Insertion of message in 6th, 7th and 8th bit of pixel values and its retrieval in case intruder changes the least significant bit of image pixels

Sudhir Batra[1], Rahul Rishi[1], and Rajkumar[2]
[1] *Technological Institute of Textile and Sciences, Bhiwani-127021 (India)*
[2] *University Institute of Engineering & Technology,*
M. D. University, Rohtak-124001 (India)
batrasudhir@rediffmail.com, rahulrishi@rediffmail.com,
rajyadav76@rediffmail.com

Abstract

A new method for insertion of message using 6th, 7th and 8th bits of pixel values of an image is developed. This method is an improvement over the earlier method [7] in which 6th and 7th bits of pixel values of an image were used. A method to retrieve the message is also given. Another advantage offered by this technique is that the message can be retrieved even if the intruder changes the least significant bit of all the image pixels in which message has been embedded.

1. Introduction

Steganography is a technique used for hiding the information. It conceals the existence of message whereas cryptography on the other hand conceals the contents of the message. Using this, the sender starts with a cover object, which can be an image, video, music or any other computer file, and embeds a hidden message into the cover object by slightly distorting it in a way that enables the intended recipient to retrieve the hidden message from the distorted object and simultaneously concealing the hidden message from the third party [1, 2]. This idea was first described by simmons [6].

The popular and oldest method for hiding the message in a digital image is the LSB method. In LSB method we hide the message in the least significant bits (LSB's) of pixel values of an image[4]. One of the major disadvantage associated with LSB method is that intruder can change the least significant bit of all the image pixels. In this way hidden message will be destroyed by changing the image quality, a little bit, i.e. in the range of +1 or -1 at each pixel position. Singh, Batra and Sharma [7] obtained an algorithm to hide the message in 6th and 7th bit of pixel value. This algorithm overcomes the above disadvantage associated with the least significant bit insertion method. However, it has its own disadvantage, that is the chance that the message bit will be inserted at pseudorandom location at first instance is only 50%. In this paper, an insertion method is obtained which increases the chance of insertion at first instance to 85.93% and includes all advantages offered in [7]. 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 t_1, sender sends the cover object with message and at some other time t_2 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. Methods for insertion / retrieval of message in case intruder does not manipulate the recipient copies are given in section 2 in the form of algorithms. The steps explaining how to retrieve the message in case intruder changes the least significant bit of pixels of recipient copies have been discussed in section 3.

2. Description of the Model and Algorithms

2.1 Sender and Recipient agree on a model using which message communication is to be taken place.

This Model is described as under:

1) Sender and Recipient agree on the cover object in which message is supposed to be hidden.
2) They share a secret key to decide the sequence of random locations in the cover object (for detail see [3] and [5]).
3) Recipient knows the algorithm using which sender hides the message.
4) Intruder can change least significant bit of all pixel values of cover object and that only once, i.e. either in the cover object containing the message or in the cover object not containing the message.
5) Sender and recipient fix some time slots in which cover object contains the message. Sender sends the cover object without message at some time t_1 and the cover object with message at some other time t_2.

2.2 Algorithm for insertion of message bit b using 6^{th}, 7^{th} and 8^{th} bit of the pixel value.

Let b^c denote the complement of b, i.e. if b=0 then b^c=1 and if b=1 then b^c=0.

(i) Find pseudo- random location (*l*) in the cover object using secret key for inserting the message bit 'b' (i.e. 0 or 1)

(ii) Check whether at location (*l*) pixel value is, 00000000, 00000001, 11111110 or 11111111. If yes, then this is an invalid location (for reasoning see remark 3.3) and therefore go to (i).

(iii) Check whether at location (*l*);

(a) 6^{th}, 7^{th} and 8^{th} bits are b b b, b b b^c or b^c b b?

If yes, no change at (*l*) is required. Message bit is already there, go to end.

(b) 6^{th}, 7^{th} and 8^{th} bits are b b^c b^c, b^c b^c b or b^c b^c b^c ?

If yes, then make these b b b, b b b^c or b^cb b by adding or subtracting 1. However, if it is not possible to make these as b b b, b b b^c or b^c b b by adding or subtracting 1, then make these b^c b b^c or b b^c b and go to end.

(c) 6^{th}, 7^{th} and 8^{th} bits are b^c b b^c or b b^c b?

If yes, then make these b b b, b b b^c, or b^c b b by adding or subtracting 1, otherwise go to end.

