TI Journals
International Journal of Engineering Sciences
www.waprogramming.com

ISSN
2306-6474

Using SFLA and LSB for Text Message Steganography in 24-Bit RGB Color Images

Maryam Habibi [1], Ronak Karimi [2], Masoud Nosrati *[3]

[1] *Kermanshah Branch, Islamic Azad University, Kermanshah, Iran.*

ARTICLE INFO

Keywords:

Steganography
shuffled leaping frog algorithm (SLFA)
data hiding
secure communication

ABSTRACT

In this study, we intend to increase the robustness of text message steganography by hiding the message in appropriate places of image.

Due to it, shuffled leaping frog algorithm (SLFA) is used to search among blocks of image data for similar ones in message. The SLFA is a memetic meta heuristic that is designed to seek a global optimal solution by performing a heuristic search.

Experimental results of applying the proposed method on 20 sample images shows the appropriateness of method. Also, some parameters like the length of data blocks are discussed which play an important role in efficiency of proposed algorithm.

1. Introduction

Secure communication is one of the most challenging topics in now-a-days digital world. It is hard to find a secure channel for communication all the times. So, some sciences and techniques came to existence in order to alternate the secure channel, such as cryptography [1], watermarking [2] and steganography.

Steganography is the art and science to hide data in a cover media such as text, audio, image, video, etc. [3] In other words, steganography is the process of hiding a secret message within a larger one in such a way that someone cannot know the presence or contents of the hidden message [4].

Steganography will hide the message so there is no knowledge of the existence of the message in the place [5]. The term Steganography is forked from the Greek words "stegos" meaning "cover" and "grafia" meaning "writing" defining it as "covered writing" [6]. The notion of data hiding or steganography was first introduced with the example of prisoners' secret message by Simmons in 1983 [7][8].

There are several media for covering the secret message. This paper will introduce a new method for hiding data in 24-bit RGB image files. Shuffled leaping frog algorithm (SLFA) got to work for securing stego-data against stego-analysis methods ("stego-analysis" also called "steganalysis" means detection of covered data in media [9][10]). The most important existing method for steganalysis is RS analysis [11], which investigates the statistical features of image to detect the message. Statistical analysis looks for the different blocks of pixels in image that aren't match with the context. In this way, it can understand about the existence of covered data.

It is clear that spirit of steganography is embedding message in a cover media. Regarding the process of embedding, applying steganography techniques for increasing the robustness is largely done in two phases:

1. Techniques before embedding
2. Techniques after embedding

First phase generally includes some activities like increase the statistical irregularity of image. For example, by adding impulse noise to carrier image before embedding, RS analysis results will fail to show the truth.

In second phase, it is tried to increase the robustness of secret messages against steganalysis [12]. It means, after embedding the data in special places of image, some techniques are hired to change the statistical features of pixels' blocks, so that RS get unable to detect the existence of message.

The method that is presented in this study focuses on the before embedding techniques. It aims finding suitable places in carrier image that causes less changes in original image. In this way, changes in color histogram is less and detecting the existence of stego-text will become harder. Finding the suitable places is a process which is implemented by shuffled leaping frog algorithm. System in whole, takes an RGB

* Corresponding author.
Email address: minibigs_m@yahoo.co.uk

24-bit color image and a secret message as input, and gives the modified image that contains the secret message in its least significant bits and a key string for extraction of message from modified image as output.

2. Backgrounds

2.1. Least Significant Bits

In proposed method, we will use LSB technique, because it is simple and it can help to convey the meaning of method fluently. So, this section will have a brief look at LSB.

An image can be represented by a collection of color pixels. The individual pixels are represented by their optical characteristics like "brightness", "chroma" etc. Each of these characteristics can be digitally expressed in terms of 1s and 0s [26]. There are different color spaces that present different forms for storing images. A color space is a method by which it is possible to specify, create and visualize color [27]. The most common color space among all is RGB (Red, Green, and Blue). Each pixel in a 24-bit bitmap image in this space is described by 3 sets of 8 bits (3 bytes), that each set contains the intensity value of individual red, green and blue. Combination of these values forms the characteristics of the pixel. Figure 1 illustrates this matter.

