COMPARISON STUDY BETWEEN CLASSIC-LSB, SLSB AND DSLSB IMAGE STEGANOGRAPHY

Dr. Mohammed Abbas Fadhil Al-Husainy

Department of Multimedia Systems, Faculty of Sciences and Information Technology,
Al-Zaytoonah University of Jordan.
Amman-Jordan
E-mails: dralhusainy@yahoo.com , alhusainy@zuj.edu.jo

Abstract

Steganography is one of many techniques that are used to hide secret information to prevent any attackers to make damage in this information or use it in illegal form. Classic Least Significant Bit (LSB), Segmented Least Significant Bit (SLSB) and Developed Segmented Least Significant Bit (DSLSB) are techniques that are based on hiding secret information in the least significant bit of the pixels in the stego-image. In this paper, we have given to researchers who are interested in this field an analytical comparison between these techniques. This comparison will help them to stand on the strengths and weaknesses of these techniques.

Keywords—Security,Distortion, Embedding,Substitution

1 INTRODUCTION

Recently, steganography is implemented by using digital media. Secret message is embedded inside digital cover media like text, images, audio, video or protocols depending upon the requirement and choice of the sender. Compared with the other types of steganography, the image steganography is most widely used. The reason behind the popularity of image steganography is the large amount of redundant information present in the images that can be easily altered to hide secret messages inside them, and because it can take advantage of the limited power of the human visual system (HVS). With the continued growth of strong graphics power in computer and the research being put into image based steganography, this field will continue to grow at a very rapid pace [1, 2, 3].

Steganography has a wide range of applications. The major application of steganography is for secret data communication. Covert channels in TCP/IP involve masking identification information in the TCP/IP headers to hide the true identity of one or more systems. Cryptography is also used for the same purpose but steganography is more widely used technique as it hides the existence of secret data. Another application of steganography is feature tagging. Captions, annotations, time stamps, and other descriptive elements can be embedded inside an image, such as the names of individuals in a photo or locations in a map [3, 4].

The Steganography technique is the perfect supplement for encryption that allows a user to hide large amounts of information within an image. Thus, it is often used in conjunction with cryptography so that the information is doubly protected; first it is encrypted and then hidden so that an adversary has to first find the hidden information before decryption take place. The problem with cryptography is that the encrypted message is obvious. This means that anyone who observes an encrypted message in transit can reasonably assume that the sender of the message does not want it to be read by casual observers. This makes it possible to deduce the valuable information. Thus, if the sensitive information will be transmitted over unsecured channel such as the internet, steganography technique can be used to provide an additional protection on a secret message [5, 6, 7, 8].

A good technique of image steganography aims at three aspects. First one is capacity (the maximum data that can be stored inside cover image). Second one is the imperceptibility (the visual quality of stego-image after data hiding) and the last is robustness [5].

When hiding information inside images usually Classic-Least Significant Bit (LSB) method is used. Classic-LSB embedding is no more secured now-a-days. In the LSB method the 8th bit of

every Byte of the carrier file is substituted by one bit of every bit of the secret information [9]. The LSB method usually does not increase the file size, but depending on the size of the information that is to be hidden inside the file, the file can become noticeably distorted.

Mohammed A.F. Al-Husainy in [10] proposed the Segmented-LSB technique (SLSB). The main goal of Segmented-LSB technique is to enhance the performance of the Classic-LSB technique by supporting it with three strong points:

- Decrease the distortion/noise that will be appearing in the pixels of the stego-image.
- Increase the capability of hiding very long secret message in a small stego-image.
- Increase the immunity of the stego-image against the Human Visual System (HVS) attacks.

The drawback of SLSB is the long time that is spending during the embedding operation. Mohammed A.F. Al-Husainy in [11] introduced DSLSB that is minimizing the time of the embedding operation.

In the following sections, we will give a simple and intensive explanation about each of the three techniques under study. Each technique has been tested through hide secret messages of different lengths in some images. Recorded results from these experiments were analyzed and placed in the tableand charts enable researchers to note the differences in the application of each technique.

Two definitions that are used in these techniques are listed below:

A ***secret message*** is an English message might be contains alphabetic letters (‘a’…’z’) or numbers (‘0’…’9’) or any special symbols like: ('space character', '.', ',', '(', ')').

