A Secure Robust Gray Scale Image Steganography Using Image Segmentation

Article *in* Journal of Information Security · January 2016

DOI: 10.4236/jis.2016.73011

CITATIONS
8

READS
504

2 authors:

Mohammed J. Bawaneh

7 PUBLICATIONS **79** CITATIONS

SEE PROFILE

Atef Ahmed Obeidat

Al-Balqa' Applied University

24 PUBLICATIONS **64** CITATIONS

SEE PROFILE

Journal of Information Security, 2016, 7, 152-164
Published Online April 2016 in SciRes. http://www.scirp.org/journal/jis
http://dx.doi.org/10.4236/jis.2016.73011

Scientific Research Publishing

A Secure Robust Gray Scale Image Steganography Using Image Segmentation

Mohammed J. Bawaneh, Atef A. Obeidat

Department of Information Technology, AL-Huson Polytechnic, Al-Balqa Applied University, Salt, Jordan
Email: mohmmed_jazi@yahoo.com, atefob@gmail.com

Received 31 January 2016; accepted 8 April 2016; published 11 April 2016

Abstract

Steganography is the art of hiding a secret message in some kind of media. The main goal is not to hide only the secret message but also the existence of communication and secure data transferring. There are a lot of methods that were utilized for building the steganography; such as LSB (Least Significant Bits), Discrete Cosine Transform (DCT), Discrete Fourier Transform, Spread-Spectrum Encoding, and Perceptual Masking, but all of them are challenged by steganalysis. This paper proposes a new technique for Gray Scale Image Steganography that uses the idea of image segmentation and LSB to deal with such problem. The proposed method deals with different types of images by converting them to a virtual gray scale 24 bitmaps, finds out the possible segments inside image and then computes the possible areas for each segment with boundaries. Any intruder trying to find the transformed image will not be able to understand it without the correct knowledge about the transformation process. The knowledge is represented by the key of image segmentation, key of data distribution inside segment (area selection), key of mapping within each area segment, key agreement of cryptography method, key of secret message length and key of message extension. Our method is distinguishing oneself by one master key to generate the area selection key, pixels selection keys and cryptography key. Thus, the existence of secret message is hard to be detected by the steganalysis. Experiment results show that the proposed technique satisfied the main requirements of steganography; visual appearance, modification rate, capacity, undetectability, and robustness against extraction (security). Also it achieved the maximum capacity of cover image with a modification rate equals 0.04 and visual quality for stego-image comparable to cover image.

Keywords

Gray Scale Image Steganography, Image Segmentation, Random LSB

How to cite this paper: Bawaneh, M.J. and Obeidat, A.A. (2016) A Secure Robust Gray Scale Image Steganography Using Image Segmentation. *Journal of Information Security*, **7**, 152-164. http://dx.doi.org/10.4236/jis.2016.73011

1. Introduction

The network environment turns into a substantial portion of humans work life or even of their normal life. Network provides us with a rapid carrier to deliver the digital data from one point to another in a convenience way. Users look to enjoy the merits that networks have provided in sending and receiving process. Sometimes they may want to keep the secret communications or the copyrights, so a various technologies have been recognized as a helpful way in dealing with transferring data securely. The most common ones are steganography, cryptography, and watermarking. Steganography has always been confused with encryption and watermarking. Although they have the same goal (transferring data securely), but they are different in manipulating the data.

Steganography was derived from the Greek word steganos, meaning covered or secret, and graphy (writing or drawing) [1]. On the simplest level, steganography is the process of hidden writing, whether it consists of invisible ink on a paper or copyright information on a digital media. The purpose of steganography is to hide the existence of communication.

There are many ways to classify the steganography techniques [2]. They may be clustered according to the type of cover file (Image, Audio, Text or video steganography) or the manipulation procedure in embedding process (Injection, Substitution, Distortion or Generation Steganography).

This paper will present a random LSB method consolidated with the image segmentation. Here, the cover-object, where the data to be embedded, is always any type of images, the embedded data may be a text, an image or any type of files. The input image will be converted to 24 bits gray scale one in order to construct the host medium. Host image dimensions will be used to find out the segments which are employed in hidden process. Areas are built according to the selected segment that represents the segment key. The master key must be inserted in the hidden and extraction process by the user in aim to generate the other manipulation keys (selection, generation and cryptography). Secret message length, extension and the selected segment must be kept by the user in order to extract the secret message.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 describes the materials and methods. The experimental result is shown in Section 4. Finally, Section 5 concludes this paper with directions for future work.

