Image Steganography using LSB, LSB+Huffman Code, and LSB+Arithmetic Code

Wa'el Ibrahim A. Almazaydeh
Applied Science Department
Al-Balqa Applied University

H. S. Sheshadri, PhD
Department of E & C
PES College of Engineering, Mandy

ABSTRACT

The digital world is growing day by day; many new risks have emerged during the exchange of information around the world; and many ways have evolved to protect the information. In this paper, this paper will conceal information into an image by using three methods that concentrate on the compression of the date before hiding it into the image and then compare the results using Peak Signal to Noise Ratio (PSNR). The three methods that will be used are Least Significant Bit (LSB), Huffman Code, and Arithmetic Coding and then the result will be compared.

General Terms

Steganography, Security, Compression, Arithmetic Coding, and Zigzag Scanning.

Keywords

Least Significant Bit (LSB), Arithmetic Coding, ASCII code, PSNR, zigzag scanning

1. INTRODUCTION

Internet is growing day by day, and the number of the users of the internet is increasing every day. The number of Internet users continues to grow; about 26 percent of the world's 6.8 billion people-that is, 1.7 billion people-have some form of Internet access according to that study from 2001 to 2009 [2]. Internet has a large role in transferring data throughout the world, much of which is sensitive and needs a security to protect it whether in its place or during the transmission.

There are two main techniques to protect the data during the transmission: the first one is Cryptography and the second one is Steganography; Cryptography is a method used for altering and changing the letters and anyone who is unauthorized can see the message but he can't know the message because the letters is unreadable. However, Steganography is a method to conceal the secret message inside a cover media (like an image), so that who is unauthorized can't see the message because the message is hidden into a cover media.

Steganography is a branch of information hiding and its main goal is to communicate or transit the data securely in a completely undetectable manner. Literally meaning writing in a cover is the practice of hiding messages within other messages in order to conceal the existence of the original. Steganography refers to data or a file that has been concealed inside a digital image, video or audio file [5].

Compression data is the art to reduce the size of the media (like text, image, audio, and video) to store or to transmit that media. There are two types of compression: lossy and lossless.

In the lossy method for data compression, the original data is not completely restored after decompression. Mainly used for the image data compression and decompression. In the lossless method for data compression, the original data is exactly restored after decompression. Mainly used for text data compression and decompression. It can also be applied to image compression [4].

This study has used MATLAB program to implement my study and to get the results. MATLAB has a full package of tools that help to deal with images effectively and accurately.

2. RELATED WORK

This study relies on the Least Significant Bit (LSB) because it is the most famous technique used for the image Steganography. This paper has used three techniques to hide a secret message within an image by using three methods: Least Significant Bit (LSB), Huffman Code to compress the secret message before sending it to the target, and Arithmetic Coding to compress the secret message before sending it to the target. PSNR is used here to compare the results among the three techniques.

This paper also explains many security methods to increase the degree of security for the Steganography method like Zigzag Scanning, Huffman Code, and Arithmetic Coding.

2.1 Steganography

The term 'Steganography' hints to the art of “invisible” communication. Steganography is very different from cryptography, in the aspect that cryptography's chief aim is to protect the message from unauthorized attackers, whereas Steganography strives to conceal the existence of the message itself and its transmission taking place. When a message is encrypted, it is visible to the whole world and also vulnerable to malicious attacks, where as if the data is hidden in an image, apart from the entities involved in the communication process, no third party will be able to sense this transmission of data. Hence, being more secure and robust, Steganography finds a wider range of applications as compared to encryption. This technique of hiding information by embedding other, seemingly harmless messages with our message to be transferred, Steganography employs the replacement of bits that apparently are useless or not used, with bits that contain some important, invisible data within it. The type of data that can be hidden ranges from plain and cipher texts to images [3].

Figure 1 shows the Steganography technique [1]:

- Secret Message: the information that you want to embed inside the cover media.
- Stegokey: the key used in the steganography process.
- Cover Media: the medium used in Steganography process (such as: image, video, audio, etc).