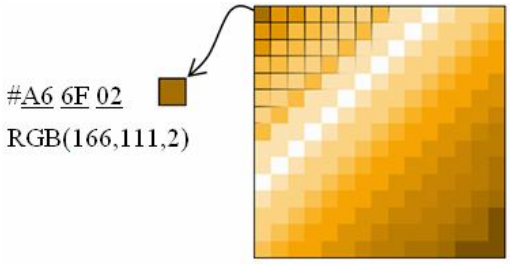


Figure 1. A pixel in RGB color space

Least Significant Bits (LSB) insertion is a simple approach for embedding information in image file. The simplest steganography techniques embed the bits of the message directly into least significant bit plane of the cover image in a deterministic sequence. Modulating the least significant bit does not result in human-perceptible difference because the amplitude of the change is small [28].

To hide a secret message inside an image, a proper cover image is needed. Because this method uses bits of each pixel in the image, it is necessary to use a lossless compression format, otherwise the hidden information will get lost in the transformations of a lossy compression algorithm. When using a 24-bit color image, a bit of each of the red, green and blue color components can be used, so a total of 3 bits can be stored in each pixel. For example, the following grid can be considered as 3 pixels of a 24-bit color image, using 9 bytes of memory:

(00100111 11101001 11001000)
(00100111 11001000 11101001)
(11001000 00100111 11101001)

When the character A, which binary value equals 10000001, is inserted, the following grid results:

(00100111 11101000 11001000)
(00100110 11001000 11101000)
(11001000 00100111 11101001)

In this case, only three bits needed to be changed to insert the character successfully. On average, only half of the bits in an image will need to be modified to hide a secret message using the maximal cover size. The result changes that are made to the least significant bits are too small to be recognized by the human visual system (HVS), so the message is effectively hidden [29].

As you see, the least significant bit of third color is remained without any changes. It can be used in combination of LSBs of other pixels or individually for checking the correctness of 8 bits which are embedded in these 3 pixels. In other words, it could be used as "parity bit" [4].

2.2. Shuffled leaping frog algorithm

The shuffled frog-leaping algorithm is a memetic metaheuristic that is designed to seek a global optimal solution by performing a heuristic search. It is based on the evolution of memes carried by individuals and a global exchange of information among the population [30]. In essence, it combines the benefits of the local search tool of the particle swarm optimization [31], and the idea of mixing information from

parallel local searches to move toward a global solution [32]. The SFL algorithm has been tested on several combinatorial problems and found to be efficient in finding global solutions [30].

The SFL algorithm involves a population of possible solutions defined by a set of frogs (i.e. solutions) that is partitioned into subsets referred to as memeplexes. The different memeplexes are considered as different cultures of frogs, each performing a local search. Within each memeplex, the individual frogs hold ideas, that can be influenced by the ideas of other frogs, and evolve through a process of memetic evolution. After a number of memetic evolution steps, ideas are passed among memeplexes in a shuffling process [33]. The local search and the shuffling processes continue until convergence criteria are satisfied [30].

First, an initial population of '*P*' frogs is created randomly. For *S*-dimensional problems, each frog *i* is represented by *S* variables as $X_i=(x_{i1}, x_{i2}, \ldots\ldots, x_{iS})$. The frogs are sorted in a descending order according to their fitness. Then, the entire population is divided into *m* memeplexes, each containing *n* frogs (i.e. $P=m*n$). In this process, the first frog goes to the first memeplex, the second frog goes to the second memeplex, frog *m* goes to the m^{th} memeplex, and frog *m+1* goes to the first memeplex, and so on (figure 2).

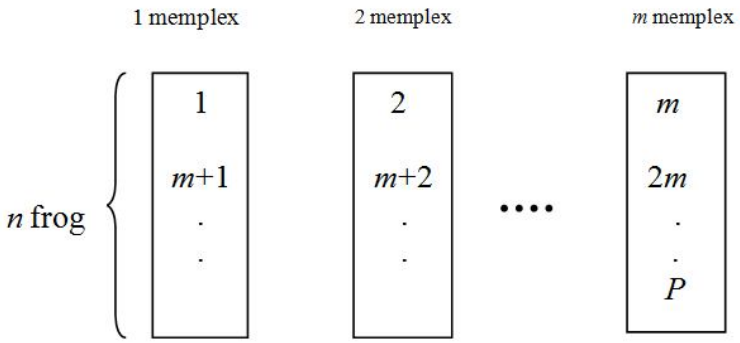


Figure 2. Division of *P* frogs in *m* memeplexes

The SFL algorithm is described by the pseudocode in Appendix 1 and the corresponding flowchart in figure 3.

