A Proposed Technique For Hiding Data Into Video Files

Mohamed Elbayoumy[1], Mohammed Elmogy[2], Ahmed Abouelfetouh[3] and Rasha Elhadary[4]

[1] **Information systems department, Faculty of computer science and information systems, Mansoura University, Mansoura, Egypt**

[2] **Information technology department, Faculty of computer science and information systems, Mansoura University, Mansoura, Egypt**

[3] **Information systems department, Faculty of computer science and information systems, Mansoura University, Mansoura, Egypt**

[4] **Information systems department, Faculty of computer science and information systems, Mansoura University, Mansoura, Egypt**

Abstract

The quick development of data transmission through the internet made it easier to send and receive the data accurately and in a faster way between the source and the destination. One of the most significant factors of the information technology and data communication is the security of the information. For security objectives the concept of steganography is being used. The importance of steganography is that it avoids drawing suspicion to the existence of a hidden message. Over the past few years, numerous steganography techniques that embed hidden messages in multimedia objects have been proposed. These techniques hide information or messages in images in such a manner that the alterations made to the image are perceptually invisible. In this paper, we proposed an image based steganography technique that combines cryptography and steganography and depends on modifying the pixel values slightly to contain the hidden data. Our main objectives are to enhance the security of the communication, maximize the embedding capacity and achieve a high degree of flexibility and invisibility.

Keywords: *Least Significant Bit (LSB), Cover image, Stego-image, Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR).*

1. Introduction

Cryptography and steganography are very common widely used techniques that manipulate information (messages) in order to encrypt and hide their existence, respectively. Steganography is the art/ science of hiding the existence of the communication between the sender and the receiver. The word steganography comes from the Greek words Steganós (Covered) and Graptos (Writing) and literally means "hidden writing" [1]. People have used steganography through the centuries to hide the transmission of messages. Breaking of steganography is known as steganalysis. Steganalysis [2] is the discovery of the existence of hidden information; therefore, like cryptography and cryptanalysis, the goal of steganalysis is to discover hidden information and to break the security of its carriers. Cryptography encodes a message into another format so it cannot be understood. Breaking of cryptography is known as cryptanalysis. The difference between steganography and cryptography is that the cryptography focuses on keeping the contents of a message secret whereas steganography focuses on keeping the existence of a message secret. Steganography, when combined with cryptography, is a powerful tool which enables people to communicate without possible eavesdroppers even knowing there is a form of communication between two entities. Seth et al. [3] have proposed a technique using the combination of encryption and steganography to enhance the security of the data to be sent. The whole process is carried in three steps which are encryption, steganography and decryption.

Hiding information into a medium requires following elements [4]: a) the cover medium (C) that will hold the secret message. b) The secret message (M) that may be plain text, digital image file or any type of data. c) The stegonographic algorithm and d) the stego-key (K) may be used to hide and extract the message. Figure 1 illustrates the overall process of steganography. Steganography can be divided into five types: Text Steganography, Image Steganography, Audio Steganography, Video Steganography and Protocol Steganography [5]. An image can be described as a numeric representation that forms a grid and the individual points are referred to as pixels. Grayscale images use 8 bits for each pixel and are able to display 256 different colors or shades of grey. Digital color images are typically stored in 24-bit files and use the RGB color model, also known as true color [6].

There are two main steganographic fields [7]: The first is Spatial Domain Techniques which rely on directly changing some bits in the image pixel values to hiding data. Least significant bit (LSB) based steganography [8] is one of the simplest techniques that hides a secret message in the LSBs of pixel values without perceptible distortions. The other field is the Transform Domain

Technique in which the message is inserted into transformed coefficients of image giving more information hiding capacity and more robustness against attacks.

The main characteristics of the data hiding techniques can be summarized into four points: a) Perceptibility: embedding message does not distort cover medium to a visually unacceptable level. b) Capacity: the amount of information can be hidden with relative to the change in perceptibility. c) Robustness to attacks: can embedded data be destroyed or changed according to some image processing or manipulation. d) Tamper Resistance refers to the difficulty for an attacker to alter a message once it has been embedded in a stego-image [9].

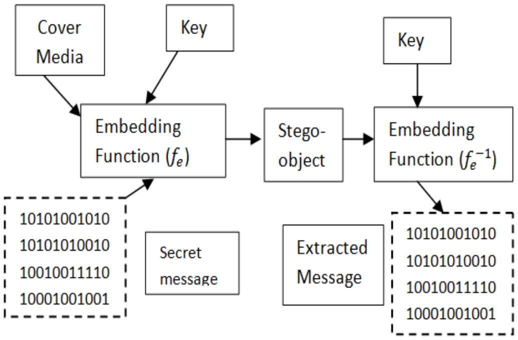


Fig. 1 The block diagram of data hiding.

This paper is organized into five sections as follows: In section 2, the related work on the field of image steganography is presented. Section 3 describes in details the framework and the components of the developed system. Section 4 provides the experimental results of the system and the statistical evaluation of these results. Section 5 concludes the paper and address the future scope of our work.

2. Related Work

Steganography is gaining attraction by people due to the security issues over internet. Steganography has evolved into a digital strategy of hiding a file in some form of multimedia, such as an image, an audio file or even a video file.

There are many researchers working in this field and have proposed various techniques to hide data. For example, Neeta et al. [8] proposed the Least Significant Bit (LSB) modification technique suggesting that data can be hidden in the least significant bits of the cover image so that the human eye can not notice the hidden image in the cover file. In LSB steganography, the least significant bits of the cover media's digital data are used to hide the message. LSB replacement steganography changes the last bit of each of the data values to the next bit of the message to be hidden. The difference between the original image and the stego image (containing the message) will be hardly noticeable to the human eye. The advantages of LSB techniques are: Popularity, Easy to understand and comprehend, High perceptual transparency, Low degradation in the image quality. However, there are few weaknesses of using LSB. It is very sensitive to any kind of filtering or manipulation of the stego-image .Scaling, rotation, cropping, addition of noise, or lossy compression to the stego-image will destroy the message. On the other hand, for the hiding capacity, the size of information to be hidden relatively depends to the size of the cover-image.