- Encoding Algorithm: the method used in Steganography process.
- Stego-Media: the medium resulting from the adding the secret message into cover media using Stegokey and encoding algorithm.
- Decoding Algorithm: the method used to extract the secret message from Stego-media using Stegokey.

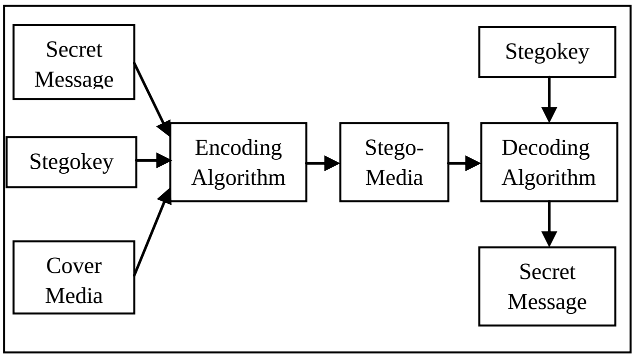


Fig 1: Steganography Technique [1]

2.2 Cryptography vs. Steganography

There are many differences between Steganography and cryptography. The following table 1 shows the differences.

Table 1: Comparison between Cryptography and Steganography [6]

Cryptography	Steganography
Known message passing	Unknown message passing
Encryption prevents unauthorized persons from discovering the contents of communication	Steganography prevents the discovery of existence of communication
It is common technology	Little known knowledge technology
Cryptography alters the structure of the secret message	Steganography does not alter the structure of the secret message

2.3 ASCII Code

American Standard Code for Information Interchange ASCII is the most common format for characters in the computer systems. In an ASCII code, each alphabetic, numeric, or special character is represented with a 7 bits binary number (a string of seven 0s or 1s). for example the ASCII Code for (A, a, X, $, #) are (65, 97, 88, 36, 35) respectively.

2.4 Least Significant Bit (LSB)

LSB is the common technique used for the Steganography. LSB method is based on substituting the redundant bits that are least important with the bits of the secret message. If we have 8 bytes of data and we want to hide the number (239) which is represented in ASCII code as (11101111). Figure 2 shows LSB process.

We Will Hide 239 which are represented as (11101111) in ASCII code by using one bit substitute:

Byte 1	**Byte 2**	**Byte 3**	**Byte 4**
1000010**0**	1000011**0**	1000100**1**	1000110**1**
1	1	1	0
1000010**1**	1000011**1**	1000100**1**	1000110**0**

Byte 5	**Byte 6**	**Byte 7**	**Byte 8**
0111100**1**	0110010**1**	0100101**0**	0010011**0**
1	1	1	1
0111100**1**	0110010**1**	0100101**1**	0010011**1**

Fig 2: Least Significant Bit (LSB) Technique.

2.5 Huffman Code

It was developed by David A. Huffman while he was a Ph.D. student at MIT, and published in the 1952 paper "A Method for the Construction of Minimum-Redundancy Codes". Huffman code is an algorithm to compression based on the frequency of occurrence of a symbol in the file that is being compressed. For example to compression the message (ABEACADABEA) we need to construct a Huffman tree which is the button-up approach according of the following steps [1]:

- Count the frequency of each character in the message as a list as shown in the table 2.
- Sort the list by frequency and make the two lowest elements into leaves, creating a parent node with a frequency that is the sum of the two lower element's frequencies [7].
- The two elements are removed from the list and the new parent node is inserted into the list by frequency. So now the list, sorted by frequency [7]
- You then repeat the loop, combining the two lowest elements.
- You repeat until there is only one element left in the list.

Table 2: the frequencies and probabilities of the text (ABEACADABEA)

Symbol	frequency	Probability
A	5	5 / 11 = 0.45
B	2	2 / 11 = 0.18
C	1	1 / 11 = 0.09
D	1	1 / 11 = 0.09
E	2	2 / 11 = 0.18

To generate a Huffman code you traverse the tree to the value you want, outputting a 0 every time you take a left hand branch and a 1 every time you take a right hand branch [7]. Figure 3 illustrate the Huffman tree for the text (ABEACADABEA).