(iv) End.

2.3 Algorithm for retrieval of message bit *b as bbb or bbb^c or b^{c}bb*

(i) Trace out the location "*l*" from the same secret key which was used during insertion process by the sender.

(ii) Check whether pixel value is equal to 00000000, 00000001, 11111110 or 11111111? If yes, then it is an invalid address. Go to (i)

(iii) Check whether at location (*l*);

(a) 6th, 7th and 8th bits are b^c b b^c or b b^c b?
If yes, then no message bit has been inserted and therefore go to (i)

(b) 6th, 7th and 8th bits are b b b , b b b^c or b^c b b.
then 'b' is the message bit.

(iv) End.

2.3 Here we see in view of algorithm 2.2 that (i) how the various pixel values are changed during insertion process? (ii) Which of the locations should be used for the insertion which should be ignored?

The following results obtained from this table can somehow be used for assessing the change in the cover object after the insertion of message. However this requires further study.

1. The probability that the message bit will be inserted at the pseudorandom location at first chance
   = 440/512 *100 = 85.93%
2. The probability that when message is inserted, without changing the pixel values (excluding the locations which are ignored or invalid) =95/220*100=43.18%
3. The probability that change in pixel value is required, when we are ignoring the location, =31/256*100 = 12.10%

For this, table (I) is given below:

Table (I)

Decimal Value	Pixel Value Before Insertion	Pixel Value After the possible Insertion of '0'	Change in Pixel value and Comment for the Insertion of '0'	Pixel value After the possible Insertion of '1'	Change in Pixel value and Comment for the Insertion of '1'
0	00000000	00000000	No Change Invalid Location	00000000	No Change Invalid Location
1	00000001	00000000	-1, Invalid Location	00000000	-1, Invalid Location
2	00000010	00000001	-1, Insert	00000011	+1 Insert
3	00000011	00000100	+1, Insert	00000011	NC, Insert

4	00000100	00000100	NC, Insert	00000011	-1 Insert
5	00000101	00000100	-1, Insert	00000110	+1 Insert
6	00000110	00000101	-1,Ignore	00000110	NC, Insert
7	00000111	00000100	+1, Insert	00000111	NC, Insert
8	00001000	00001000	NC, Insert	00000111	-1, Insert
9	00001001	00001001	NC, Insert	00001010	+1 Ignore
10	00001010	00001001	-1, Insert	00001011	+1 Insert
11	00001011	00001100	+1, Insert	00001011	NC, Insert
12	00001100	00001100	NC, Insert	00001011	-1, Insert
13	00001101	00001100	-1, Insert	00001110	+1 Insert
14	00001110	00001101	-1, Ignore	00001110	NC, Insert
15	00001111	00010000	+1, Insert	00001111	NC, Insert
16	00010000	0001000	NC, Insert	00001111	-1, Insert
17	00010001	00010001	NC, Insert	00010010	+1, Ignore
18	00010010	00010001	-1, Insert	00010011	+1 Insert
19	00010011	00010100	+1, Insert	00010011	NC, Insert
20	00010100	00000001	NC, Insert	00010011	-1 Ignore
21	00010101	00010100	-1, Insert	00010110	+1 Insert
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
127	01111111	10000000	+1, Insert	01111111	NC, Insert
128	10000000	10000000	NC, Insert	01111111	-1,Insert
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
254	11111110	11111111	+1, Invalid Location	11111111	+1 , Invalid Location
255	11111111	11111111	No Change Invalid Location	11111111	No Change Invalid Location

Note: In all the tables appearing in this paper 'NC' means no change is required.

3 Retrieval of message in case intruder changes least significant bits of pixel values of a recipient copy.

For convenience, locations with pixel values corresponding to decimal values 0, 1, 2, 3,…, 255 will be called locations 0, 1, 2, 3,…, 255 respectively. In view of fifth assumption of the model described in section 2, recipient receives two copies of the cover object one without message and the other with message. Further, in view of the fourth assumption of the model the following two cases arise:

(I) Intruder changes the least significant bits of pixel values of the cover object with message.

(II) Intruder changes the least significant bits of pixel values of the cover object without message.

Furthermore suppose that recipient does not know initially that which copy among the two copies of cover object received by him has been manipulated by the intruder.

Next we discuss the retrieval of message in both the above cases separately in 3.1 and 3.2.

