SSB-7: A New Image Steganography System for Messsage Insertion Chance Enhancement using Bit 7

Rajkumar Yadav
Dept. of Comp. Sc. & Engg.
UIET, MD University, Rohtak-124001 (India)

Ravi Saini
Deptt. Of Comp. Sc. & Engg.
UIET, MD University, Rohtak-124001 (India)

Gaurav
Dept. of Comp. Sc. & Engg.
UIET, MD University, Rohtak-124001 (India)

ABSTRACT
In this paper, a new Image Steganography method based on the spatial Domain is proposed. It does not use the known LSB's bit to embed the message. It uses the bit 7 of pixel value of cover image. The main idea is to make the bit 7 of pixel value equal to the message bit by adding or subtracting 1 to the pixel value. Our method is imperceptible to HVS because it provides minimum degradation in image quality due to only +1 or -1 change at a pixel value. Our method also provides 99.21% chances of message insertion at a pixel value which is near about optimal solution. We have compared our method with other methods in spatial domain by using various image steganography parameters.

Keywords
Steganography, Cryptography, HVS, LSB method.

1. INTRODUCTION
With the catastrophic growth of digital media, its security related issues are becoming a greater concern. Cryptography and Steganography provides the solution to security related issues. Steganography is the art and science of hiding information within some cover media. Steganography means "covered writing" in Greek [1]. Steganography is different from cryptography which is about concealing the content of message whereas Steganography is about concealing the existing of message itself [2].

Steganography techniques use different cover media like images, audio and video files for secret communication. Images provide excellent carriers for hiding the information [3]. The image steganography model is given below:

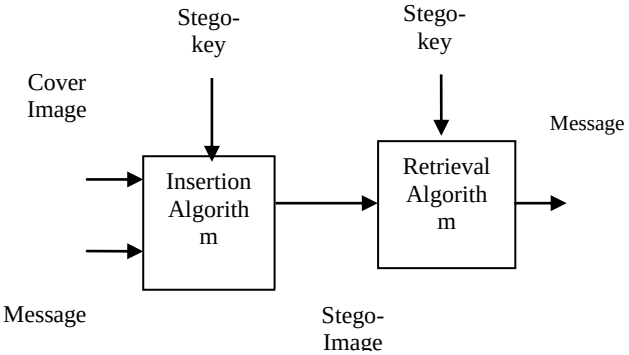

Fig 1: Image Steganography Model

Steganography, in general, have many applications including copyright protection, Feature Tagging and secret communications [4, 5].

- **Copyright Protection:** A secret copyright notice or watermark can be embedded inside an image to identify it as intellectual property. This is the watermarking scenario where the message is the watermark. A watermark can also serve to detect whether the image has been subsequently modified.
- **Feature Tagging:** Captions, annotations, time stamps and other descriptive elements can be embedded inside an image, such as the name of the individuals in a photo or location in a map. Copying the stego image also copies all of the embedded features and only parties who possess the decoding stego key will be able to extract and view the features.
- **Secret Communication:** In many situations, transmitting a cryptographic message draws unwanted attention. The use of cryptographic technology may be restricted or forbidden by law. However, the used steganography does not advertise covert communication and therefore, avoid scrutiny of the sender, message and recipient.

In our work, we changed the pixel value of cover image in accordance with the message. We have masked the message in such a way that changes in the cover image and stego image remains imperceptible to Human Visual System (HVS). Also, our technique is more immune to noise imperfections, steganalysis attacks and compression of cover image because our technique does not include any LSB of pixel value.

The rest of this paper organized as follows: Section 2 reviews various methods of image steganography in spatial domain. Section 3 comprises our proposed method i.e. SSB 7. In Section 4, some experimental results and analysis is listed and discussed. Section 5 provides conclusion of our work and also gives some attention towards future work.

2. METHODS IN SPATIAL DOMAIN

2.1 LSB Method [4]
In this method, least significant bit of pixel value is used for insertion of message. This method is easy to implement but it has many disadvantages associated with it.

- Message can be easily recovered by the unauthorized person as message is in LSB.
- As message is hidden in LSB, so intruder can modify the LSB of all the image pixels in the way the hidden message can be destroyed.
- LSB is most vulnerable to hardware imperfections or quantization of noise.

2.2 6th, 7th bit Method [7]

In this method, Parvinder et al used the 6th & 7th bit for the insertion of message. They didn't use any LSB. They overcome the disadvantages associated with LSB method. But this method also has one disadvantage associated with it. The disadvantage associated with it is that this method provides only the 50% chances of message insertion at a pixel value.

