Multi-Segment Steganography Technique

Fayik Alnawok and Basem Ahmed
Faculty of Applied Science, Al Aqsa University, Palestine

Abstract: *The idea of this paper is to invent a new strategy in steganography to get the minimum effect on the images which is used to hide data into it. This is by dividing the original image into a number of segments, these segments is achieved according to the number of characters included into the message which is going to be hided into the original image. In this approach the message will be coded by using the coding table. After the massage has been coded, it will be hidden into the image. The new technique is starting to search segments in the image that corresponded to the coded characters, in this stage the technique is mark out the positions of each encoded character included in the original message. At last we tried to test the new technique we found that the positions of the character in image doesn't been affected, beside that the new technique is also hard to been beaked by uninterested users.*

Keywords: *Steganography, coding theory.*

Received July 1, 2006; accepted February 20, 2007

1. Introduction

The earliest records of steganography go back to the 5th century B.C. Historian herodotus reports that during sieges, secret messages were tattooed to slaves. Shaved heads whom were then dispatched when their hair grew back [1]. The word steganography comes from a greek words. Steganos means (covered) and graptos means (writing).

In that period ancient greeks would use wax tablets much as we use paper to record ephemeral notes. These tablets were mounted backboards made of wood. A greek-born exile in persia was able to inscribe a warning of the mounting attack on the wood and cover it with what appeared to be a simple blank wax tablet. Computer steganography is based on two principles:

- Digitized information can be altered to a certain extent while retaining its original functionality.
- Humans are unable to distinguish minute changes in text, images or sound [2].

The study of this subject in the scientific literature may be traced to simmons, who in 1983 formulated it as the "*Prisoners' Problem*" [3].

There are many real life applications of steganography. Apparently, during the 1980's, Margaret thatcher became so irritated at press leaks of cabinet documents that she had the word processors programmed to encode their identity in the word spacing, so that disloyal ministers could be traced [4].

May we have some confusion between cryptography, steganography, watermarking and fingerprinting so we will start by distinguishing between them. Massage can be scramble or camouflages, scramble the message called cryptography, while camouflages the message divided into two kinds the first by hidden the message which called steganography the second without hidden the message which is used for document marking which also divided into watermarking and fingerprinting as purpose of embedding copyright (i.e., protecting authorship).

Digital steganography is used to hide messages in three media: text, sound and images. Steganography in text can be by adding spaces so that double space mean one and single space mean zero so these spaces can be generate words. Steganography in audio includes techniques such as low bit encoding and phase coding. Low bit encoding involves encoding the message in the least significant bits of the carrier file.

Steganography in images includes techniques such as Least Significant Bit insertion (LSB). In this paper we are going to deal with steganography on images.

2. Literature View

There are several works in the field of steganography but most of them were deal with methods for hiding a secret message in a media or deal with how to detect a hidden message which has been in a media like sound, video or images. Since our paper shows a new strategy for hiding a secret message into an image with a minimum effect on the original image. In this section we are going to explain several techniques which have been used in steganography.

2.1 Least Significant Bit Insertion

This method store one bit of information in the least bit of each byte, since in image each pixel represented by three byte so we can store three information bits in each pixel, if the letter F is to be hidden in an image

that is totally white the first three pixels would have the following binary code (knowing that the binary representation of the letter F is *01000110*): the next pattern shows the way how it will be presented *(00000000 00000001 00000000) (00000000 00000000 00000001) (00000001 00000000 00000000)* [2].

2.2 Direct-Sequence Spread Spectrum

The modulation function consists of a constant, integral valued gain factor G multiplied by a pseudo-random block φ_i of +1 and -1 values. Each block φ_i has a distinct location in the (x, y) plane. The use of blocks is not a necessity. In some applications, it might be desirable to interleave the carriers so that each one is spread over the entire area of the image. The embedded data is recovered by demodulating with the original modulating function. A *TRUE* (+1) bit appears as a positive correlation value; a *FALSE* (-1) bit is indicated by a negative correlation value. Once the carrier phase has been recovered, we project the stego-image onto each basis vector φ_i:

$$O_i = (D,\varphi_i)=1/n\sum D(x,y)\ \varphi_i(x,y) \quad (1)$$

and then threshold the o_i values to recover the bits b_i [2].

2.3 Geometric Interpretation

This method is adding a factorial component to the image, and represents information in the angle (typically 0 or 180 degrees), between this component and a known external reference vector [6].