3.1 Here we consider the case in which intruder changes the least significant bit of pixel values of the cover object with message bit '0'.

For retrieval of message, consider the following table (II) derived from table (I), in which pixel values of the cover object in different stages have been described.

Table (II)

Decimal value	**Pixel value before insertion (Recipient copy C_1)**	**Pixel value after the possible insertion of '0'**	**Pixel value after insertion with changed LSB's (Recipient copy C_2)**	**Net change, i.e. pixel value in C_2 - pixel value in C_1**
2	00000010	00000001	00000000	-2
3	00000011	00000100	00000101	+2
4	00000100	00000100	00000101	+1
5	00000101	00000100	00000101	No change
6	00000110	00000101	00000100	-2
7	00000111	00001000	00001001	+2
8	00001000	00001000	00001001	-1
9	00001001	00001001	00001000	+1
0	00000000	00000000	00000001	Invalid Location
1	00000001	00000000	00000001	Invalid Location
254	11111110	11111111	11111110	Invalid Location
255	11111111	11111111	11111110	Invalid Location

Pixel values of the two copies of the cover object received by the recipient are listed in columns II and IV of the above table. We denote these columns II and IV by C_1 and C_2 respectively. Note that we will discuss the locations 2, 3, ..., 8, 9, 0, 1, 254, 255 only. The discussion for the remaining locations 2+8i, 3+8i, …, 8+8i, 9+8i, $1 \leq i \leq 31$ is same as for the locations 2, 3, …, 8, 9 respectively. We now compare the pixel values in C_1 and C_2 and observe the following:

(i) In C_2 at locations 4 and 5, 6^{th}, 7^{th} and 8^{th} bits are respectively 101 and 101. In view of algorithm 2.3, no message bit is retrieved. But looking at the corresponding values in C_1, i.e. 100 and 101, we can say that in view of algorithm 2.2, the message bit should have easily been inserted in the following manner:

(a) At location 4, 100 to be retained as 100 for the insertion of '0' and 100 to be changed to 011 for inserting '1'.

(b) At location 5, 101 to be changed to 100 for inserting '0' and 101 to be changed to 110 for inserting '1'. In view of this discussion, we can say that intruder has changed the least significant bit of all locations of cover object with message or of cover object without message.

To nullify this effect of intruder, let us change the least significant bit of locations 4 and 5 in C_1. Now comparing these locations in C_1 with those in C_2, we observe

that roles of locations 4 and 5 have been interchanged. Therefore, the above discussion in (a) and (b) is valid in this case also. Hence to nullify the effect of intruder, we change the least significant bits of location 4 and 5 in C_2 and then using algorithm 2.3, '0' message is retrieved from both these locations.

(ii) In C_2 at locations 8 and 9, 5^{th}, 6^{th}, 7^{th} and 8^{th} bits are respectively 1001 and 1000. In view of algorithm 2.3, '0' bit is retrieved from these locations. But observing the corresponding values in C_1, i.e. , 1000 and 1001 we can say that there was no need of changing 1000 to 1001 and 1001 to 1000 for the insertion of '0'. For inserting '1', 1000 should have been changed to 0111 and 1001 should have been ignored after changing it to 1010. In view of this discussion, we can say that intruder has changed the least significant bit of pixels in C_1 or C_2. To nullify this effect of intruder, we change the least significant bit of locations 8 and 9 in C_1 or C_2. It then follows that, in any case, message '0' bit is retrieved.

(iii) At locations 2, 3, 6 or 7 if we compare the copy C_1 with C_2, we see the net change in the values of pixels is +2 or -2, which shows that intruder has changed the least significant bits. In order to nullify this, we change least significant bit of pixels values of C_1 at these locations and then compare the corresponding values of C_1 and C_2. The net changes are then +3 or -3. Therefore, we change least significant bit of pixel values in C_2 and then compare the corresponding pixel values in C_1 and C_2. In this way we see that '0' bit is retrieved from locations 2, 3 and 7 and from location 6 no message bit is retrieved.

(iv) In C_2 at location 0 and 1, the pixel values are 00000001 and 00000001 and both of these are invalid locations. Also if we compare pixel values at these locations in C_1 and C_2, it can be easily seen that these are invalid locations and no message bit is retrieved from these locations. Similarly, from locations 254 and 255, no message bit is retrieved.

3.2 Here we consider the case in which intruder changes the least significant bits of pixel values of the cover object without message.