2.3 GLM Method [9]

In 2004, Potdar et al proposes GLM (Gray level modification) technique which is used to map data by modifying the gray level of the image pixels. Gray level modification Steganography is a technique to map data (not embed or hide it) by modifying the gray level values of the image pixels. GLM Steganography uses the concept of odd and even numbers to map data within an image. It is a one-to-one mapping between the binary data and the selected pixels in an image. From a given image a set of pixels are selected based on a mathematical function. The gray level values of those pixels are examined and compared with the bit stream that is to be mapped in the image. Initially, the gray level values of the selected pixels (odd pixels) are made even by changing the gray level by one unit. Once all the selected pixels have an even gray level it is compared with the bit stream, which has to be mapped. The first bit from the bit stream is compared with the first selected pixel. If the first bit is even (i.e. 0), then the first pixel is not modified as all the selected pixels

have an even gray level value. But if the bit is odd (i.e. 1), then the gray level value of the pixel is decremented by one unit to make its value odd, which then would represent an odd bit mapping. This is carried out for all bits in the bit stream and each and every bit is mapped by modifying the gray level values accordingly.

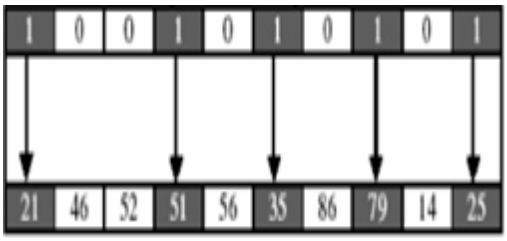

Fig 2: Data Embedding Process in GLM

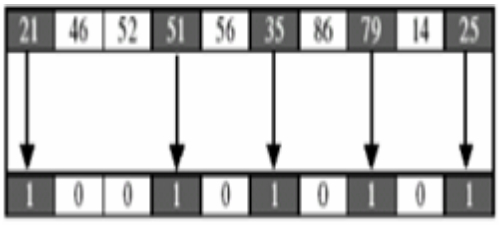

Fig 3: Data Extraction Process in GLM

2.4 6th, 7th & 8th bit Method [8]

In this method 6th, 7th and 8th bits of the image pixels are used to hide the message. Since this method involves 8th bit for hiding the message, intruder can easily change 8th bit of all image pixels and this may result in the loss of message. To avoid this, time factor has been introduced, i.e. at some time t1, sender sends the cover object with message and at some other time t2 sender sends the cover object without message. Sender and recipient agree on this time factor initially before starting any communication. The advantage of introducing time factor (slot) is that if least significant bits of all pixels are changed by the intruder even then the message can be retrieved by comparing the two cover objects, i.e. one containing the message and the other not containing the message.

2.5 Parity Checker Method [10]

In this method, Rajkumar et al gives the concept of odd and even parity. According to this method, 0 can be inserted at a pixel location if that pixel has odd parity i.e. the number of 1's in the binary value of the pixel should be odd. Similarly, 1 can be inserted at a pixel location if that pixel has even parity i.e. the number of 1's in the binary value of pixel should be even. If the corresponding parity does not exist at a pixel location either for 0 or 1, then we make corresponding parity at that pixel location (odd parity for 0 and even parity for 1) by adding or subtracting 1 to the pixel location such that the change in the image quality should not be visible to the human visual system (HVS).

3. DESCRIPTION OF PROPOSED METHOD

In our method, we used the bit 7 of pixel value for insertion of message. We modified the pixel value such that its bit 7 becomes equal to the message bit by only +1 or -1 change in pixel value. We did not used any LSB for insertion of message but manage to maintain a change of +1 or -1 only in pixel value of image. Our method removed the disadvantages associated with LSB method, 6th & 7th bit method, GLM method, 6th, 7th & 8th bit method and parity checker method but retains their advantages.

3.1 Hypothesis and Assertions

SSB-7 is based on following Hypothesis/Assertions:

Hypothesis - 1:
In digital image, small variations in pixel value are imperceptible to human eye. Our Hypothesis is that changing to +1 or -1 unit in the pixel value is imperceptible to HVS.

Assertion – 1:
The 7th bit of pixel value is chosen for insertion of message because it satisfies Hypothesis – 1 and provides minimum change in pixel value i.e. +1 or -1.

Assertion – 2:
The Boundary values of pixel i.e. 0 and 255 will not be used to embed information because they do not satisfy Hypothesis-1.

