

[13] Emam, M. M., Aly, A. A., & Omara, F. A. **An Improved Image Steganography Method Based on LSB Technique with Random Pixel Selection**. *International Journal of Advanced Computer Science & Applications*,1(7), pp. 361-366, (2016). https://doi.org/10.14569/IJACSA.2016.070350

[14] Mohammed Abuzalata; Ziad Alqadi; Jamil Al-Azzeh; Qazem Jaber, **Modified Inverse LSB Method for Highly Secure Message Hiding**, *IJCSMC*, Vol. 8, Issue. 2, February 2019, pg.93 – 103

[15] Rashad J. Rasras, Mutaz Rasmi Abu Sara, Ziad A. AlQadi, Engineering, **A Methodology Based on Steganography and Cryptography to Protect Highly Secure Messages** *Engineering Technology & Applied Science Research*, Vol.9 Issue 1, Pages 3681-3684, 2019.

[16] Zhou X, Gong W, Fu W, Jin L. **2016An improved method for LSB based color image steganography combined with cryptography**. *In 2016 IEEE/ACIS 15th Int. Conf. on Computer and Information Science* (ICIS), Okayama, Japan, pp. 1–4 . https://doi.org/10.1109/ICIS.2016.7550955

[17] Wu D-C, Tsai W-H. **A steganographic method for images by pixel value differencing. Pattern Recognition**. Lett. 24, 1613–1626. 2003 https://doi.org/10.1016/S0167-8655(02)00402-6

[18] Das R, Das I. **Secure data transfer in IoT environment: adopting both cryptography and steganography techniques**. *In Proc. 2nd Int. Conf. on Research in Computational Intelligence and Communication Networks*, Kolkata, India, pp. 296–301, 2016. https://doi.org/10.1109/ICRCICN.2016.7813674