Within each memeplex, the frogs with the best and the worst fitness are identified as X_b and X_w, respectively. Also, the frog with the global best fitness is identified as X_g. Then, an evolution process is applied to improve only the frog with the worst fitness (i.e. not all frogs) in each cycle. Accordingly, the position of the frog with the worst fitness is adjusted as follows:

$$D_i = random() * (X_b - X_w) \qquad \textit{(Change in frog position)}$$

$$X_w = CurrentPosition(X_w) + D_i \;\; ; \; D_{Max} \geq D_i \geq -D_{Max} \qquad \textit{(New position)}$$

where *random()* is a random number between *0* and *1*; and D_{max} is the maximum allowed change in a frog's position. If this process produces a better frog (solution), it replaces the worst frog. Otherwise, the calculations in equations *(Change in frog position)* and *(New position)* are repeated with respect to the global best frog (i.e. X_g *replaces* X_b). If no improvement becomes possible in this latter case, then a new solution is randomly generated to replace the worst frog with another frog having any arbitrary fitness (as shown in figure 3). The calculations then continue for a specific number of evolutionary iterations within each memeplex [30]. The main parameters of the SFL algorithm are: number of frogs *P*, number of memeplexes, and number of evolutionary iterations for each memeplex before shuffling [34].

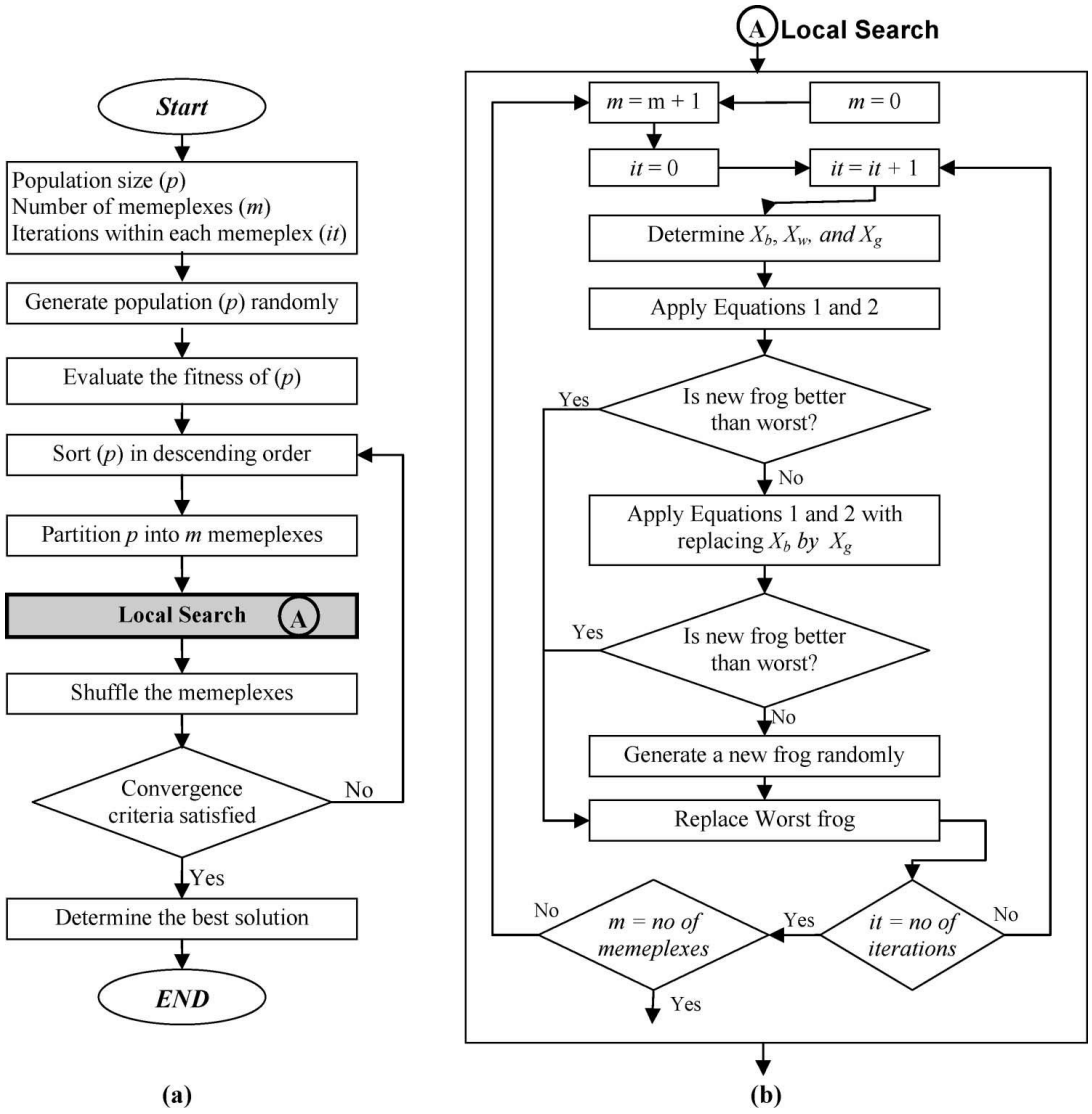


Figure 3. Flowchart of the shuffled frog-leaping algorithm.

3. Proposed method

3.1. Main idea

Main idea of the proposed method is finding some blocks of data in carrier image that is very similar to blocks of message. It is clear that the first step is blocking the data of image. As it was mentioned in section 2.1, usable data of image for hiding message are LSBs. So, for the first step, an array of original message LSBs should be extracted which is a set of 0s and 1s. Also, message should be turned to 0-1 format. Now, 2 sets of data will be available like figure 4.