Chan et al. [10] have developed the optimal pixel adjustment procedure (OPAP). OPAP reduces the distortion caused by the LSB substitution method. In OPAP method the pixel value is adjusted after the hiding of the secret data is done to improve the quality of the stego image without disturbing the data hidden. The results obtained showed significant improvement than the method by genetic algorithm and optimal LSB substitution.

Fridrich et al. [11] proposed another approach for embedding in spatial domain. In their method, noise that statistically resembles common processing distortion, e.g., scanner noise, or digital camera noise, is introduced to pixels on a random walk. The noise is produced by a pseudo random noise generator using a shared key. A parity function is designed to embed and detect the message signal modulated by the generated noise.

Wang et al. [12] have proposed a method that hides the data in the target pixel by finding the characteristics of four pixels surrounding it. This method depends on Pixel Value Differencing that is used to provide a high quality stego image in spite of the high capacity of the concealed information. The number of insertion bits is dependent on whether the pixel is an edge area or smooth area. In edge area the difference between the adjacent pixels is more, whereas in smooth area it is less. While human perception is less sensitive to simple changes in edge areas of a pixel, it is more sensitive to changes in the smooth areas.

Masking and filtering technique [6], usually restricted to 24 bits and gray scale images, hide information by marking an image, in a manner similar to paper watermarking. The technique performs analysis of the image, thus embed the information in significant areas so that the hidden message is more integral to the cover image than just hiding it in the noise level.

DCT coefficients transform a signal or image from the spatial domain to the frequency domain. It can separate the image into high, middle and low frequency components. Much of the signal energy lies at low frequency which contains most important visual parts of the image while in high frequency sub-band, high frequency components of the image are usually removed through compression and noise attacks. So the secret message is embedded by modifying the coefficients of the middle frequency sub-band, so that the visibility of the image will not be affected [13].

Yang et al. [14] proposed a simple reversible data hiding scheme based on Integer Wavelet Transform. This model shows that both the host media and secret message can be completely recovered, without distortion, if the stego images remain intact. In addition, a smart adjustment of IWT coefficients employed in the proposed method can effectively embed data bits into the IWT block while keeping distortion low. Based on IWT domain, the stego-images generated by the proposed method are equipped with a certain degree of robustness to protect against image processing operations.

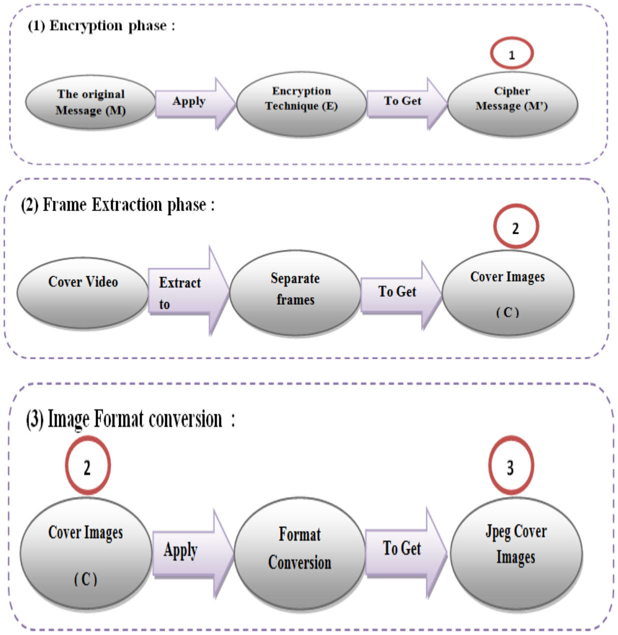
Unlike spatial and transform domain techniques discussed above, model based techniques try to model statistical properties of an image, and preserve them in the embedding process. Another model based technique was proposed by Radhakrishnan et al. [15] in which the message signal is processed so that it would exhibit the properties of an arbitrary cover signal, they call this approach data masking. As argued if Alice wants to send an encrypted message to Bob, the warden Wendy would be able to detect such a message as an encrypted stream since it would exhibit properties of randomness. In order for a secure channel to achieve covertness, it is necessary to preprocess the encrypted stream at the end points to remove randomness such that the resulting stream defeats statistical tests for randomness and the stream is reversible at the other end.