2. Related Work

Steganography is a very old technology that was utilized in the time of the ancient Greeks to pass the secret messages from one area to another. They wrote the secrete message on a different hosts such as the head of slaves or the wood part of a wax then sent it to the receiver area [1].

In recent years, the steganography is widely known due to availability of different environment such as images, audios, videos and documents for hosting the secret message. Several applications and model were and will be built on those environments. In our work we concentered on the grayscale image steganography, so the ulterior drawing out presents some of the published studies in this field.

Nilizadeh [3] proposed a novel spatial domain method for steganography in RGB images. It utilized the blue layer of a certain block to hide the secret message. The difference between neighborhood pixels was used to build a block matrix of pixels. The pixels of block are chosen randomly to hold the required bits in order to improve the security. The proposed method is resistance against the frequency and spatial domain attacks as mentioned in results and analysis.

Chang and others [4] proposed a simple image steganography for secure communication. It based on the discrete cosine transformation (DCT) technique. It built a DCT for secret message and cover image. Also it stored the most significant information of each DCT secret block into the non-significant parts of each DCT cover block. According to results, the proposed method improved the human visual appearance.

Kadry and others [5] developed a new generating technique for image. It based on generating the stego-image from the text to be sent. The most important merits in the proposed method are unnecessarily for an additional image to hide the text, small size and powerfully execution comparing with other techniques.

Bawaneh [6] proposed a random LSB to embed the secret message inside the RGB color image. It used the linear congruent generator to finds out the location of random pixel in the cover image. The secret key was a combination of four parameters (Seed, Multiplier, Non-common factor, and Cycle length). The method utilized red, green or blue channel to hide the message bit. The selection of channel to be used for hiding based on the modification rate for each channel. The minimum modified rate was employed to embed secret message. Results

show that, the random LSB is better than the sequential LSB in term of visual appearance and satisfies sufficient security to secret message.

Juneja and others [7] proposed an improved LSB Steganography technique for images. They presented an embedding algorithm for storing encrypted messages in non-adjacent and random pixel according to edges detection filter and smooth areas of images. The method used 1, 3 or 4 bits randomly selected pixels to hide the message. The experimental results show that, the stego-image will not have any suspicion to detector in visual appearance also the length of message cannot be detected using the steganography detection methods.

Asthana and others [8] proposed a novel and more robust image steganography technique. It used the cover-image and text files representation in a given array that takes the alteration component based approach and the method of Palette Based Images. Message data were encrypted inside the stretched palette of image. Results show that, the demonstration between different images and their capacities.

Farn and others [9] proposed a new steganographic method for data hiding in jig swap puzzle images. The color image is divided into blocks then each block is rearranged to a new position according to the secret data and the stego-key. Experimental results show that, the method is undetectable and robust against modifications.

Sharma and others [10] proposed new steganographic algorithm by using 8 bit (grayscale) or 24 bit (color image) to improve the security against the steganalysis attack. It based on logical operations to embed the most significant bit of secret message in least significant bit of cover image. Results show that, the large similarity between cover and stego-image in visual appearance.

Udhayavene and others [11] introduced a new technique to improve the information security over the network. The proposed technique based on the idea of different key length (DKL). The result was compared with LSB algorithm according to Mean Square Error, Peak Signal Noise Ratio, Relative Pay Load and Rate of Embedding. It displayed that, the DKL algorithm is more efficient than the LSB algorithm.

Sandip Sadashiv and others [12] proposed an adaptive gray scale image steganography that used pixel value differencing and modulus function. The cover image is divided into three types of blocks (smoother, edge and error block). The smoother blocks embed small number of bits compared to edged block. The error block does not contain any bit from the secret message. The result shows that, the high visual quality for stego-image compared with previous work.

Ballesteros and others [13] proposed a scheme to hide a gray-scale image within an RGB image. It based on two bands election of the cover image with secret image histogram and a pixel searching process. The result shows that, the imperceptibility of the stego-image to hide the secrete message.