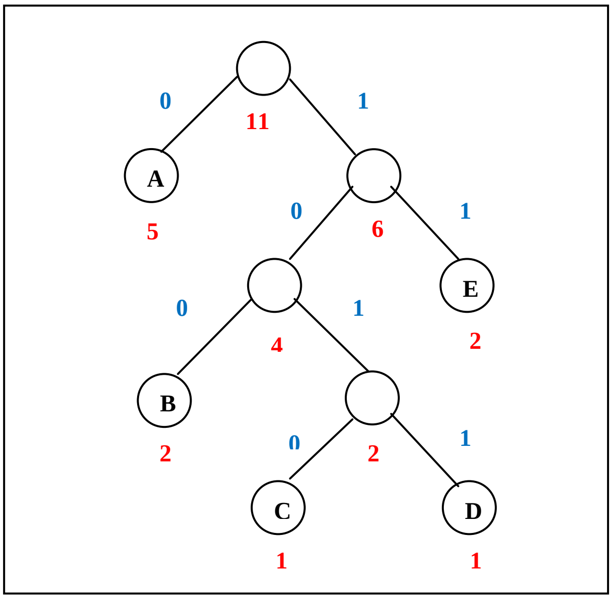


Fig 3: Huffman tree to (ABEACADABEA) [1]

According the above Huffman tree we obtain the following code word in table 3.

Table 3: The code word of the text (ABEACADABEA) using Huffman tree

Symbol	Code word
A	0
B	100
C	1010
D	1011
E	11

After completion Huffman tree to the text (ABEACADABEA) we obtain (23 bit) code word:

01001101010010110100110

While the message in ASCII code is represented as 77 bits (11 characters × 7 bits), so Huffman code saves more than 25% in the size of the message [1].

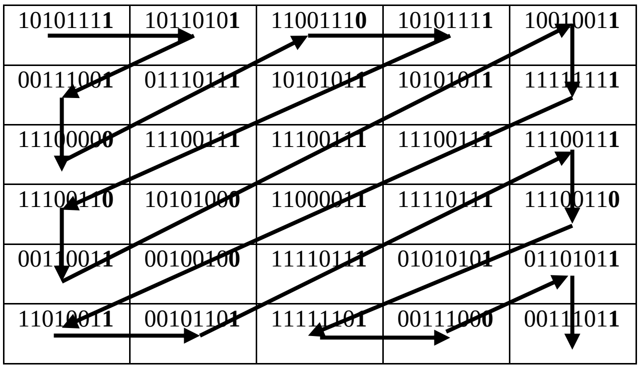
2.6 Arithmetic Coding

It is most often used when we have to code binary symbols or bits. Each bit begins the coding process. The arithmetic codes generate non-block codes; that is a correspondence between source symbols and code words does not exist. Instead, an entire sequence of source bits is allocated to a single code word which defines an interval of real numbers between 0 and 1 [8].

As the number of symbols or bits in the message increases, the interval used to represent it becomes smaller and the number of bits needed to represent the interval becomes larger. Each symbol in the message reduces the size of the interval according to its probability of occurrence. Since the symbols are not coded one at a time, this technique can achieve the highest possible coding efficiency [8].

2.7 Zigzag Scanning

This study uses zigzag scanning to increase the security in the process of hiding the secret message into the image. Zigzag scanning selects the pixels that will be hidden secret message inside; figure 3 shows the zigzag scanning process [1].



1010111**1**	1011010**1**	1100111**0**	1010111**1**	1001001**1**
0011100**1**	0111011**1**	1010101**1**	1010101**1**	1111111**1**
1110000**0**	1110011**1**	1110011**1**	1110011**1**	1110011**1**
1110011**0**	1010100**0**	1100001**1**	1111011**1**	1110011**0**
0011001**1**	0010010**0**	1111011**1**	0101010**1**	0110101**1**
1101001**1**	0010110**1**	1111110**1**	0011100**0**	0011101**1**

Fig 3: Zigzag Scanning

2.8 PSNR

Peak Signal to Noise Ratio (PSNR) (equation 1) is measured on a logarithmic scale. It depends on the mean squared error (MSE) between an original image and stego image , relative to $(2^n - 1)^2$ (the square of the highest-possible signal value in the image, where n is the number of bits per image sample) [9].

$$PSNR_{db} = 10\log_{10}\frac{(2^n - 1)^2}{MSE} \qquad (1)$$

"PSNR can be calculated easily and quickly and is therefore a very popular quality measure, widely used to compare the ‘quality’ of compressed and decompressed video images" [9].