Image LSBs:

1	0	0	1	0	1	0	1	1	1	0	0	0	0	1	1	1	0	1	0	1	…

Message bits:

1	1	0	0	1	1	0	1	1	0	1	1	1	0	1	0	…

Figure 4. Bits of image and message

It is clear that there might be similar patterns of message bits in image LSBs. For the next step, divide the LSBs and message bits to some blocks with similar length. Finding the most similar blocks of message in LSBs is the next step. Embedding the data with most similarity is the final goal. Figure 5 illustrates the matter.

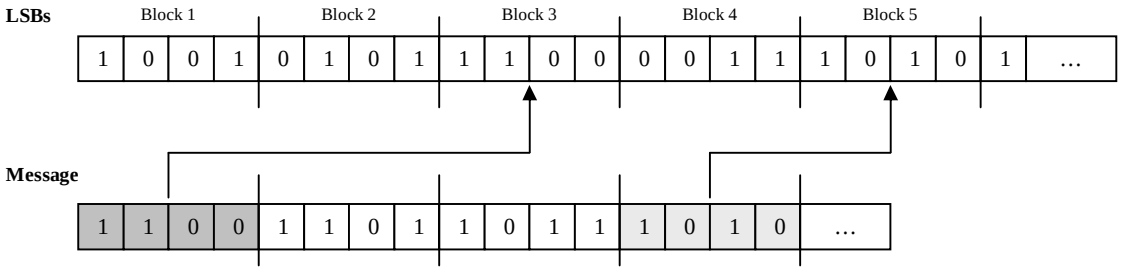


Figure 5. Blocking and finding the similarities. ($L_{block}=4$)

Final step is embedding and creating stego-key. After finding the similarities, blocks of message will be embedded to image LSBs. Address of embedded data blocks is saved in a stego-key file.

3.2. Proposed algorithm

According to up mentioned steps, embedding algorithm is listed in follow:

1. Start.
2. Extract the image LSBs and change the text message to bits.
3. Create 2 block of data by LSBs and message bits.
4. Find the most similar blocks of message in LSBs by SLFA algorithm.
5. Embed the message blocks in image LSBs and save the address of blocks in stego-key file.
6. end.

// Find the most similar blocks of message in LSBs by SLFA algorithm

4.1. Start.
4.2. Set:
Population size *P=(ImageHeight * ImageWidth)*
Number of memeplexes *m=(number of message blocks)*
Iteration within each memeplex *it=(optional)*
4.3. Generate population *P* randomly.
4.4. Evaluate the similarity of *P* with message blocks.
4.5. Sort the *P* in descending order.
4.6. Partition *P* into *m* memeplexes.
4.7. Do the local search (fitness criteria is the most similarity to message blocks).
4.8. Shuffle the memeplexes in *P*.
4.9. If the P satisfies the convergence criteria, then determine the best solution.
else, go to 4.5.
4.10. end.