A ***stego-image***, for the purpose of testing, that is candidate to be used in this work is a bitmap images (.bmp) type. In general, each file of type (.bmp) is consisting of a header part which is containing much information like (Width and Height of the image, number Palette, number of bits for each pixel) followed by the data of the bitmap image pixels, follow that usually most image files have many unused bytes that are added by the operating system to keep the size of the image file measure (in Kilobyte). The pixels of each image are representing as a two dimensional array, but in these techniques treat the pixels of the image as a one dimensional array list of bytes, (where each byte has a value between (0…255)), by reading the bytes of the two dimensional image row by row and stores them as a one dimensional array list.

2 CLASSIC-LSB IMAGE STEGANOGRAPHY TECHNIQUE

The Least Significant Bit (LSB) steganography technique works by representing each character (byte) of the secret message as a set of 8-bits (where 1 byte ≡ 8 bits). And then hide/replace the bits of the characters in the least significant bit of the pixels in the stego-image. When the secret message has *n* characters, then LSB technique need at least (*n**8) pixels in the stego-image to hid the bits of the *n* characters.

By substitute LSB of each pixel in the stego-image with one bit (from the 8-bits) of each character in the secret message, this replacement operation will cause some distortion/noise in the stego-image. By using Human Visual System (HVS), the attackers may doubt that the stego-image contain a secret information in it. In general, whenever the length of the secret message (number of characters) increase, then the noise in the stego-image probably will increase as a result. This will make a restriction in hiding a very long message in a small stego-image. Therefore, we will tend to choose a short message to hide it in a large stego-image to minimize the noise that is happened in the pixels of the stego-image and to put aside the doubt about containing the stego-image any secret information.

Also, when any attacker success to know that the stego-image has a secret message, it is easy to get this message by reconstructed it from the LSB of the pixels in the stego-image.

Algorithm:*Classic-LSB*
(for embedding the characters of the secret ***Message*** in the stego-image ***Image*****)**
// Hiding Operation
Step1: Create ***MessageB***, which is a list that contains a binary representation (bits) of all characters in the secret ***Message***. The number of elements (size) of this list is (***n****8), where *n* is the number of characters in the secret ***Message***.
Image: is an image of size *m*, where *m*represent number of Bytes and equal (Width × Height × Palette)
Step2: ***j***=0
For ***i* = 1** To **(*n**8)**
{
Substitute the ***MessageB***[***i***] instead of $Image_{LSB}[j]$
j=***j***+1
}

// Extracting Operation
Step1: Read from LSB of the pixels in the stego-image and store it in ***MessageB***.
Step2: Reconstruct the secret ***Message*** from ***MessageB***.

3 SEGMENTED LSB IMAGE STEGANOGRAPHY TECHNIQUE

In the following paragraphs, the explanation of the operations that are doing by the Segmented LSB (SLSB) will be given.

Before listing the algorithm's steps that describe the operations of (SLSB), some data structures that are using in the algorithm are defined follow:

1. ***MessageB***: is a list that contains a binary representation (bits) of all characters in the secret ***Message***. The number of elements (size) of this list is (***n****8), where *n* is the number of characters in the secret ***Message***.
2. ***ImageB***: is a list of the Least Significant Bit (LSB) of all pixels in the stego-image. The number of elements (size) of this list is (***m***), where *m* is the size of the ***Image*** and its equal (Width × Height × Palette).
3. ***SegmentLength***: is a positive integer number between (2 … (***n****8)/2) which represents the length of each segment (number of bits) in the ***SegmentList***.
4. ***SegmentsList***: is a list of segments that is created from the ***MessageB*** by splitting it to ***k*** segments, where ***k*** = (***n****8) / ***SegmentLength***. And each segment has number of bits equal ***SegmentLength***.
5. ***SegmentIndex***: is a list of indices, each index represents the first index of a sequence of bits in ***ImageB*** that is having a best match with the bits of one of the segments in ***SegmentsList***. We must note that there is no overlapping between the sequences of match bits in this technique.