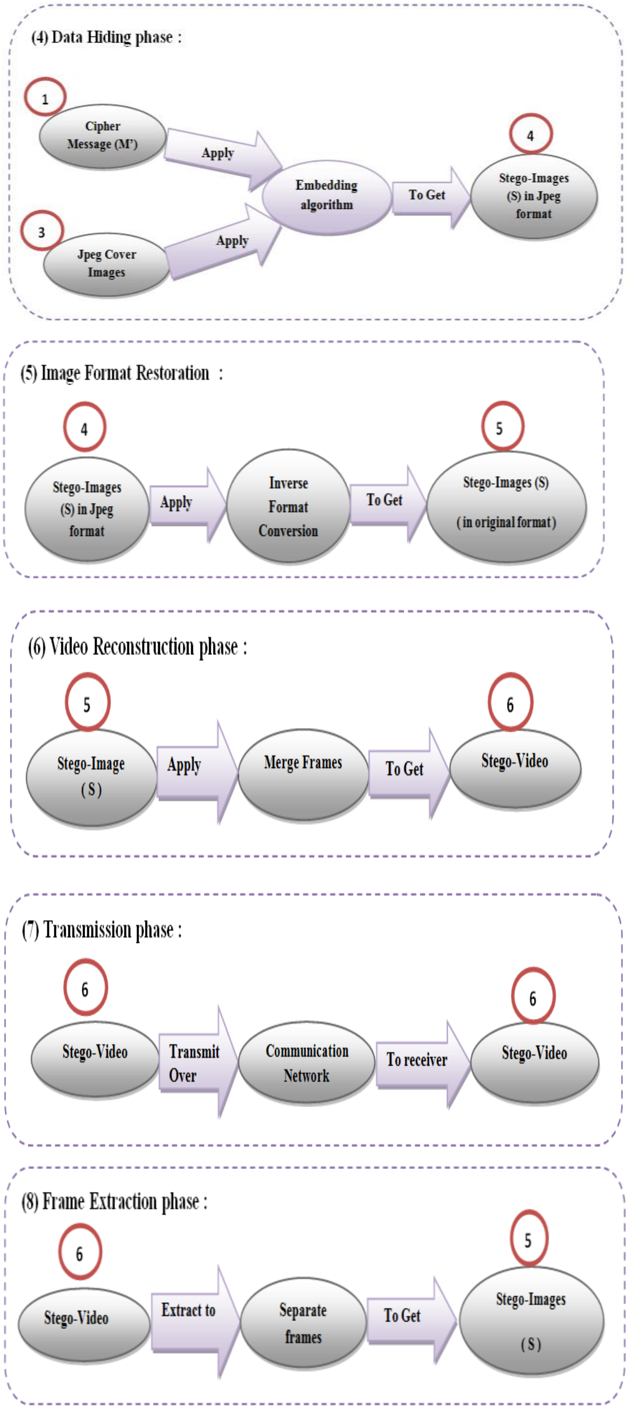
In this paper, we propose a technique that is carried out with the objective of hiding the message or a secret data into an image which acts as a cover using a new technique. The primary motivation of the current work is to increase the data embedding capacity and the PSNR of the stego image (peak signal to noise ratio). After the previous overview of the current steganographic techniques, we have to say that our main contribution in this system is to give a new direction on how to improve existing methods of hiding secret messages, by **combining** steganography and cryptography providing a more layer of security. That is, a message is encrypted before being hidden in a message in order to achieve a better level of secrecy (which provides a basic example on how to combine cryptography and steganography). The main **advantages** regarded in our system are: greater embedding capacity, more security, more flexibility and more invisibility.

3. The Proposed Framework

The proposed system is a data-hiding technique that uses high resolution digital video file as a cover object. The intended recipient only needs to process the required steps in order to reveal the message; otherwise the existence of the hidden information is virtually undetectable. The proposed technique provides the ability to hide a significant quality of information making it different from typical data hiding mechanisms because here we considered application that require significantly larger payloads like picture-in-video. The proposed technique is composed of two main phases: the first phase is the encryption phase, in which the secret message is converted into another format (usually a binary data). Our main effort will clearly appear in the second phase, Data hiding phase.

The overall process is composed of ten steps: First step, the encryption phase in which the secret message is converted into another format (usually a binary data) and here a previously known encryption algorithm will be used. The second step is to extract frames out of the cover video. Third step is a pre-processing step by converting the extracted frames into a common standard universal format (Jpeg) if they are not already in this format. This step is inverted after applying the embedding algorithm in the next step to restore the image in its original format. Next step is to apply the embedding algorithm (described in details in next section) to hide the message into the extracted frames. Sixth step is to reconstruct the video file after the data is embedded into its frames. Seventh step is to transmit the Stego-video over the network to the intended receiver. At the receiver end, the eighth step is that video is separated again to extract the frames holding the secret message, and then the inverse of phases 2 and 1 is applied respectively to get the original message in its main format. The overall process of the proposed data hiding technique is divided into the consecutive steps shown in figure 2.





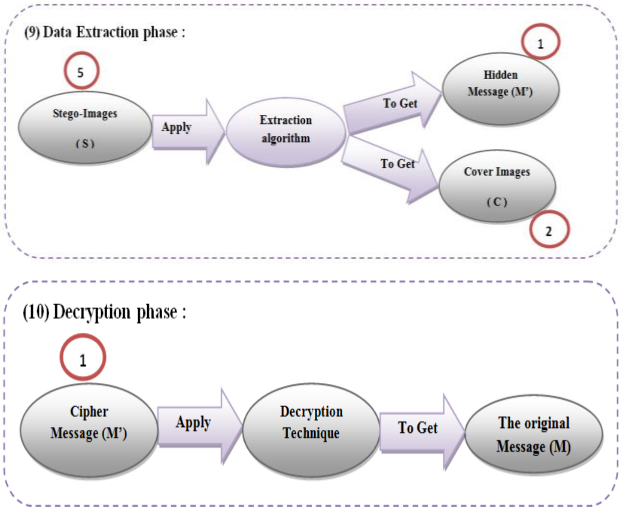


Fig. 2 The proposed Data hiding steps.

3.1 Data Hiding and Extraction Algorithms

Our objective is to propose a new technique to hide a secret text message (M) in a cover image (C) using the steganographic technique described below:

First, we decompose the original image into blocks of size (3 by 3) pixels. If the width or height of the image is not a multiple of 3, add zero padding pixels. After the decomposition we can work on every single block independently. Second, we will convert the image pixel values to its binary equivalents, and convert the secret message characters to binary values using Unicode mapping. Third step is to manipulate the values of pixels in every block (this step is repeated for all image blocks of pixels). Determine the minimum pixel value in the block (name it P_{min}), keep it as it is, for other pixels in the block modify its value to be the difference between its value and the minimum one (name it D= P-P_{min}). Note that adjacent pixels of an image always have very close values, so the differences stored will be small values. Then determine the maximum value of these differences in the block (name it Max (D)). After the third step is applied for all blocks here comes the fourth step which is determining the greatest value of all maximum differences among all blocks (greatest Max (D) in whole image). Then determine the number (name it N) of bits just enough to store this greatest Max (D) value. Fifth step is a frequent swap process, for each pixel (other than the block minimum pixel) we will bring the N rightmost bits (that store difference) to N leftmost places, and bring the remaining zeros to the right places. Sixth step is to modify the values of pixels to embed the secret message bits. Perform a bit-by-bit insertion as follows: fill the (8 – N) empty right bits of every non-minimum pixel with an equivalent number of bits from the message sequence until the message is finished. At last, we will merge all the pixel blocks after modifying 8 out of 9 pixels in each block, so we get a new modified values image with the secret message embedded. Note that we have a great storage capacity as we use bits in most of the 9 pixels of each block.