When the PSNR value is high the changing of the image resolution is low and when the PSNR is low the changing of the image resolution is high. So my goal is to get the high PSNR value.

3. METHODOLOGY

This paper has apply three techniques for the Steganography: the first one is to hide a binary secret message into the Least Significant Bit of the image pixels, the second one is to compress a binary secret message using Huffman Code before hiding it into the Least Significant Bit of the image pixels (called LSB+HUFF), and the third one is to compress a binary secret message using Arithmetic Coding before hiding it into the Least Significant Bit of the image pixels (called LSB+ARITH). This study uses a one bit of the Least Significant Bit (LSB) only and uses the Grayscale images only.

Figure 4 shows the methodology:

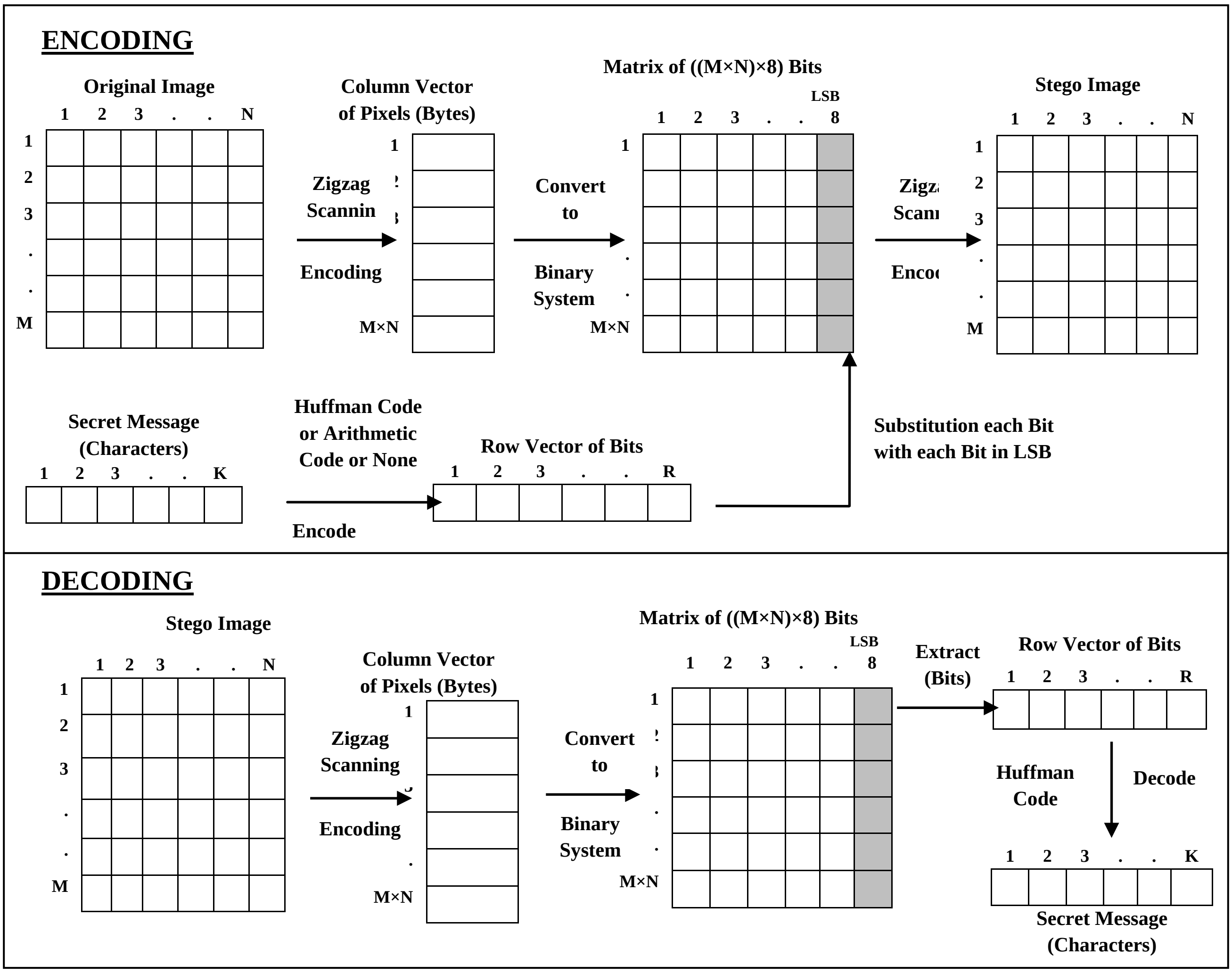


Fig 4: The Encoding and Decoding of this study.

3.1 Steganography using LSB

In this method, the study has converted the image to a column of binary values for all pixels by using zigzag scanning with size equal to (M×8) where M is the total number of pixels in the image; On the other side, the study has converted the secret message to a row of binary values with size equal (1×K) where K is the total number of bits in the image, knowing that for each character in the secret message we need 7 bits.

The size of data (Secret Message) that you can imbed into the image using this method is computed as the following:

$$S1 = M \times N - 27 \tag{2}$$

Where S1 is the size of the secret message, M is the number of rows in the image, N is the number of columns in the image, and 27 is the number of bits as illustrated in figure 5 part A; where the bits from 1 to 7 of the Steganography type may be 1, 2, 3, …, 127. When the Steganography type equals 1 means that Steganography process is LSB, when the Steganography type equals 2 means that Steganography process is LSB+Huffman code, and when the Steganography type equals 3 means that Steganography process is LSB+Arithmetic coding. The bits from 8 to 27 are the length of the secret message, 8 to 27 are the length of the secret message.