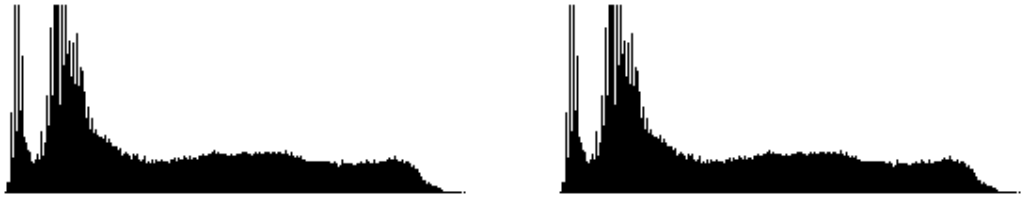
4. Experimental results

For testing the proposed method, it was implemented in Matlab 2007. We set a 24-bitmap color BMP as carrier image and the first page of current paper as secret message for the input of algorithm. Image original size was 1000*880. We chose $L_{block}=8$ for example. Look at figure 6.

A. Original image B. Image after embedding secret message

Figure 6. Embedding the secret message into a sample image

As it is seen, there is no significant change in the shape and integrity of image. It is interesting to check the histogram metrics. Figure 7 shows the histogram of sample image before and after embedding. Results show that there is no change in the histogram of both images. It means, our proposed method was successful for data embedding.



A. Histogram of original image B. Histogram of image after embedding secret message

Figure 7. Embedding the secret message into a sample image

There is a close relationship between the length of blocks and the efficiency of this method. *Eff* is our selected criteria for measuring the efficiency of method based on changes in modified image rather than original image. *Eff=1* means there is no change in original image after embedding the message and *Eff=0* shows that original image completely changed.

For calculation of *Eff*, let an input original image with *Height * Width* pixels.

Structure of value of each pixel is *(0-255,0-255,0-255)*. So, maximum change of each pixel is *255+255+255=765*.
Accordingly, domain of changes in values of all image pixels is:

$$nP = (Height * Width) \qquad (number\ of\ pixels)$$

$$DoC = (nP * 765) \qquad (domain\ of\ changes)$$

Eff is calculated as:

$$Eff = \frac{\sum_{Pixel=1}^{nP} |ModifiedValue - OriginalValue|}{DoC}$$

Following table shows the results of application of proposed method on 40 sample pictures. Sample pictures are selected from 4 groups of sizes. Results show that the size of image strongly effects the *Eff*.

Table 1. Application of proposed method on 20 sample pictures.

Image size	Length of message	Number of blocks	*Eff*
800*600	3940 Bytes	394	0.968325
800*600	3940 Bytes	394	0.957451
800*600	3940 Bytes	394	0.995125
800*600	3940 Bytes	394	0.975367
800*600	3940 Bytes	394	0.964648
800*600	3940 Bytes	394	0.939984
800*600	3940 Bytes	394	0.968125
800*600	3940 Bytes	394	0.959579
800*600	3940 Bytes	394	0.999972
800*600	3940 Bytes	394	0.970127
1200*900	3940 Bytes	394	0.999596
1200*900	3940 Bytes	394	1
1200*900	3940 Bytes	394	1
1200*900	3940 Bytes	394	0.991987
1200*900	3940 Bytes	394	0.989968
1200*900	3940 Bytes	394	1
1200*900	3940 Bytes	394	1
1200*900	3940 Bytes	394	1
1200*900	3940 Bytes	394	0.998996
1200*900	3940 Bytes	394	0.999979

Length of blocks (which depends on number of blocks) is an important factor. Very large amount of blocks length (L_{block} = *(very big)*) causes the less efficiency. Also, very small amount of L_{block} causes the growth of number of frogs and it will increase the execution complexity of program. Figure 4 shows the total relationship of efficiency of algorithm (in the term of runtime complexity) and L_{block} . It is important to know that proper L_{block} for different images will be different.