Raju and others [14] proposed a new technique that is different from standard LSB technique. It used key from the cover image to select the secret message bits. The PSNR was utilized to ensure the successful implementation of the proposed algorithm.

3. Materials and Methods

The main focus of this paper is the grayscale image steganography using the image segmentation. Our method looks to divide the cover image into several segments such that each one stores multi areas, but only one segment will be utilized in the hidden and the extraction process. The used method utilizes one master key to generate multi keys for area selection, cryptograph and pixel selection. It deals with any type of secret messages and cover images. Ulterior subsections show how the hidden and the extraction process will be carried out to reach the main goal from this work.

3.1. Hidden Process (LSB Substitution)

The method that is used to hide the data in the cover image is the Least Significant Bits (LSB) method. It's one of the most famous embedding methods. This method bases on replacing the least significant bits of a given byte from the cover image pixels. LSB is very fast and simple, but the distortion in the cover image is noticeable when the number of replacement bits exceeds three for each pixel [15].

In this work each bit requires one pixel (3 bytes) from the cover image, so each byte in the used pixel will store the same bit form the secrete message as shown in **Figure 1**.

The target pixels that will store secret message bits are selected randomly from different segment areas. Randomness process has two merits; firstly it increases the security of the data since the message bits are going to be scattered all over the image areas in a random way. Secondly it makes the effect of changing relatively less ap-

parent; due to the replacement pixels are not adjacent.

Hidden process needs for several requirements to accomplish their work. Those requirements can be summarized by: cover image conversion, cover image segmentation, segment area boundaries computation, and secret message conversion, number of areas determination, key initialization, area generators initialization and embedding process. **Figure 2** shows the frame work of the proposed hidden process; those steps can be summarized as ulterior and illustrated in next subsections.

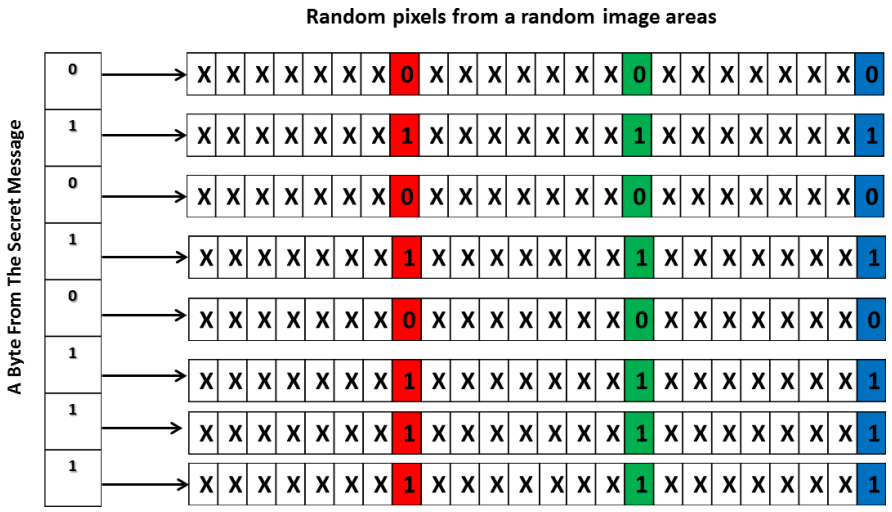

Figure 1. Pixels replacement.