3.2 Steganography using LSB+Huffman Code

In this method, the study has converted the image to a column of binary values for all pixels by using zigzag scanning with size equal to (M×8) where M is the total number of pixels in the image; On the other side, the study has converted the secret message using Huffman Code to decrease the size of the secret message to a row of binary values with size equal (1×K) where K is the total number of bits in the image.

The size of data (Secret Message) that you can imbed into the image using this method is computed as the following:

$$S2 = M \times N - 47 \tag{3}$$

Where S2 is the size of the secret message, M is the number of rows in the image, N is the number of columns in the image, and 47 is the number of bits as illustrated in figure 5 part B; where the bits from 1 to 7 are of the Steganography type. The bits from 8 to 47 are the length of two variables (A and B) that I need for Huffman code compression process.

3.3 Steganography using LSB+Arithmetic Coding

In this method, the study has converted the image to a column of binary values for all pixels by using zigzag scanning with size equal to (M×8) where M is the total number of pixels in the image, On the other side, the study has converted the secret message using Arithmetic Coding to decrease the size of the secret message to a row of binary values with size equal (1×K) where K is the total number of bits in the image.

The size of data (Secret Message) that you can imbed into the image using this method is computed as the following:

$$S3 = M \times N - 107 \qquad (3)$$

Where S3 is the size of the secret message, M is the number of rows in the image, N is the number of columns in the image, and 47 are number of the bits as illustrated in figure 5 part C; where the bits from 1 to 7 are the Steganography type. The bits from 8 to 107 are of the length of five variables (A, B, C, D, and E) that I need for Arithmetic Coding compression process.