Algorithm:*Segmented-LSB* (*SLSB*)
// Hiding Operation
(for embedding the characters of the secret ***Message*** in the stego-image ***Image*****)**
Step1: Calculate the ***TotalSize*** (in byte) that is required to store:
(1) Length of secret message (number of character)
(2) ***SegmentLength***
(3) Size of ***SegmentList***
Step2: For ***i* = 1** To **((*n**8) / *SegmentLength*)**
{
For ***j* = ((*TotalSize**8)+1)** To ***m***
{
***x* = 1**
***BestMatch* = 0**
***BestIndex* = -1**
w=j

For ***w* = *j*** To **(*j*+ *SegmentLength*)**
{
Find the number of matched bits ***MBits*** in ***Segment*[*i*][*x*]** with the bits of ***ImageB*[*w*]**
***x* = *x*+1**
}
If (***MBits***>***BestMatch***)
{
BestMatch* = *MBits
BestIndex* = *j
}
}
SegmentIndex*[*i*] = *BestIndex
Substitute the bits of ***Segment*[*i*]** instead of the bits in ***ImageB*** starting at ***BestIndex***
}

Step3: Store the bits representation of the above three information (in Step1) in the Least Significant Bit (LSB) at the start of the ***ImageB*** list (from **bit #1** to **bit #(*TotalSize**8)**).

// Extracting Operation

Step1: Read from the stego-image the information that is stored in Step 3 of the hiding operation.

Step2: Reconstruct the segments of the secret message by using the extracted information in Step1.

Step3: Reassembling the all the segments that are constructed in Step2 to regenerate the characters of the secret message.

4 DEVELOPED SEGMENTEDLSB IMAGE STEGANOGRAPHY TECHNIQUE

In DSLSB, same data structures that have been used in SLSB are using here. When we applying SLSB to hide secret messages in stego-images. We note that the time that is spending through the full search to find the ***BestIndex*** in Step2 is long when compare it with the Classic LSB. To minimize this time, DSLSB uses additional factor to overcome this drawback in the hiding operation. The factor is ***AcceptedErrorRatio*** which has a value between (0…100) represent the percentage of dissimilarity that is accepting between the bits of each segment in ***Segment*** and the bits in ***ImageB***. By using this ratio in DSLSB, the technique can exclude many unnecessary rounds of search during the hiding operation.

Algorithm:*Developed Segmented-LSB*(*DSLSB*)

// Hiding Operation

Step1: Calculate the ***TotalSize*** (in byte) that is required to store:

(4) Length of secret message (number of character)
(5) ***SegmentLength***
(6) Size of ***SegmentList***

Step2: Set an error ratio ***AcceptedErrorRatio*** (between 0…100)

For ***i* = 1** To **((*n**8) / *SegmentLength*)**
{
For ***j* = ((*TotalSize**8)+1)** To ***m***
{
***x* = 1**
***BestMatch* = 0**
***BestIndex* = -1**
w=j
Initially Assume ***Error=100;***
While (***w*<(*j*+ *SegmentLength*))** and **(*Error*>*AcceptedErrorRatio*)**
{
Find the number of matched bits ***MBits*** in ***Segment*[*i*][*x*]** with the bits of ***ImageB*[*w*]**

Calculate error ratio ***Error*** between the bits of ***Segment[i][x]*** and ***ImageB[w]***
x = x+1
w = w+1
}
If (***MBits***>***BestMatch***) or **(*Error*<*AcceptedErrorRatio*)**
{
BestMatch = MBits
BestIndex = j
}
}
SegmentIndex[i] = BestIndex
Substitute the bits of ***Segment[i]*** instead of the bits in ***ImageB*** starting at ***BestIndex***
}

Step3: Store the bits representation of the above three information in the Least Significant Bit (LSB) at the start of the ***ImageB*** list (from **bit #1** to **bit #(*TotalSize**8)**).

// Extracting Operation

Step1: Read from the stego-image the information that is stored in Step 3 of the hiding operation.

Step2: Reconstruct the segments of the secret message by using the extracted information in Step1.

Step3: Reassembling the all the segments that are constructed in Step2 to regenerate the characters of the secret message.