For retrieval of message, consider the following table (III) in which pixel values of the cover object in different stages have been described.

Table- III

Decimal value	**Pixel value before insertion**	**Pixel value before insertion with changed LSB's (Recipient copy C_1)**	**Pixel value after insertion of '0' (Recipient copy C_2)**	**Net change, i.e. pixel value in C_2 - pixel value in C_1**
2	00000010	00000011	00000001	-2
3	00000011	00000010	00000100	+2
4	00000100	00000101	00000100	-1
5	00000101	00000100	00000100	No Change
6	00000110	00000111	00000101	-2
7	00000111	00000110	00001000	+2
8	00001000	00001001	00001000	-1
9	00001001	00001000	00001001	+1
0	00000000	00000001	00000000	Invalid Location

1	00000001	00000000	00000000	Invalid Location
254	11111110	11111111	11111111	Invalid Location
255	11111111	11111110	11111111	Invalid Location

Pixel values of the two copies of the cover object received by the recipient are listed in columns III and IV of the above table. We denote these columns III and IV by C_1 and C_2 respectively. Note that we will discuss the location 2, 3, ..., 8, 9, 0, 1, 254, 255 only. The discussion for the remaining locations 2+8i, 3+8i,…,8+8i, 9+8i, $1 \leq i \leq 31$ is same as for the locations 2, 3, …, 8, 9 respectively. We now compare the pixel values in C_1 and C_2 and observe the following:

(i) In C_2 at locations 4 and 5, 6^{th} ,7^{th} , 8^{th} bits are 100 and 100 respectively. In view of algorithm 2.3 message bit '0' is retrieved. To check the role of intruder, we compare the pixel values in C_1 and C_2 at these locations. It then follows in view of algorithm for insertion that nothing wrong has been done by the intruder and by algorithm 2.3 message bit '0' is retrieved . Here note that even intruder has changed the least significant bit at these locations, the message bit '0' is retrieved without doing any other treatment, because the same message bit '0' is retrieved when we compare pixel values at these locations given in columns II and IV.

(ii) At locations 2, 3, 6 and 7 the net changes are -2 and +2. This shows that intruder has changed the least significant bit of pixel values at these locations. In order to nullify the changes made by the intruder, we change least significant bit of all pixel value in column C_2 and then compare it with column C_1 .We see that the net changes are -3 or +3 now. Therefore, let us change least significant bit of pixel values in C_1 and then compare this with C_2. Hence from locations 2, 3 and 7 using algorithm 2.3, message bit '0' is retrieved and from location 6 no message bit is retrieved

(iii) On comparing the pixel values at locations 8 and 9 in C_1 and C_2, we see that 001 has been changed to 000 and 000 has been changed to 001. But for the insertion of '0' there was no need of changing 001 to 000 and 000 to 001 at locations 8 and 9. Further, for the insertion of '1', location 8 should have been ignored by changing 1001 to 1010 in C_1 and at location 9, 1000 should have been changed to 0111. This shows that intruder has changed the least significant bit of pixel values at some stage. To nullify this effect, we change the least significant bit of these locations in C_1 or C_2 and in any case message bit '0' is retrieved.

(iv) It can be seen easily that from locations 0, 1, 254, 255 no message bit is retrieved.

3.3 Remark

(i) At location 0, message '0' bit can be inserted. But if we wish to insert '1' at this location then the minimum change required is +3, therefore we would like to ignore this location for the insertion of '1'. The minimum change required for ignoring this for the insertion of 1 is +2. Hence we ignore this location for both '0' and '1' by retaining its pixel value 00000000 and we call this location as invalid location. Due to the same reasoning we ignore location 255.

(ii) At location 1, suppose message '0' has been inserted using algorithm 2.2. Further, suppose that intruder has changed 00000001 to 00000000 in C_1 or C_2. This will give wrong impression to the recipient that intruder has changed the invalid location 00000000 (discussed in (i) above) to 00000001 in C_1 or C_2. Hence location *l* is ignored for both '0' and '1' by changing its pixel value as 00000000 and treat this location also as invalid location. Similarly, we ignore location 254 by changing its pixel value as 11111111 and treat this also as invalid location.

3.3 So far, we discussed the case of insertion and retrieving of the message bit '0' in 3.1 and 3.2.

Similarly, we can discuss the insertion and retrieving of the message bit '1'. For this, two tables IV and V analogous to tables (II) and (III) are given below.