2.4 Novel

Assume that we want to embed a secret image *S* into a cover image *C* and produce a stego-image *C*. In the image embedding module, we apply Vector Quantization (VQ) [7, 8] to compress the secret image *S* and then generate a set of indices for *S* after the process of image compression. Then, we encrypt the set of indices by Data Encryption Standard (DES) and apply Discrete Cosine Transform (DCT) [9] on the cover image *C*. Finally, we embed the set of encrypted indices into the DCT coefficients of the cover image *C* for obtaining a stego-image *C*. On the other hand, in the image extracting module, at first, we extract the set of encrypted indices from the DCT coefficients of the stego-image *C*. Next, we decrypt the set of encrypted indices and then apply VQ on the set of decrypted indices to decompress the secret image *S*.

2.5 Using luminance in Steganography

Colors can be described by the three parameters luminance, hue and saturation. Changes in luminance are better perceptible than changes in chrominance information; the embedding function must preserve the luminance values of the picture and change only the hue and saturation information [11].

From the literature review we conclude that the researches are differ according to the ways they think and the point of view to the problem, so here we developed a new multi segment steganography technique.

3. The Research Method

Any steganography technique consists of two functions:

- Hiding the message into the image.
- Retrieving the message from the image.

3.1 Hiding the Message

Initially, we have the image, the message which is going to be hidden and a secret password which must be known by the receiver to read the message which has been hidden in the image.

According to the English language there is 24 (from *a* to *z*) characters, in case of writing any message we use the above 24 characters plus the ten digit (from 0 to 9) and the space character, so totally we get 35 characters.

The idea that each character in the message is encoded in a byte, so we can represent 255 characters. Each character can be coded by six random code and these codes will be stored in table called code table known by each sender and receiver.

The password will be coded according to the first code of the six random codes in the code table of each character, now each character of the message coded into a random code of its code table.

To put the coded character into it's appropriated position in its segment in the image. Let the chosen code be *X* then we start to search the image from the byte *B*10* about the value equal to *X*, where *B* is the code of the i^{th} character from the password where

$$i = \text{character number in the message mod password length} \quad (2)$$

Let us find it at the byte *B2* when we find it we mark the bytes *B2-3* and *B2+3* since each pixel represented by three bytes we change on the same color of the three colors RGB. Let now explain how to mark the bytes *B2-3* and *B2+3*, we change the value of these two bytes into one of the three values *X*, *(2/3)*X*, or *(4/3)*X* depend on which one closer to the original value of each byte of *B2+3* and *B2-3*.

3.2 Retrieving the Message

The receiver must know the secret password and the codes table, now he start to search from the byte *B*10* about a byte *B2* where the value of bytes *B2-3* and *B2+3* equal to one of the values *X*, *(2/3)*X*, or *(4/3)*X*,

where B is the code of the i^{th} character from the password where i is from equation 1. Now we replace X by corresponding character from codes table.

3.3 Some Problems

One of the faced problem in the new technique occurred in case of the original values of the bytes C, $C+3$, and $C-3$ seemed as if it have a specific character from the message then when we retrieve the message we will retrieve a wrong character. We solved this problem by check the bytes of the original image at the inserting step of the message into the image. In case of finding bytes in the image look like as there is a character, the technique is going to add binary 1's to left most significant bit onto one of the three bytes.

4. Implementation of the New Technique

4.1 Code Table

The code table is build up according to the idea of having a random numbers (from 0 to 221), the number 221 comes from that we have 35 characters, and each of them have 6 random codes according to the techniques above. After that each character will be assigned to six random numbers. Now let the sender and receiver share the password "1234QTR" and the code table, as shown in Table 1.

4.2 Message Hiding

In this stage the sender wants to send the message "MESSAGE". First he should decode the message into corresponding code for each character by choosing random character from it list of code words, let the random numbers be 5, 0, 4, 3, 2, 1, 3. So the message becomes 92, 174, 113, 111, 21, 167, 192[1] and the secret password coding is 142, 69, 170, 217, 117, 65, 80. The next step starting to search the image from the byte 142*10; i.e., from byte 1420 about the value 92 let us found it at byte 1500 we changed the value of the bytes 1497 and 1503 into 1420, or 2/3*1420 or 4/3*1420 these factors have been chosen depends on that we want the new values closer to the original value, the next step we start searching from the byte 690 about the value 174 and we found it at the byte 800, now we change the value of the bytes 797 and 803 into 174 and so on for the reaming characters of the message we are going to hide according to the next pseudo code .