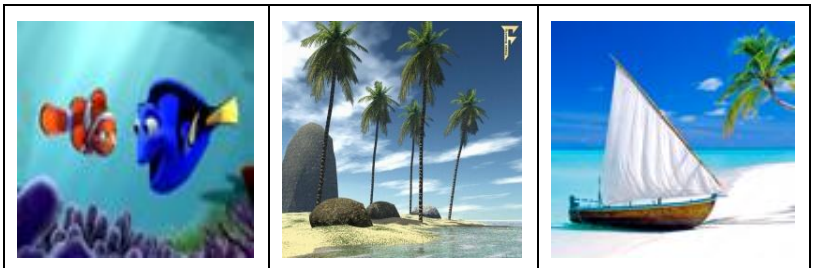
5 EXPERIMENTSAND RESULTS

Some experiments are performed by applying Classic LSB, SLSB and DSLSB for hiding some secret messages in different (.bmp) images. The results from these experiments were recorded in the following tables and charts to help the readers note the differences in the performance between these techniques.

Fig. 1 shows the stego-images, of different sizes, that are used in the experiments. Table 1 summarizes the recorded results from the experiments using ***SegmentLength*** = 10 and ***AcceptedErrorRatio***=1.5%. The effect on the performance of SLSB and DSLSB when we change the values of ***SegmentLength*** and ***AcceptedErrorRatio*** was presented and discussed in [10, 11]

Fig. 2 shows graphically (a) Number of LSB changed (b) Signal to Noise Ratio (SNR) (c) Time in second of the experiments in Table 1.

The required programs to implement the Classic-LSB,SLSB and DSLSB techniques are written by using C++ programming language and executing them on a computer system of 2.53GHz processor with 4.0 GB memory and Microsoft Windows 7 operating system.

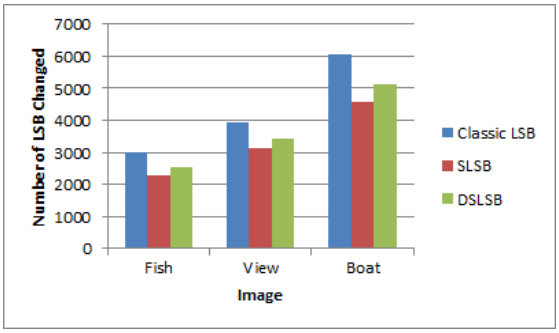


Fish (100×75×3) | View (96×96×3) | Boat (128×128×3)

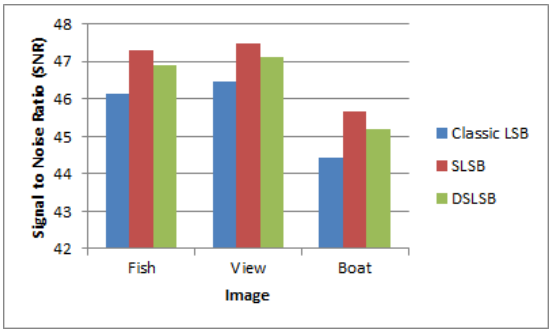
Fig.1. Stego-Images (.bmp) of Size (Width × Height × Palette)

Table 1. Recorded Results of ImplementingClassic-LSB, SLSB and DSLSB

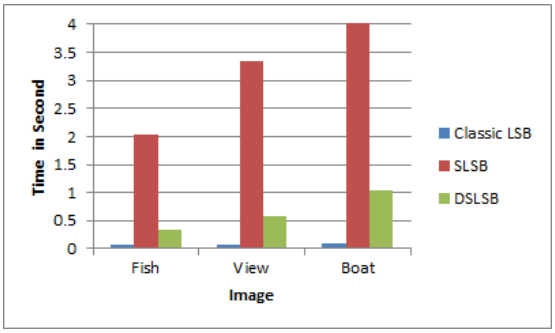
	Classic-LSB			SLSB			DSLSB		
Stego-Image	**Fish**	**View**	**Boat**	**Fish**	**View**	**Boat**	**Fish**	**View**	**Boat**
Length of Secret Message (Character)	750	1000	1500	750	1000	1500	750	1000	1500
Number of Changed LSB	3010	3957	6065	2302	3141	4574	2520	3411	5110
Signal to Noise Ratio (SNR) of Stego-Image	46.141	46.484	44.448	47.305	47.487	45.673	46.912	47.128	45.191
Time of Hiding Operation (Second)	0.062	0.078	0.093	2.044	3.338	9.376	0.343	0.592	1.029
Time of Extracting Operation(Second)	0.078	0.062	0.109	0.073	0.063	0.109	0.078	0.062	0.109