Table IV

Decimal Value	**Pixel Value Before Insertion (Recipient Copy C_1)**	**Pixel Value After possible Insertion of '1'**	**Pixel value after Insertion with Changed LSB's (Recipient Copy C_2)**	**Net change, i.e. pixel value in C_2 - pixel value in C_1**
2	00000010	00000011	00000010	NC
3	00000011	00000011	00000010	-1
4	00000100	00000011	00000010	-2
5	00000101	00000110	00000111	+2
6	00000110	00000110	00000111	+1
7	00000111	00000111	00000110	-1
8	00001000	00000111	00000110	-2
9	00001001	00001010	00001011	+2
0	00000000	00000000	00000001	Invalid Location
1	00000001	00000000	00000001	Invalid Location
254	11111110	11111111	11111110	Invalid Location
255	11111111	11111111	11111110	Invalid Location

Table V

Decimal Value	**Pixel Value Before Insertion '1'**	**Pixel Value Before Insertion with Changed LSB's (Recipient Copy C_1)**	**Pixel value after the possible Insertion of 1. (Recipient Copy C_2)**	**Net change, i.e. pixel value in C_2 - pixel value in C_1**
2	00000010	00000011	00000011	NC
3	00000011	00000010	00000011	+1
4	00000100	00000101	00000011	-2
5	00000101	00000100	00000110	+2
6	00000110	00000111	00000110	-1
7	00000111	00000110	00000111	+1
8	00001000	00001001	00000111	-2
9	00001001	00001000	00001010	+2
0	00000000	00000001	00000000	Invalid Location
1	00000001	00000000	00000001	Invalid Location
254	11111110	11111111	11111110	Invalid Location
255	11111111	11111111	11111110	Invalid Location

References

1. RJ Anderson , " Stretching the Limits of Steganography", In Information Hiding ,Springer Lecture Notes in Computer Science, vol 1174, pp 39-48, 1996.
2. RJ Anderson and FAP Petitcolas, "On the Limits of Steganography", IEE Journal on Selected Areas in Communications, vol 16, no 4, pp 474-481, May 1998.
3. E Franz, A Jerichow, S Moller, A Pfitzmann and I Stierand, "Computer Based Steganography", Information Hiding, Springer Lecture Notes in Computer Science vol 1174, pp 7-21, 1996.
4. Neil F Johnson and Sushil Jajodia, "Exploring Steganography: Seeing the Unseen", IEEE Computer, pp 26-34, Feb 1998.
5. Yeuan- Kuen Lee and Ling-Hwei Chen, "A Secure Robust Image Steganographic Model, 10th National Conference on Information Security, Hualien, Taiwan, pp 275-284, May 2000.
6. GJ Simmons, "The Prisoners Problem and the Subliminal Channel", Proceeding of Crypto'83, Pleneum Press, pp 51-67, 1983.
7. Parvinder Singh, Sudhir Batra and H.R Sharma, "Evaluating the Performance of Message Hidden in First and Second Bit Plane", WSEAS Transaction on Information Science and Technology, vol. 2, no. 8, pp 1220-1222, Aug. 2005.

Authors

Sudhir Batra received his M.Sc. from Maharshi Dayanand University in 1994. In 1994 he received junior research fellowship from the Council of Scientific and Industrial Research, New Delhi. In 1996, he received senior research fellowship from the same agency. In 2001, he received his Ph.D. from Mahrashi Dayanand University. Since 1997, he has been at the Technological Institute of Textile & Sciences, Bhiwani (Haryana). His research areas of interest are Group Rings, Algebraic Coding Theory and Theory of Finger-printing Codes.

Rahul Rishi received his B.Tech in Computer Science from Maharshi Dayanand University in 1995. In 2005, he received his Ph.D. from Mahrashi Dayanand University. Since 1997, he has been at the Technological Institute of Textile & Sciences, Bhiwani (Haryana). His research areas of interest are Uncertainty Management through Relational Databases, Information Security and Theory of Finger-printing Codes.

Raj Kumar received the M.Tech degree in Computer Science and Engineering from Kurukshetra University Kurukshetra (INDIA) in 2002. He is currently working in Computer Science and Engineering Department at University Institute of Engineering & Technology (M.D. University Rohtak, Haryana) India. His research interest includes Information Hiding Techniques, Network Security and Biometrics