1. *Start message hiding.*
2. *Code the message by using random (code word corresponding to each character of the message).*
3. *Code the password with the first code word from the list of code word corresponding to each character of the password.*
4. *For i=1 to length (message)*
   *For (j=i^{th} password code * 10 to size of the image)*
   If (byte (j) == code word of i^{th} character of the message)
   Then Stop the loop
   *If (absolute value (byte (j-3) – 3/4 * i^{th} character of the message) < byte (j-3) – i^{th} character of the message) and (absolute value (byte (j-3) – 3/4 * i^{th} character of the message) < byte (j-3) – 4/3* i^{th} character of the message)*
   Then byte (j-3) = 3/4 i^{th} character of the message.*
   *Else if (absolute value (byte (j-3) – 4 /3 * i^{th} character of the message) < byte (j-3) – i^{th} character of the message) and (absolute value (byte (j-3) – 4/3 * i^{th} character of the message) < byte (j-3) – 3/4* i^{th} character of the message)*
   Then byte (j-3) = 4/3 i^{th} character of the message.*
   Else byte (j-3) = i^{th} character of the message.
   Do steps b,c, and d for the byte(j+3).
5. *If the i equal the length (password)*
   Then i= i mod length (password).
6. *End.*

Table1. Codes table.

Character	List of Code words					
Q	117	128	63	66	171	3
W	168	180	156	10	91	190
E	174	82	212	192	12	209
R	80	115	169	11	130	103
T	65	137	143	58	61	183
Y	182	218	201	50	153	216
U	53	23	220	149	127	22
I	177	62	84	210	89	35
O	158	72	139	45	41	17
P	101	200	57	173	83	64
A	203	95	21	123	202	184
S	5	120	150	111	113	102
D	78	59	54	13	86	108
F	34	104	56	138	119	208
G	144	167	131	4	46	16
H	73	28	0	118	145	18
J	42	100	33	155	205	19
K	109	37	98	60	166	6
L	71	52	106	75	9	191
Z	140	90	25	204	77	32
X	48	209	76	121	188	160
C	154	39	93	94	112	151
V	195	81	117	49	129	194
B	105	165	135	172	179	44
N	211	14	176	26	38	124
M	47	88	199	164	178	92
Space	162	79	96	215	87	30
0	114	213	97	110	27	36
1	142	187	99	206	186	157
2	69	132	116	198	24	51
3	170	43	40	197	181	2
4	217	161	175	125	207	68
5	146	148	196	134	20	136
6	133	7	107	214	8	15
7	147	159	122	189	70	152
8	74	55	185	126	193	29
9	163	85	1	221	67	31

[1] The numbering of the list of code words starts from 0.

4.3 The Process of Message Retrieving

The receiver has the same code table of the sender and the same password as receive the stegano image.

Now the receiver starts coding the password to the first code word corresponding to each character form the password. The coded password is 92, 174, 113, 111, and 21,167,192

1. *Start message retrieving*
2. *Code the password with the first code word from the list of code word corresponding to each character of the password.*
3. *For i=1 to length(password)*
4. *For j=code word(j^{th} password character) to size(stegano image)*
   *If (byte(j-3)==byte(j) and byte(j+3)== byte(j)) or byte(j-3)*3/4==byte(j) and byte(j+3)== byte(j)) or byte(j-3)*4/3==byte(j) and byte(j+3)== byte(j)) or byte(j-3)==byte(j) and byte(j+3)== byte(j)) or byte(j-3)*3/4==byte(j) and byte(j+3)== byte(j)) or byte(j-3)*4/3==byte(j) and byte(j+3)*3/4== byte(j)) or byte(j-3)==byte(j) and byte(j+3)== byte(j)) or byte(j-3)*3/4==byte(j) and byte(j+3)== byte(j)) or byte(j-3)*4/3==byte(j) and byte(j+3)*4/3== byte(j))*
   Then the i^{th} character of message code word is byte(j), and stop loop.
5. *Decoding the codes word got from step 4 using codes table to retrieve the hiding message.*
6. *End.*

After implementation the new strategy for more than 150 messages, the size of messages between one character and a paragraph of 20 words approximately 100 characters, the relationship between the size of the message and the number of the segments is extrusive, also there will be an effect on the image by increasing the number of characters, but it will never affect the position of the characters.

The new strategy implementation tested by using the Images with two kind of extension (jpg and bmp) and we found that the effect of the new strategy was the same into the two extensions. Here we present some stegano images, the original image which is Figure 1 with the size 2.25 MB after the message hiding into the image the size does not affected.

The next image which is Figure 2 contains 700 and Figure 3 contains 1500 character. As seen there is no effect on the last two images that can be detected by human eye.