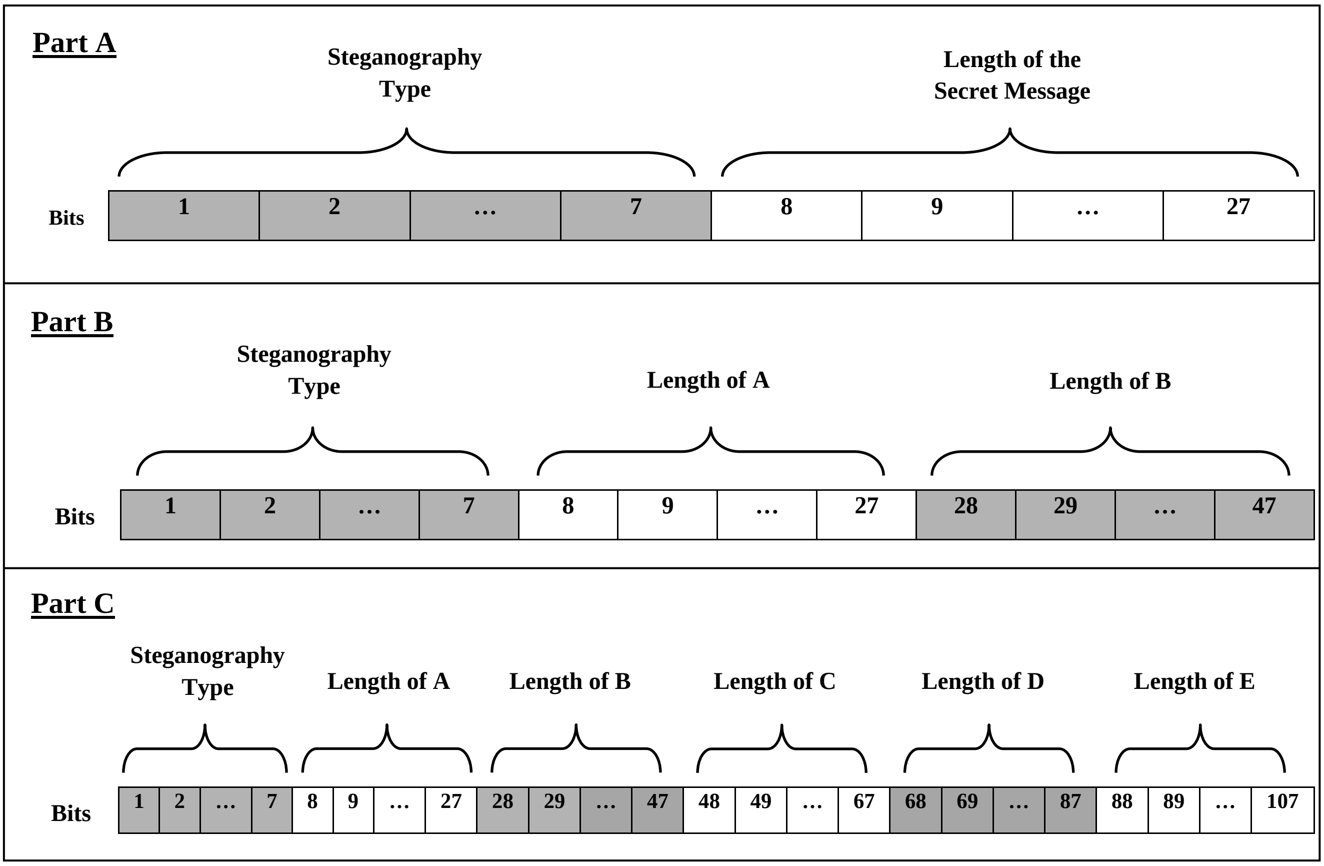


Fig 5: The number of reserved bits for LSB, LSB+Huffman, and LSB+Arithmetic.