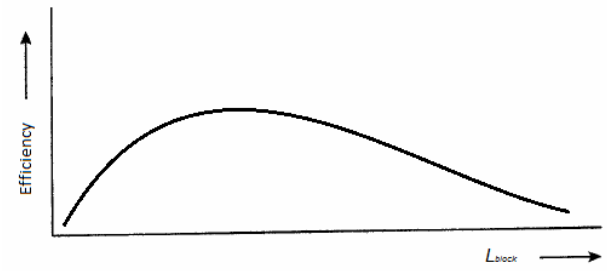


Figure 8. Relationship of efficiency of method (in the term of complexity) and L_{block}

4. Conclusion

In this paper, a heuristic SLFA-based approach for steganography in image carrier was introduced. First, we told that there are two different types of techniques for hiding the steganography: techniques before embedding, techniques after embedding. Our proposed method focused on the first type by trying to find appropriate places in carrier image to embed the message with the least changes of bits. To achieve this goal, some steps were specified, including: extracting the image LSBs and changing the text message to bits; creating 2 block of data by LSBs and message bits; finding the most similar blocks of message in LSBs by SLFA algorithm; embedding the message blocks in image LSBs and saving the address of blocks in stego-key file.

Experimental results of applying the proposed method on 20 sample images shows the efficiency of method. Least changes in sample images confirms the method.

References

[1] Ahmed Al-Vahed, Haddad Sahhavi. An overview of modern cryptography, World Applied Programming, Vol (1), No (1), April 2011.
[2] Matt L. Miller, Ingemar J. Cox, Jean-Paul M.G. Linnartz, Ton Kalker. Published in "Digital Signal Processing in Multimedia Systems, Ed. K. K. Parhi and T. Nishitani, Marcell Dekker Inc., 461-485, (1999)
[3] Inoue H, Miyazaki A, Katsura T (2002). An image watermarking method based on the wavelet transform. In: International Conference on Image Processing, IEEE ICIP, Kobe, vol 1, pp 296–300.