5. Conclusion and Future Work

In this research we explain a new steganography method our conclusion from the implementation of the new technique we found that it's flexible and have a high degree of complexity in retrieving the hidden message by unauthorized uses because of the relation between the password and the segments in the image, also the relation between the position of the message characters and the changes which have been done on the image, that make the strategy hard to being braked.

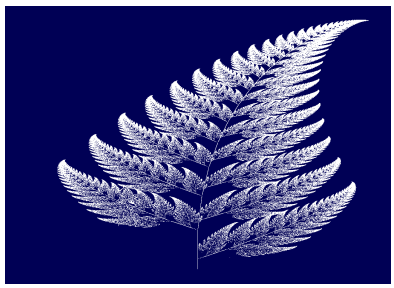

Figure 1. The original image.

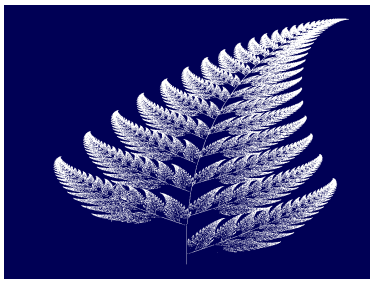

Figure 2. The stegano image with 700 characters.

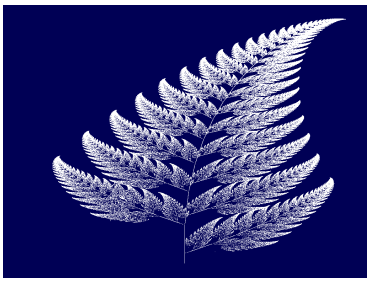

Figure 3. The stegano image with 1500 characters.

In the future work, we looking forward to find a better way of having the position of character by marking specific bytes at specific distance from the character we may try to replace this marking with better one since it is a weak point in our algorithm.

We may try to implant the pervious suggestion by using ranges which playing a big role to omit the problem of overlapping between characters. Also trying to add some complexity on the relation between the positions of the character code word and the changing which take palace in the image.

References

[1] Bender W., Gruhl D., and Morimoto N., "Techniques for Data Hiding," *IBM Systems Journal*, ISSN:0018-8670, vol. 35, no. 3-4, pp. 313-336, 1996.

[2] Chang C., Yeh J., and Hsiao J., "A Novel Scheme for Securing Image Steganography," *Lecture Notes in Computer Science*, Springer Berlin/ Heidelberg, vol. 2195, pp 804-811, 2001.

[3] Solaiman C., Lovanyi I., Cazuguel G., and Roux C., "An Optimization of Finite-State Vector Quantization for Image Compression," *Signal Processing: Image Communication*, vol. 15, pp. 545-558, 2000.
[4] Gersho A. and Gray R., *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, Boston, 1992.
[5] Johnson N. and Jojodia S., *Exploring Steganography: Seeing the Unseen*, Computing Practices. 1998.
[6] Maxemchuk N., "Electronic Document Distribution," *AT&T Technical Journal*, vol. 73, no. 5, pp. 73-80, 1994.
[7] Rao K. and Yip P., *Discrete Cosine Transform-Algorithms, Advantages, Applications*, Academic Press, 1990.
[8] Rosenbaum R., Schumann H., "A Steganographic Framework for Reference Colour Based Encoding and Cover Image Selection," *Lecture Notes in Computer Science*, Springer Berlin/ Heidelberg, vol. 1975, pp. 415-434, 2000.
[9] Sellars D., *An Introduction to Steganography*, http://www.cs.uct.ac.za/courses/CS400W/NIS/papers99/dsellars/stego.html, 1990.
[10] Simmons G., "The Prisoners' Problem and the Subliminal Channel," *in Proceedings of the Advances in Cryptology (CRYPTO'83)*, pp. 51-67, 1984.
[11] Smith J. and Dodge C., "Developments in Steganography," *Third International Information Hiding Workshop*, Dresden, Germany, 1999.

Fayik Alnaowk is an assistant professor and staff membership in the Department of Mathematics at Al Aqsa University, Gaza. 1993. Dean of Faculty of Science from September 2005 till now. He earned his PhD degree in cryptography from Ain Shams University in Egypt in 2001. He has supervised two PhD students in coding theory and cryptography. He also lead and teach modules at BSc in mathematics and computer science.

Basem Ahmed is a staff member in the Department of Computer Science in Al Aqsa University, Gaza. Holding a master degree in computer science from Al-Balqaa University, Jordan in 2005, got his BSc from Islamic University, Gaza in 2002, and worked as instructor at Islamic University, Gaza, Palestine since 2002. He also worked as systems analyst and design at National Nets Center, Jordan for 3 years. He has publication in the field of artificial intelligence.