Here is the pseudo-code declaring these steps:

```
Input : Cover image (I)  , Secret message (M)
Output : Text embedded image (I')
Step 1: Divide image I of size (M × N) into 3X3 pixel blocks, If $M\%3 \neq 0$ or $N\%3 \neq 0$, Pad with additional zeros.
Step 2: Convert every letter of message M to its equivalent Unicode series of bits so the message
        becomes $M_b$.
Step 3: Convert every pixel value of image I to its binary value ($P_b$).
Step 4: For each block B in image I :
        Determine the minimum pixel value ($P_{bmin}$) in B
        For pixel $P_{bmin}$ : Set $P_b = P_{bmin}$
        For pixels other than $P_{bmin}$: Set $P_b = P_b - P_{bmin}$ (the difference D)
        Determine the maximum D in block B ($D_{max}$)
Step 5: Find the maximum difference of all image blocks Max($D_{max}$) , then determine N = the
        number of bits enough to Save Max($D_{max}$)
Step 6: For each pixel in block B other than $P_{min}$ :
        Swap the position of right-most N bits with the left-most bits.
        If Mb size ≠ 0: Set the new empty right-most bits to an equivalent number of bits in message $M_b$.
Step 7: Merge all the modified-pixels blocks into the new Image I'.
```

Algorithm 1. Data Hiding Algorithm

For The inverse process, the data extraction, here is the pseudo code explaining it:

```
Input:    Text embedded (Stego) Image (I')
Output: Cover image (I), Hidden message (M)
Step 1: Divide image I' into 3X3 pixel blocks.
Step 2: From the appropriate positions, Read the
        numbers of the blocks used for the embedding process to get the list of blocks [b'1, b'2, b'3…].
Step 3: For each block $B'_i$ in this list, do:
        From pixel (1,1) in the block, read the position of the minimum pixel ($P_{min}$) in the block before embedding.
        Using the data hiding flags, Read the positions of the pixels used to hide data in
        This block
        Retrieve the embedded bits from these pixels and store it in a temporary list (M')
        According to the minimum value $P_{min}$, restore the original values of the pixels
        before embedding using $P_{original}=P_{min}+P_b$ (the difference D stored in the
        embedding step).
Step 4: Concatenate all the bits in the temporary list M' to obtain the message bit
        stream $M_b$
Step 5: Merge all the blocks consisting of the original
        pixel values $P_{original}$ to reconstruct the original image (I)
Step 6: Convert the bit-stream $M_b$ back to its original
        format (using Unicode mapping for text data format for example) to get the original hidden message (M).
```

Algorithm 2. Data Extraction Algorithm

3.2 Data hiding/Extraction Diagrams

After describing the embedding process in details in the previous subsection, here comes the graphical representation of these steps. Figure 3 is a representation of our algorithm in the form of block diagram.

The Flowchart of the data hiding process is shown in figure 4. As described, the original message is first converted to its binary equivalent. Then we check to decide if we need to pad the original image with additional zeros to make the width and height a multiple of 3. The image is divided into blocks then the minimum pixel of each block is determined. The secret binary data is hidden in appropriate pixels and references to these pixels are stored. This process is repeated until the whole message is hidden in the pixels of the cover image to get the final stego image.

According to the steps in the data extraction algorithm mentioned above, here is a reduced model of the data extraction process is shown in figure 5.