3.2 Insertion Algorithm

i) Find pseudo random location (L) in cover image from secret key to insert the message bit b. (For detail see [6] and [12]).

ii) Check whether at location (L); pixel value is 00000000 or 11111111. If yes, ignore this location and go to step (i).

iii) Check whether the bit 7 of pixel location (L) is equal to message bit b. If yes, then message is already present at location (L). So, we do not need to change the pixel value and go to END. If NO, then go to next step.

iv) Try to make bit 7 of pixel location (L) equal to message bit b by adding or subtracting 1 to the pixel value.

v) END.

3.3 Retrieval Algorithm

i) Trace out the location (L) from the same secret key as used for insertion of message.

ii) Pixel value is 00000000 or 11111111? If yes, then it is invalid location. Go to step (i).

iii) Check the bit 7 of pixel location (L). If it is 0 then 0 is the message bit else 1 is the message bit.

iv) END.

3.4 Practical Example

Suppose the message bit is zero and pixel value of selected location is 174. So, the value of bit 7 is 1. We have to make this bit 7 equal to 0 by adding or subtracting 1. We can make the above adjustment by subtracting 1 from the pixel value of selected location. So, the modified pixel becomes 173 after insertion of 0.

Table 1: A – Original pixel value, B - Modified pixel value after insertion of 0

	Decimal Value	Binary Value							
		1	2	3	4	5	6	7	8
A →	174	1	0	1	0	1	1	***1***	0
B →	173	1	0	1	0	1	1	***0***	1

4. RESULTS

4.1 Comparison with other methods based upon chances of message insertion at a pixel value:

Now, we see how various pixel values chances during insertion of message. Table II shows how pixel values changes during insertion of 0 and Table III shows how pixel values changes during insertion of 1.

Table – II

Decimal Value	Pixel value before insertion of '0'	Pixel value after insertion of '0'	Change in Pixel value & comment for insertion of '0'
0	00000000	00000000	BV, Ignore
1	00000001	00000001	NC, Insert
2	00000010	00000001	-1, Insert
3	00000011	00000100	+1, Insert
4	00000100	00000100	NC, Insert
5	00000101	00000101	NC, Insert
6	00000110	00000101	-1, Insert
7	00000111	00001000	+1, Insert
8	00001000	00001000	NC, Insert
9	00001001	00001001	NC, Insert
10	00001010	00001001	-1, Insert
11	00001011	00001100	+1, Insert
12	00001100	00001100	NC, Insert
13	00001101	00001101	NC, Insert
14	00001110	00001101	-1, Insert
15	00001111	00010000	+1, Insert
.	.	.	.
.	.	.	.
.	.	.	.
127	01111111	10000000	+1, Insert
128	10000000	10000000	NC, Insert
.	.	.	.
.	.	.	.
.	.	.	.
254	11111110	11111101	-1, Insert
255	11111111	11111111	BV, Ignore

Table – III

Decimal Value	Pixel value before insertion of '0'	Pixel value after insertion of '0'	Change in Pixel value & comment for insertion of '0'
0	00000000	00000000	BV, Ignore
1	00000001	00000010	+1, Insert
2	00000010	00000010	NC, Insert
3	00000011	00000011	NC, Insert
4	00000100	00000011	-1, Insert
5	00000101	00000110	+1, Insert
6	00000110	00000110	NC, Insert
7	00000111	00000111	NC, Insert
8	00001000	00000111	-1, Insert
9	00001001	00001010	+1, Insert
10	00001010	00001010	NC, Insert
11	00001011	00001011	NC, Insert
12	00001100	00001011	-1, Insert
13	00001101	00001110	+1, Insert
14	00001110	00001110	NC, Insert
15	00001111	00001111	NC, Insert
.	.	.	.
.	.	.	.
.	.	.	.
127	01111111	01111111	NC, Insert
128	10000000	01111111	-1, Insert
.	.	.	.
.	.	.	.
.	.	.	.
254	11111110	11111110	NC, Insert
255	11111111	11111111	BV, Ignore

From Table II & III we can calculate the following:

i) Chances of message insertion at a pixel value

= (pixel values where we can insert message/total possible values of a pixel) * 100

= (254/256) * 100

= 99.21 %

ii) Chances when no change in pixel value is required after insertion of message

= (pixel values where no change is required after insertion of message/total pixel values where we can insert the message) * 100