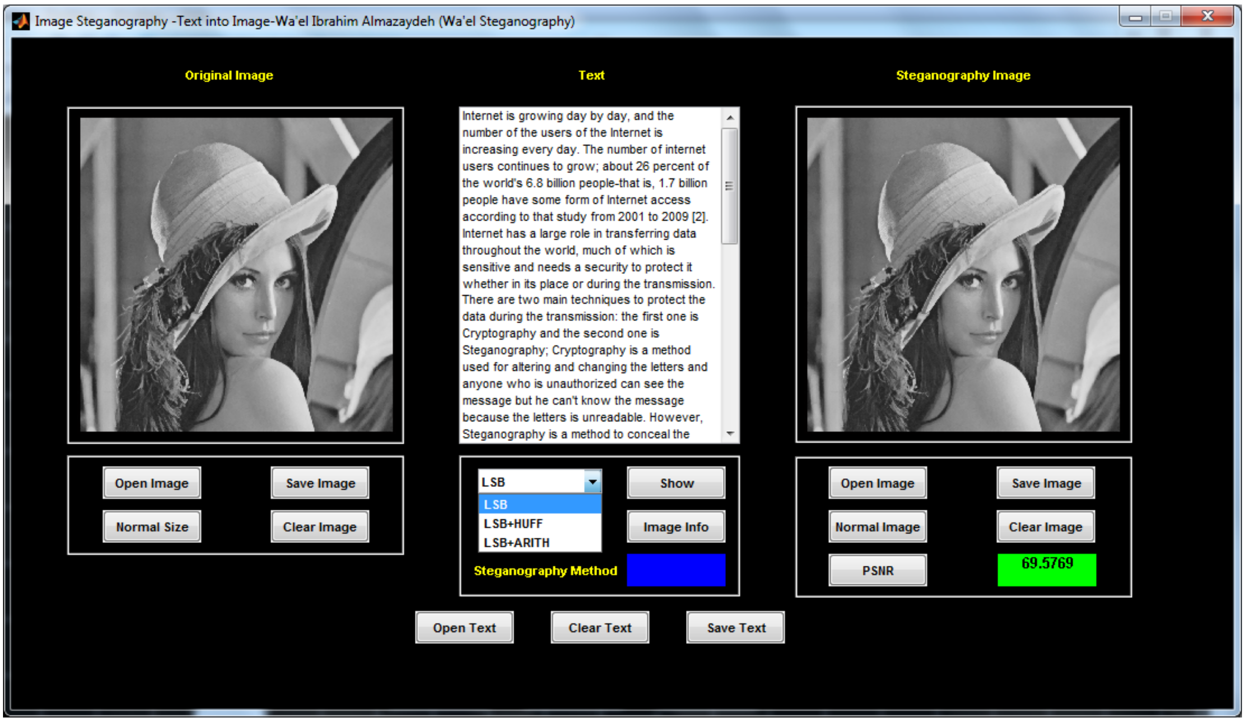
4. EXPERIMENTAL RESULTS

4.1 The Implementation

This paper has created a MATLAB program to implement this study and to get the results; it has called this program (Wa'el-Steganography) as shown in the figure 6. This study has used Lenna image (as an example) with size (1024×1024×3 pixels) and its extension is (jpeg). The following steps explain the technique:

- Convert the Lenna image to gray scale.
- The size of data (Secret message) that can be embedded in the image by using LSB method is:
- 1024 × 1024 – 27 = 1048549 bits.
- The size of data (Secret message) that can be embedded in this image by using LSB+HUFF method is:

  1024 × 1024 – 47 = 1048529 bits.
- The size of data (Secret message) that can be embedded in this image by using LSB+ARITH method is:
- 1024 × 1024 – 107 = 1048469 bits.
- Select the Secrete Message (as an example) is the Introduction of this paper. The count of the bits of this Introduction is:
- 2136 characters × 7 bits = 14952 bits.
- Choose the Steganography method; weather (LSB), (LSB+HUFF), or (LSB+ARITH).
- Compare the results between (LSB) method, (LSB+HUFF) method, and (LSB+ARITH) using the PSNR.



Internet is growing day by day, and the number of the users of the Internet is increasing every day. The number of internet users continues to grow; about 26 percent of the world's 6.8 billion people-that is, 1.7 billion people have some form of Internet access according to that study from 2001 to 2009 [2]. Internet has a large role in transferring data throughout the world, much of which is sensitive and needs a security to protect it whether in its place or during the transmission. There are two main techniques to protect the data during the transmission: the first one is Cryptography and the second one is Steganography; Cryptography is a method used for altering and changing the letters and anyone who is unauthorized can see the message but he can't know the message because the letters is unreadable. However, Steganography is a method to conceal the

PSNR 69.5769

Fig 6: (Wa'el-Steganography) program

4.2 The Results

Table 4 shows the results after implementation the three methods: LSB, LSB+HUFF, and LSB+ARITH; I have used the Introduction of this paper as a secret message for the three methods to compare the results among of them.