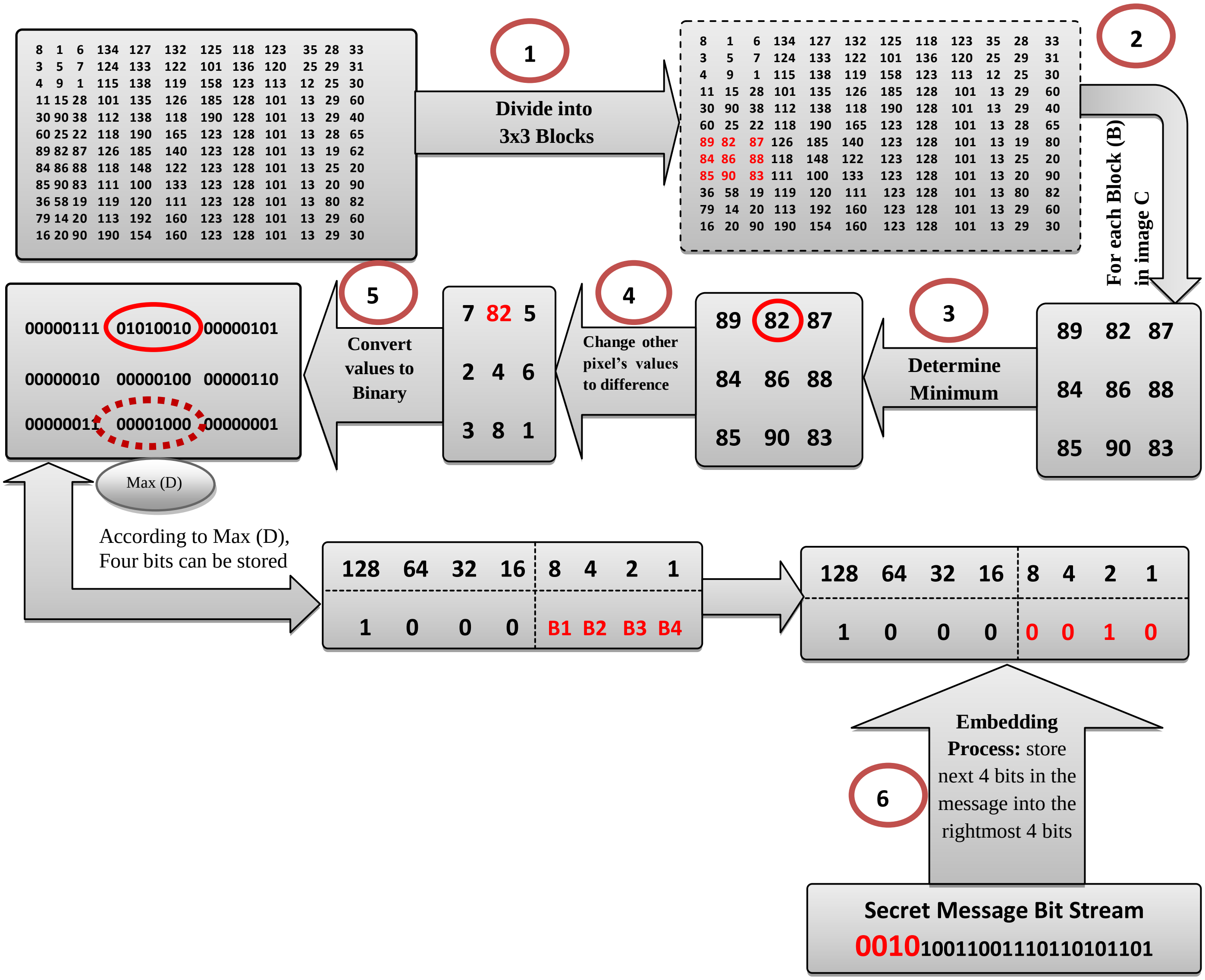


Fig. 3 Data hiding block diagram.

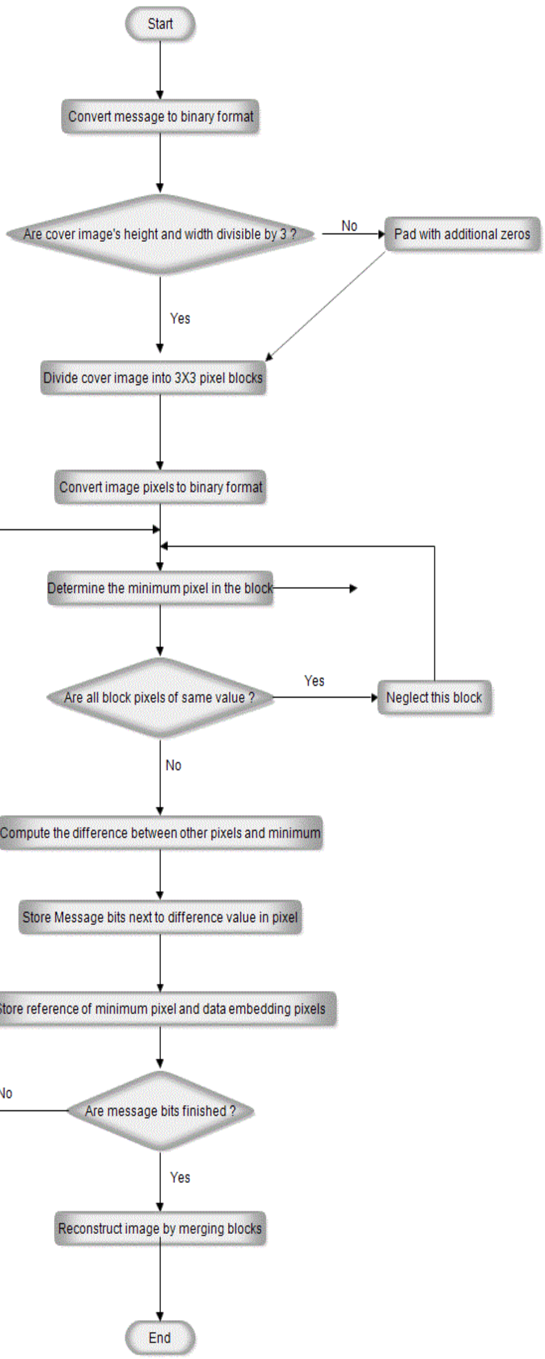

Start
Convert message to binary format
Are cover image's height and width divisible by 3 ?
No
Pad with additional zeros
Yes
Divide cover image into 3X3 pixel blocks
Convert image pixels to binary format
Determine the minimum pixel in the block
Are all block pixels of same value ?
Yes
Neglect this block
No
Compute the difference between other pixels and minimum
Store Message bits next to difference value in pixel
Store reference of minimum pixel and data embedding pixels
No
Are message bits finished ?
Yes
Reconstruct image by merging blocks
End

Fig. 4 Data Hiding Flow-Chart.

x

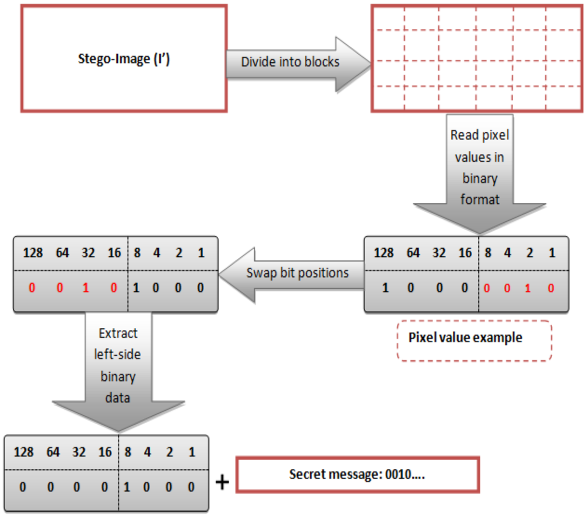


Fig. 5 Data Extraction block diagram.