= (127/254) * 100

= 50%

The comparison table of our method with 6^{th}, 7^{th} bit method and 6^{th}, 7^{th} & 8^{th} bit method is shown below:

Table IV

Method	Message bit Insertion at pseudo random location at first chance	No change in pixel value when message bit is inserted
6^{th} & 7^{th} bit	50%	50%
6^{th}, 7^{th} & 8^{th} bit	85.93%	43.18%
SSB – 7	99.21%	50%

4.2 Subjective Test

Detection of the message from the stego image is the primary hurdle in the path of Steganography. For checking whether our method overcomes this hurdle we apply subjective test. The subjective tests are made by people who look for visual differences between images (cover image and stego image) trying to find out which one is original. If the percentage of success goes above 80%, we took five images of same size and hide the different length data into those images. In the subjective tests, each two images i.e. cover image and stego image we presented to 50 analyzers with the following question:

"Which one is the original image"? The result are shown in Table V

Table V

Image	Image Size (Pixels)	Message Size (bits)	Success (%)
Picture – 1	128400	2048	89
Picture – 2	128400	16384	85
Picture – 3	128400	65536	94
Picture – 4	128400	67832	92
Picture – 5	128400	129400	86

Analyzing the results from Table V, we can conclude that, SSB – 7 did not generate any indication of correct identification of the original image. Figure 4 shows the original image in which we want to hide the data. Figure 5 is stego image after insertion of 16384 bit data. Figure 6 shows the difference of Figure 4 and Figure 5.

Fig 4: Original Image

Fig 5: Stego Image after Insertion of 16, 384 Bit Message

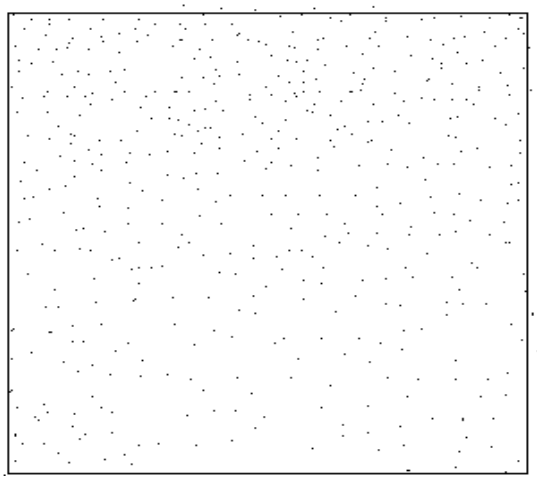

Fig 6: Difference of Figure 4 & 5

4.3 Histogram Analysis

Figure 7 shows the histogram of original image given in Figure 4. Figure 8 shows the histogram of stego image given in Figure 5. The histograms of images show great sequenced peaks which show the presence of large amount of information. For less amount of information the change in the histograms will be less.

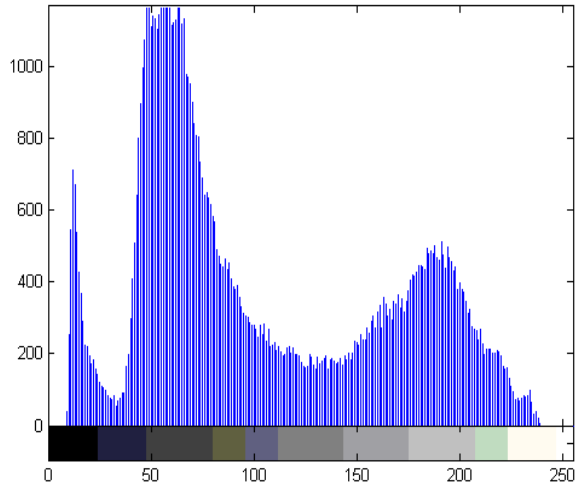

Fig 7: Histogram of original image given in Figure 4

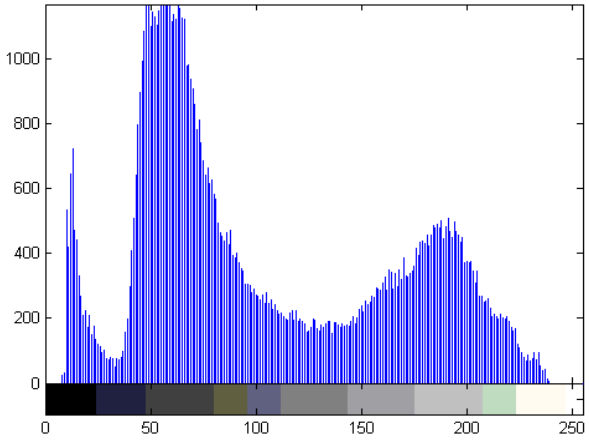

Fig 8: Histogram of stego image given in Figure 5

4.4 MSE and PSNR comparison with other methods

We have applied the MSE (Mean Square Error) and PSNR (Peak Signal to Noise Ratio) to compare SSB – 7 with other methods. MSE and PSNR are calculated with the following formulae:

$$MSE = \frac{1}{[n \times m]^2} \sum_{i=1}^{n} \sum_{j=1}^{m} (x_{ij} - y_{ij})^2 \text{ ---------(1)}$$

$$PSNR = 10 \log_{10}[(255)^2 / MSE] \text{ ----------(2)}$$

where n is the number of rows in the image matrix and m is the number of columns in the image matrix.

The results on three images are shown in Table VI

Table VI

Method	Picture – I	Picture – II	Picture – III
SSB – 7	42.4	44.0	44.6
LSB Method	32.1	33.4	32.7
6^{th} & 7^{th} bit Method	32.8	31.2	33.8
GLM Method	33.4	35.3	36.1
Parity Checker Method	30.3	28.4	29.0

From Table VI we conclude that our method provide better PSNR values than other methods.

5. CONCLUSION

This work presents a new spatial domain technique for image steganography. SSB–7 hides the data using the bit 7 and the change in the pixel value is minimal. SSB–7 is immune to noise imperfections and compression techniques because it does not include any LSB. We showed that our method provide no clue to the intruder to identify the difference between original image and stego image. Future work will concentrate on applying this technique in frequency domain and improving its robustness.

6. REFERENCES

[1] A. Gutub, M. Faltani, "A Novel Arabic Text Stegnography Method Using Letter Points and Extension", WASET International Conference on Computer Information and System Science and Engineering (ICCISSE), Vienna, Austria, May 25-27, 2007.

[2] RJ Anderson, FAP Petitcolas, "On the Limits of Stegnography", IEE Journal on selected Areas in Communications, Vol. 16 No 4, pp 474-481, May 1998.

[3] Johnson, Neil F., Zoran Duric, S. G. J., Information Hiding: Steganography and Watermarking – Attacks and Countermeasures (Advances in Information Security, Volume I). Kluwer Academic Publishers, February 15, 2001.

[4] Neil F Johnson, Sushil Jajodia, "Exploring Stenography: Seeing the Unseen", IEEE Computer, Feb 1998, pp 26-34.

[5] W Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding," IBM Systems Journal, Vol. 35, No. 3 and 4, pp. 313-336, 1996.

[6] E Franz, A Jerichow, S Moller, A Pfitznaun, I Stierand, "Computer Based Stegnography", Information Hiding, Springer Lecture Notes in Computer Science, Vol. 1174, pp 7-21, 1996.

[7] Parvinder Singh, Sudhir Batra, HR Sharma, "Evaluating the Performance of Message Hidden in 1^{st} and 2^{nd} Bit Plane", *WSEAS Transactions on Information Science and Applications*, issue 8, vol 2, Aug 2005, pp 1220-1227.

[8] Sudhir Batra, Rahul Rishi, Rajkumar , "Insertion of Message in 6^{th} , 7^{th} and 8^{th} Bit of Pixel Values and Its Retrieval in Case Intruder Changes the Least Significant

Bit of Image Pixels", International Journal of Security and its Applications, issue 3, vol, 4, July 2010, pp 1-10.

[9] Vidyasagar M. Potdar, Elizabeth Chang, "Grey Level Modification Stegnography for Secret Communication", 2nd IEEE International Conference on Industrial Informatics INDIN 2004 June 24th, 26th June, Berlin, Germany, Submitted Tuesday, May 25, 2004.

[10] Yadav, Rajkumar, Rishi, Rahul, Batra, Sudhir, "A new Steganography Method for Gray Level Images using Parity Checker", International Journal of Computer Applications (0975-8887) Volume 11-No. 11, December 2010.

[11] Jessica Fridrich, Miroslav Goljan , Rui Du, "Detecting LSB Steganography in Color and Gray-Scale Images", IEEE Multimedia, issue 4, vol 8, 2001.

[12] Yeuan-Kuen Lee, Ling-Hwei Chen, "A Secure Robust Image Stegnography Model", 10th National Conference on Information Security, Hualien, Taiwan, pp 275-284, May 2000.