[4] Masoud Nosrati, Ronak Karimi, Mehdi Hariri. Embedding Stego-Text in Cover Images Using linked List Concepts and LSB Technique. World Applied Programming, Vol (1), No (4), October 2011. 264-268.
[5] Masoud Nosrati, Ronak Karimi, Mehdi Hariri. An introduction to steganography methods. World Applied Programming, Vol (1), No (3), August 2011. 191-195.
[6] S. Katzenbeisser, F.A.P. Petitcolas, Information Hiding Techniques for Steganography and Digital Watermarking, Artech House, Norwood, MA, 2000.
[7] G. J. Simmons, "The prisoners' problem and the subliminal channel" in Proc. Advances in Cryptology (CRYPTO '83), pp. 51-67.Berglund, J.F. and K.H. Hofmann, 1967. Compact semitopological semigroups and weakly almost periodic functions. Lecture Notes in Mathematics, No. 42, Springer-Verlag, Berlin-New York.
[8] Masoud Nosrati, Ronak Karimi, Mehdi Hariri. Audio Steganography: A Survey on Recent Approaches. World Applied Programming, Vol (2), No (3), March 2012. 202-205
[9] Niels Provos, Peter Honeyman. Hide and Seek: An Introduction to Steganography. IEEE SECURITY & PRIVACY, MAY/JUNE 2003.
[10] Abbas Cheddad, Joan Condell, Kevin Curran, Paul Mc Kevitt, Digital image steganography: Survey and analysis of current methods Signal Processing 90 (2010) 727–752
[11] R. Popa, An Analysis of Steganographic Techniques, The "Politehnica" University of Timisoara, Faculty of Automatics and Computers, Department of Computer Science and Software Engineering, http://ad.informatik.uni-freiburg.de/mitarbeiter/will/dlib_bookmarks/digital-watermarking/popa/popa.pdf, 1998
[12] ElShafie DR, Kharma N, Ward R (2008) Parameter optimization of an embedded watermark using a genetic algorithm. In: International symposium on communications, control and signal processing, ISCCSP, St Julians12–14 March, pp 1263–1267
[13] Provos N (2001) Defending against statistical steganalysis. In: SSYM'01 Proceedings of the 10th conference on USENIX Security Symposium, USENIX Association Berkeley, CA, USA, vol 10, pp 323–335.
[14] Wu N, Hwang M (2007) Data hiding: current status and key issues. Int J Network Security 4(1):1–9.
[15] Chen W (2003) A comparative study of information hiding schemes using amplitude, frequency and phase embedding, PhD thesis, National Cheng Kung University, Taiwan.
[16] Elham Ghasemi, Jamshid Shanbehzadeh, and Nima Fassihi. High Capacity Image Steganography Based on Genetic Algorithm and Wavelet Transform, S.I. Ao et al. (eds.), Intelligent Control and Innovative Computing, Lecture Notes in Electrical Engineering 110, DOI 10.1007/978- 1-4614-1695-1 30.
[17] Ran-Zan Wang, Chi- Fang Lib, and Ja- Chen Lin, "Image hiding by optimal LSB substitution and Genetic algorithm", 2001 Pattern Recognition Society. Published by Elsevier Science Ltd.
[18] J. K. Mandal, A. Khamrui. A Data Embedding Technique for Gray scale Image Using Genetic Algorithm (DEGGA). International Confrence on Electronic Systems (ICES-2011).
[19] Shen Wang, Bian Yang and Xiamu Niu "A Secure Steganography Method based on Genetic Algorithm" Journal of Information Hiding and Multimedia Signal Processing c 2010 ISSN 2073-4212. Volume 1, Number 1, January 2010
[20] Vijay Kumar Sharma, Vishal Shrivastava. Improving the Performance of Least Significant Bit Substitution Steganography against Rs Steganalysis by Minimizing Detection Probability. International Journal of Information and Communication Technology Research, Volume 1 No. 4, August 2011.
[21] Z. Liu and L. Xi, "Image information hiding encryption using chaotic sequence," in Proceedings of the 11th International Conference on Knowledge-Based Intelligent Information and Engineering Systems and the XVII Italian Workshop on Neural Networks, pp. 202–208, 2007.
[22] Y. Zhang, F. Zuo, Z. Zhai, and C. Xiaobin, "A new image encryption algorithm based on multiple chaos system," in Proceedings of the International Symposium on Electronic Commerce and Security (ISECS '08), pp. 347–350, August 2008.
[23] R. Munir, B. Riyanto, S. Sutikno, and W. P. Agung, "Secure spread spectrum watermarking algorithm based on chaotic map for still images," in Proceedings of the International Conference on Electrical Engineering and Informatics, 2007.
[24] Z. Dawei, C. Guanrong, and L.Wenbo, "A chaos-based robust wavelet-domain watermarking algorithm," Chaos, Solitons and Fractals, vol. 22, no. 1, pp. 47–54, 2004.
[25] Lifang Yu, Yao Zhao, Rongrong Ni, Ting Li. Hindawi Publishing Corporation, EURASIP Journal on Advances in Signal Processing, Volume 2010, Article ID 876946. doi:10.1155/2010/876946.
[26] Soumyendu Das, Subhendu Das, Bijoy Bandyopadhyay and Sugata Sanyal, "Steganography and Steganalysis: Different Approaches", International Journal of Computers, Information Technology and Engineering (IJCITAE), Vol. 2, No 1, June, 2008, Serial Publications.
[27] Adrian Ford, Alan Roberts, Colour Space Conversions, 1998, Available at: http://www.poynton.com/PDFs/coloureq.pdf
[28] Mohamed Amin, Muhalim and Ibrahim, Subariah and Salleh, Mazleena and Katmin, Mohd Rozi (2003) Information hiding using steganography. Project Report. Available at: http://eprints.utm.my/4339/1/71847.pdf
[29] Robert Krenn, Steganography and steganalysis, Internet Publication, March 2004. Available at: http://www.krenn.nl/univ/cry/steg/article.pdf
[30] Eusuff, M.M. and Lansey, K.E., Optimization of water distribution network design using the shuffled frog leaping algorithm. J. Water Resour. Planning Mgmt, 2003, 129, 210 – 225.
[31] Kennedy, J. and Eberhart, R., Particle swarm optimization, in Proceedings IEEE International Conference on Neural Networks, IEEE Service Center, Piscataway, NJ, pp. 1942 – 1948, 1995.
[32] Duan, Q.Y., Gupta, V.K. and Sorooshian, S., Shuffled complex evolution approach for effective and efficient global minimization. J. Optimization Theory Appns, 1993, 76, 502 – 521.
[33] Liong, S.-Y. and Atiquzzaman, Md., Optimal design of water distribution network using shuffled complex evolution. J. Instn. Engrs, 2004, 44, 93 – 107.
[34] Emad Elbeltagiy, Tarek Hegazyz and Donald Griersonz. A modified shuffled frog-leaping optimization algorithm: applications to project managementStructure and Infrastructure Engineering, Vol. 3, No. 1, March 2007, 53 – 60