3.3 Noise Detection

To avoid any kind of noise that may occur to the stego image during transmission to the receiver end (intentionally by an adversary or unintentionally during to transmission fault), we have added a checking step to our system. We just compared a unique property of the stego-image at the both sides of transmission. If the image is modified, then this unique property will not match that of the sent image. This unique property is the hash code. A **hash code** is a series of values that is used to identify an object during equality testing. If the hash code computed at the sender and the receiver are identical, we are now sure that no modifications were carried out over the image during transmission. Else, we now have a flag indicating that the image was modified. The next figure describes this process.

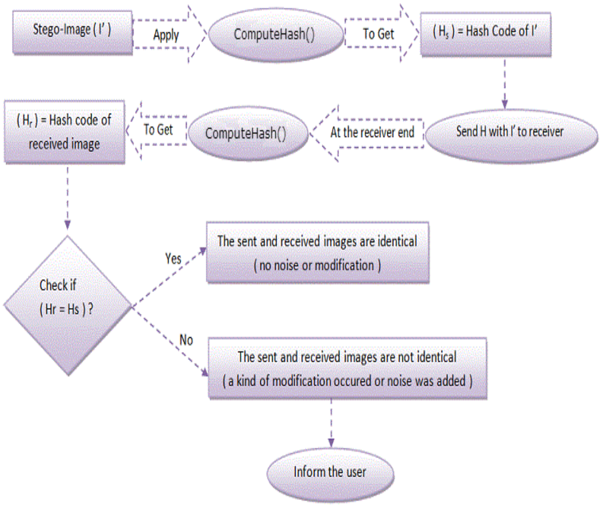

Stego-Image (I')
Apply
ComputeHash()
To Get
(H_s) = Hash Code of I'
Send H with I' to receiver
At the receiver end
ComputeHash()
To Get
(H_r) = Hash code of received image
Check if (Hr = Hs) ?
Yes
The sent and received images are identical (no noise or modification)
No
The sent and received images are not identical (a kind of modification occured or noise was added)
Inform the user

Fig. 6 Hash Value validation process.

Our data hiding system has many operational **advantages**: First of all is **great storage capacity** by using bits in most of the nine pixels of each block. Another advantage is **invisibility**. In other words any viewer may doubt in finding hidden data in an image, especially when it is distorted or not clear enough, but in case of a video, it may be considered a quality shortage of taking the video without concentrating on a special frame as frames are changed quickly. **More security** is ensured also as all people can view or download the video but only the intended person, who knows the embedding procedure and the extraction procedure as well, will analyze the video file – definitely some frames – to get the hidden message. **No observed change in cover file size** is an additional feature, adding data to an image file may make an observed increase in its storage size (especially when hiding large amount of data), but in case of video file there is no big difference in the clip's size. **Low Computation Complexity** as it is not very expensive computationally for embedding and extracting a hidden message to be carried out. At last **flexibility** is realized as this technique can be performed in a similar way using an animation or any other time changing picture.

4. Results and Evaluation

In terms of implementation and execution, our system consisted of four modules: the frame extraction module (performed using Matlab) whose output is the image (frame) that is used to hide data. Our algorithm works on the second and third modules. Second module is the data hiding module (applied using C# programming). The third is data retrieval module (applied using C# programming). The fourth module is frame re-construction which merges the stego image again in its position of the video file.

We have applied our system on a great number of images; most of them are in the four common image formats: GIF, PNG, TIFF and BMP. In the next figures we show an example of an image in each of these formats before and after applying our data hiding algorithm to prove that there is no visible degradation between them. We will also provide the histogram of each image example before and after data hiding to show the very little change in the statistical analysis of both images.

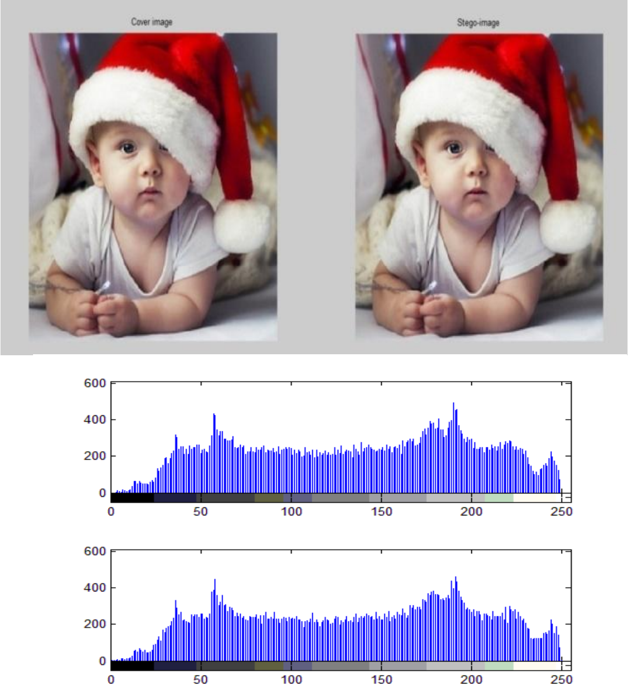


Fig. 7 "baby.bmp" (before, after, histogram before, histogram after)