Table 4: comparison the PSNR among LSB LSB+HUFF, and LSB+ARITH methods

The Number of copies of introduction	Number of characters	Number of bits	Steganography Method		
			LSB (PSNR)	LSB+HUFF (PSNR)	LSB+ARITH (PSNR)
1	2136	14952	69.5769	71.1952	71.2424
5	10680	74760	62.2651	64.5688	64.5423
10	21360	149520	59.6072	61.6047	61.5981
15	32040	224280	57.8352	59.8472	59.8498
20	42720	299040	56.5864	58.6051	58.6384
25	53400	373800	55.6157	57.636	57.6384

Figure 7 shows the results (as a diagram) for the same values in table 4.

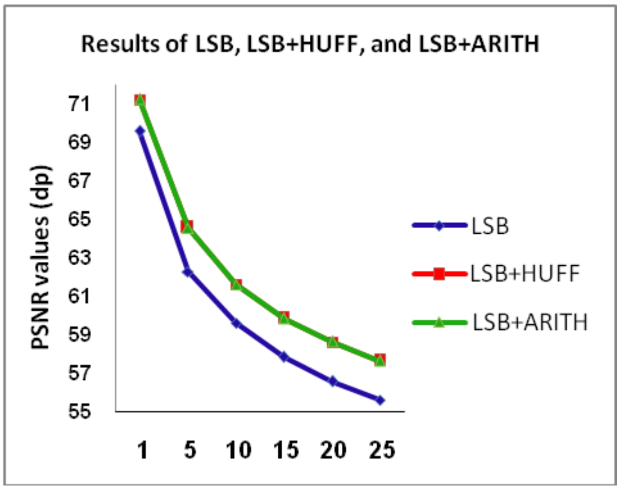


Fig 7: The PSNR of LSB and LSB+HUFF

5. CONCLUSION

This paper explains three techniques of Steganography to hide a secret message into an image: Least Significant Bit (LSB), Least Significant Bit and Huffman Code (LSB+HUFF), and Least Significant Bit and Arithmetic Coding (LSB+ARITH), I have used the Peak Signal to Noise Ration (PSNR) to compare the results of the three methods; the high values of PSNR of the three methods were (LSB+ARITH) and (LSB+HUFF) then (LSB). In the future work, I will create a new method to hide a secret message into an image that has the highest PSNR value from the results in this paper.

6. REFERENCES

[1] Wa'el Ibrahim A. Al-Mazaydeh. Image Steganography using LSB and LSB+Huffman Code. International Journal of Computer Applications (0975 – 8887), Volume 99– No.5, August 2014.

[2] Michael E. Whitman, Herbert J. Mattord. Principles of Information Security, Forth Edition.

[3] Dharmesh Mistry, Richa Desai, and Megh Jagad. Hidden Data Transmission using Image Steganography. International Journal of Computer Applications (0975 – 8887), Volume 130 – No.14, November 2015.

[4] www.rfwireless-world.com. Difference between lossless data compression vs lossy data compression. http://www.rfwireless-world.com/Terminology/lossless-data-compression-vs-lossy-data-compression.html. [Accessed: 24-august-2016]

[5] Abhishek Koluguri, Text Steganography Methods and its Tools, International Journal of Advanced Scientific and Technical Research, Issue 4 volume 2, March-April 2014.

[6] K.Thangadurai and G.Sudha Devi, 2014 International Conference on Computer Communication and Informatics (ICCCI -2014), Jan. 03 – 05, 2014, Coimbatore, INDIA. 2014 IEEE.

[7] www.siggraph.org. A quick tutorial on generating a huffman tree https://www.siggraph.org/education/materials/HyperGraph/video/mpeg/mpegfaq/huffman_tutorial.html. [Accessed : 2-august-2016]

[8] Sukhpreet Kaur, Vanita Rani. Designing an Efficient Image Encryption-Compression System using a New HAAR, SYMLET and COIFLET Wavelet Transform. International Journal of Computer Applications (0975 – 8887) Volume 129 – No.15, November2015.

[9] Iain E. G. Richardson. H.264 and MPEG-4 Video Compression, The Robert Gordon University, Aberdeen, UK 2003.