(a) Number of LSB Changed



(b) Signal to Noise Ratio (SNR)



(c) Time in Second

Fig.2. Graphic representation of the performance of Classic-LSB, SLSB and DSLSB for three points: (a) Number of LSB Changed, (b) Signal to Noise Ratio (SNR) and (c) Time in Second

6 CONCLUSION

From the above table of results and figures, we can extract and conclude the following points:

- Classic-LSB technique spends short time in the hiding operation because it does not do any type of search to find a good matching between the Bits of the secret message with the LSBs of pixels in the stego-image.
- Because Classic-LSB did not perform any search for best matching, this will lead to increase the number of LSBs that are changed by this technique, and as a result the SNR will be decrease.
- Whereas SLSB is perform an exhaustive search to find the best match between the Bits of the secret message and the LSBs of the pixels in the image. This will make the SLSB technique very slow.
- The exhaustive search in SLSB technique helps it to minimize the number of LSBs that are changed by this technique, and as a result the SNR will be increase.

- On the third side, DSLSB technique success to make balance through employing additional factor (***AcceptedErrorRatio***) to minimize the time that is require to do the exhaustive search, in the hiding operation, by excluding many unnecessary rounds in the search. This makes the time of DSLSB technique near to Classic-LSB technique. But this also makes a little increase in the number of LSBs changed and decreases the value of SNR.
- The time of the extraction operation in the three techniques isapproximately equal.

References

[1] Kaur, R. Dhir, & G. Sikka,"A new image steganography based on first component alteration technique", International Journal of Computer Science and Information Security (IJCSIS), 6, pp.53-56, 2009.http://arxiv.org/ftp/arxiv/papers/1001/1001.1972.pdf

[2] Alvaro Martin, Guillermo Sapiro, &GadielSeroussi,"Is Steganography Natural", IEEE Transactions on Image Processing, 14(12), pp.2040-2050, 2005.doi: 10.1109/TIP.2005.859370

[3] Bhattacharyya, A. Roy, P. Roy, & T. Kim,"Receiver compatible data hiding in color image", International Journal of Advanced Science and Technology, 6, pp.15-24, 2009.http://www.sersc.org/journals/IJAST/vol6/2.pdf

[4] EE. KisikChang, J. Changho, & L. Sangjin,"High Quality Perceptual Steganographic Techniques", Springer. 2939, pp.518-531, 2004.doi: 10.1007/978-3-540-24624-4_42, http://www.springerlink.com/content/c6guuj5xnyy4wj3c/

[5] C. Kessler,"Steganography: Hiding Data within Data" An edited version of this paper with the title "Hiding Data in Data", Windows & .NET Magazine, 2001. [Online] Available: http://www.garykessler.net/library/steganography.html (October 4,2011)

[6] Gandharba Swain, &S.K.Ienka,"Steganography-Using a Double Substitution Cipher", International Journal of Wireless Communications and Networking. 2(1), pp.35-39, 2010. ISSN: 0975-7163. http://www.serialspublications.com/journals1.asp?jid=436&jtype

[7] Hideki Noda, MichiharuNimi, &Eiji Kawaguchi,"High- performance JPEG steganography using Quantization index modulation in DCT domain", Pattern Recognition Letters, 27, pp.455-46, 2006.http://ds.lib.kyutech.ac.jp/dspace/bitstream/10228/450/1/repository6.pdf

[8] Kathryn,"A Java Steganography Tool", 2005.http://diit.sourceforge.net/files/Proposal.pdf

[9] Motameni, M.Norouzi, M.Jahandar, & A. Hatami,"Labeling method in Steganography", Proceedings of world academy of science, engineering and technology, 24, pp.349-354, 2007. ISSN 1307-6884. http://www.waset.org/journals/waset/v30/v30-66.pdf

[10] Mohammed A.F Al Husainy, "Message Segmentation to Enhance the Security of LSB Image Steganography", International Journal of Advanced Computer Science and Applications, 3(3): 57-62, 2012.http://www.ijacsa.thesai.org

[11] Mohammed A.F Al Husainy, "Developed Segmented LSB Image Steganography", International Science and Technology Conference (ISTEC 2012), Dubai, December 13-15, 2012. http://www.iste-c.net