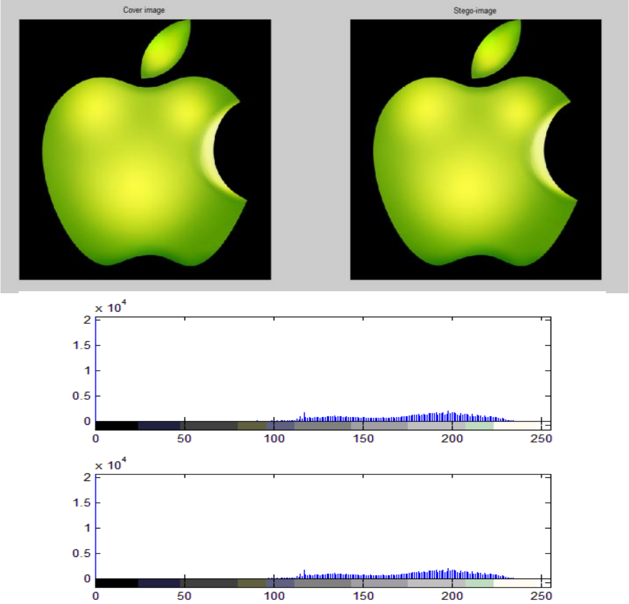


Fig. 8 "apple.bmp" (before, after, histogram before, histogram after)

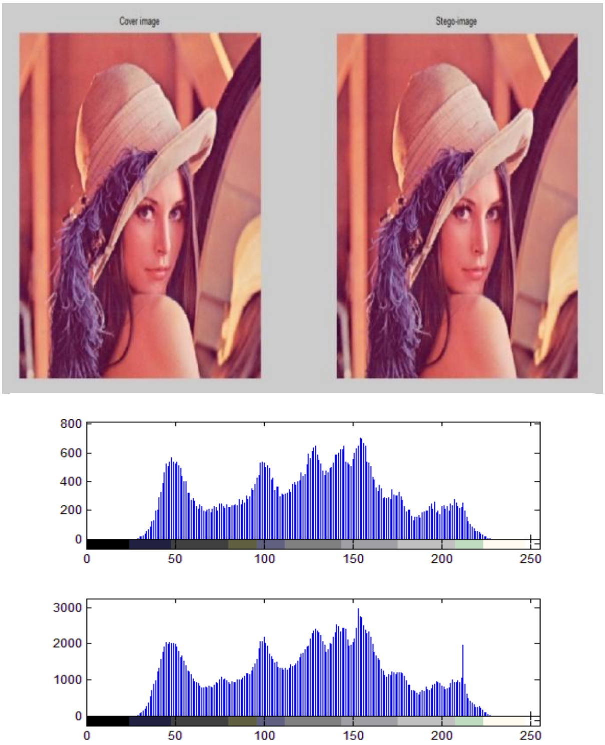


Fig. 9 “lena.tiff” (before, after, histogram before, histogram after)

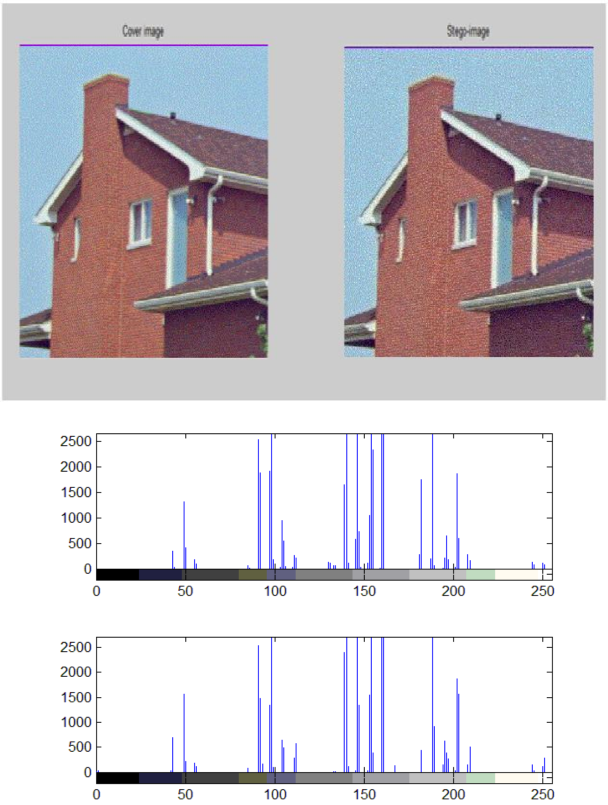


Fig. 10 “house.gif” (before, after, histogram before, histogram after)

To compare stego image with cover image, the images and results requires a measure of image quality; we have used some measurements to evaluate the performance of our system such as:

- **Mean Squared Error (MSE):** of an estimator is a way to quantify the difference between values implied by an estimator and the true values of the quantity being estimated. MSE is computed by Eq(1):

$$MSE = \frac{1}{mn}\sum_0^{m-1}\sum_0^{n-1}||f(i,j)-g(i,j)||^2 \qquad (1)$$

where **f** represents the matrix data of our original image, **g** represents the matrix data of our stego image in question, **m** represents the numbers of rows of pixels of the images and i represents the index of that row, **n** represents the number of columns of pixels of the image and j represents the index of that column.

- **Peak Signal-to-Noise Ratio** (**PSNR**): is an engineering term for the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. PSNR is calculated using Eq(2):

$$\text{PSNR}=10\log_{10}\left(\frac{255}{\sqrt{\text{MSE}}}\right)\text{dB} \qquad (2)$$

- **Retrieval Accuracy (RA):** is the ratio between the number of correct pixels retrieved and the original image’s number of pixels as in Eq(3):

$$RA = \frac{Number\ \ of\ correct\ \ bytes}{Total\ number\ \ of\ bytes\ in\ image} \times 100\% \qquad (3)$$

- **Average Difference (AD):** is the percentage of the modified pixel values between the cover and the stego images. AD is computed using Eq(4):

$$AD = \frac{1}{m\times n}\sum_{i=1}^{m}\sum_{j=1}^{n}(f(i,j)-g(i,j)) \qquad (4)$$

- **Embedding Capacity:** is the amount of bits that can be hidden in a cover object without causing statistically significant modifications or affecting the visual characteristics of the image as in Eq(5):

$$Capacity = \frac{number\ \ of\ bits\ used\ to\ hide\ data}{total\ number\ \ of\ bits\ in\ image} \times 100\% \qquad (5)$$

We have applied these measurements on our output images to evaluate our data hiding technique mathematically and the results for the examples discussed above are shown in table 1.

Table 1: Our data hiding technique evaluation parameters

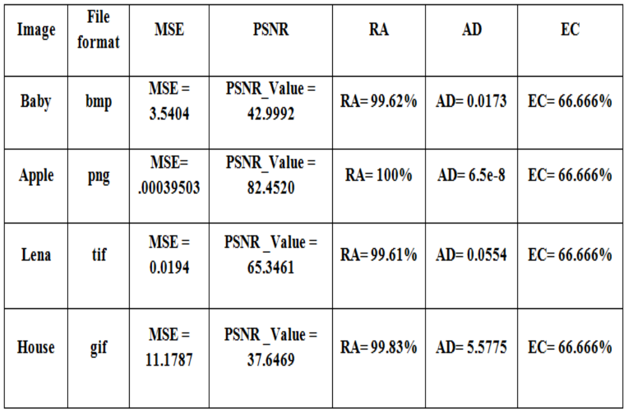


Image	File format	MSE	PSNR	RA	AD	EC
Baby	bmp	MSE = 3.5404	PSNR_Value = 42.9992	RA= 99.62%	AD= 0.0173	EC= 66.666%
Apple	png	MSE= .00039503	PSNR_Value = 82.4520	RA= 100%	AD= 6.5e-8	EC= 66.666%
Lena	tif	MSE = 0.0194	PSNR _Value = 65.3461	RA= 99.61%	AD= 0.0554	EC= 66.666%
House	gif	MSE = 11.1787	PSNR _Value = 37.6469	RA= 99.83%	AD= 5.5775	EC= 66.666%

5. Conclusion

As more people join the internet revolution, Intruders may reveal the information to others, modify it to misrepresent an individual or organization, or use it to launch an attack. One solution to this problem is through the use of steganography. Concealing information in ways that prevent the detection of hidden messages has become more important. Steganography techniques include a lot of communication methods that hide the message from being seen or discovered.

In this paper we have proposed a new algorithm in an interesting field for the researchers which shows the additional value of the combination of cryptography and steganography, so more security purpose are achieved and the level of secrecy is enhanced. We have overcome the limitations of the most common and ordinary LSB technique. In addition, we have achieved a high degree of robustness, embedding capacity, invisibility and accuracy. However the disadvantage of the proposed model is that it is susceptible to noise as we use spatial domain to hide the secret data. This can be improved in future scope if transform domain techniques are applied to hide the data.

References

[1] R. Anderson, and F. Petitcolas, "On the limits of steganography", IEEE Journal of Selected Areas in Communications, Vol. 16, 1998.
[2] Steganalysis: How to Detect Steganography, (http://www.xdatasecurity.com/about-steganography/how-to-detect-steganography.htm), last accessed: 25-02-2014.
[3] D. Seth, L. Ramanathan, and A. Pandey, “Security Enhancement: Combining Cryptography and Steganography”, International Journal of Computer Applications, Vol.9, November 2010.
[4] A. Kumar, and K. Pooja, “Steganography- A Data Hiding Technique”, International Journal of Computer Applications, Vol.9, November 2010.
[5] S. Bhattacharyya, I. Banerjee, and G. Sanyal, “Data Hiding Through Multi Level Steganography and SSCE”, Journal of Global Research in Computer Science, Vol.2, February 2011, pp.38-47.
[6] N.F. Johnson and S. Jajodia, "Exploring Steganography: Seeing the Unseen", Computer Journal, 1998.
[7] C. Ming, Z. Ru, N. Xinxin, and Y. Yixian, “Analysis of Current Steganography Tools: Classifications & Features”, International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2006.
[8] D. Neeta, K. Snehal, and D. Jacobs, “Implementation of LSB Steganography and Its Evaluation for Various Bits”, 2004.
[9] E. Lin and E. Delp, “A Review of Data Hiding in Digital Images”, Center for Education and Research Information Assurance and Security, Purdue University, 2006.
[10] C.K. Chan, and L.M. Chen,” Hiding data in images by simple LSB substitution”, Pattern Recognition, Vol.3, 2004, pp. 469-474.
[11] J. Fridrich and M. Goljan, “Digital image steganography using stochastic modulation," Symposium on Electronic Imaging, San Jose, 2003.
[12]] C.M. Wang, N.I. Wu, C.S. Tsai, and M.S. Hwang, “A high quality steganography method with pixel-value differencing and modulus function”, Journal of System Software, No. 81, 2008, pp.150–158.
[13] K.B.Shivakumar, K.B. Raja, R.K. Chhotaray, and S. Pattnaik, “Coherent Steganography using Segmentation and DCT”, 2010.
[14] C.Y. Yang, C.H. Lin, and W.C. Hu, “Reversible data hiding for high-quality images based on integer wavelet transform”, Journal of Information, Hiding Multimedia Signal Processing, Vol. 3, 2012, pp.142-150.
[15] R. Radhakrishnan, K. Shanmugasundaram, and N. Memon, “Data masking: A secure-covert channel paradigm," IEEE Multimedia Signal Processing, St. Thomas, US Virgin Islands, 2012.