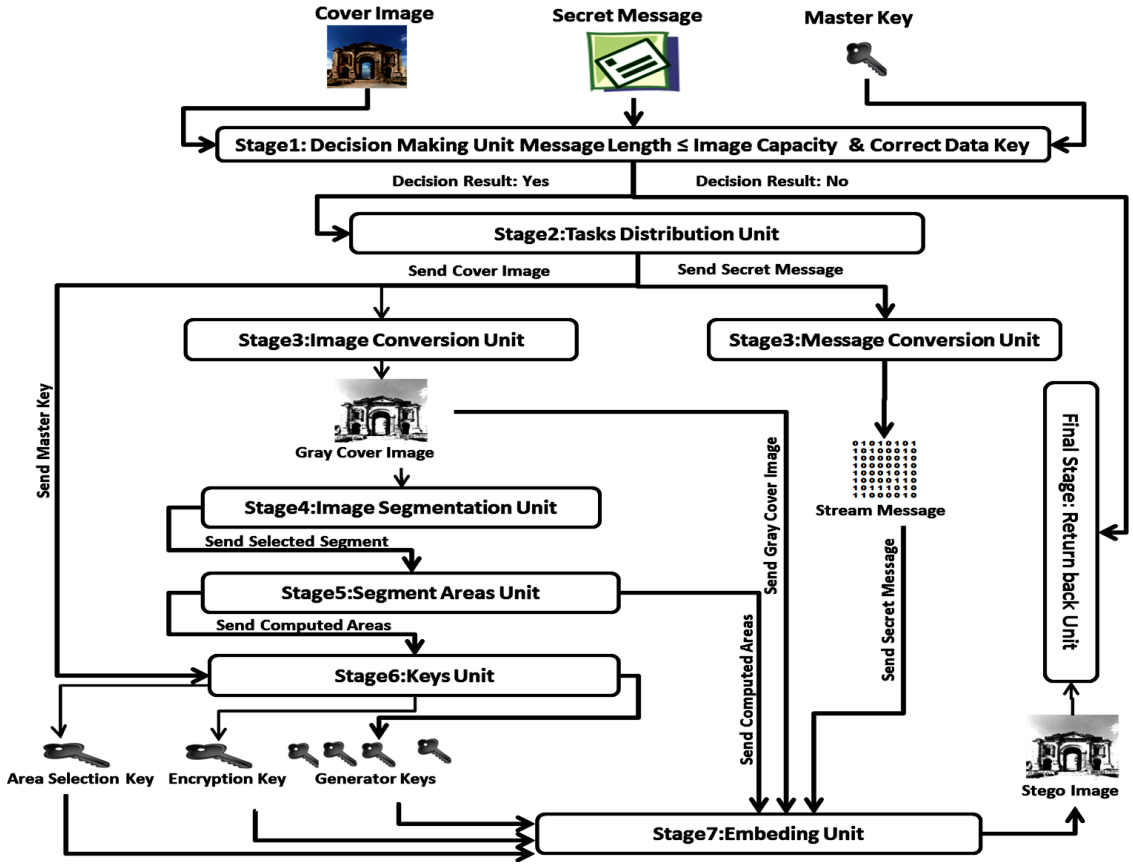

Figure 2. Frame work of hidden process.

Hidden process steps

Step 1: *Input the cover image, secret message, and master key.*
Step 2: *Convert the cover image to gray scale by using image conversion unit.*
Step 3: *Convert the secret message to stream of bytes by using message conversion unit.*
Step 4: *Compute the segments of cover image by using segmentation unit.*
Step 5: *Compute the areas for selected segment by using segment area unit.*
Step 6: *Generate the selection, encryption and distribution keys by using key unit.*
Step 7: *Embed the secret message inside cover image to get the stego-image.*

3.1.1. Cover Image Conversion

First of all, the image that is used as a cover object could be any type of images such as bmp or jpeg then the input image will be converted to 24-bits color depth. The system converts the 24-bits image to a virtual gray scale one by taking the mean value of red, green and blue colors. Result value of mean is set as the new value for red, green and blue colors as follows:

Step 1: *Color C = Get Pixel* (*x*, *y*), $0 \leq x <$ *Image Width and* $0 \leq y <$ *Image height*
Step 2: *Mean* = (*C. Red* + *C. Green* + *C. Blue*)/3.0
Step 3: *C. Red = C. Green = C. Blue = Mean*;
Step 4: *Set Pixel* (*x*, *y*, *C*)

3.1.2. Cover Image Segmentation

After completing the image conversion process, the system will deal with cover image as two dimensional array and start by the segmentation process. It takes width and height of the image in order to send them to the segmentation procedure. Inside the procedure, possible segmentations will be computed and given an index for each one. The computation of each segment is done according to height and width that were sent from caller. Segments will be stored and returned back to user as a linked list in aim to determine the selected one for the embedding process. **Figure 3** shows the form of returned linked list to the caller. Image width and Image height fields in the linked list will store the cover image height and width respectively. Row increment and column increment store the steps or differences between rows and columns respectively. The Used Index field determines the index of used segment in the embedding process, Area Head pointer points to list of areas while Head points to list of segments. Here, at the end of this stage row increment, column increment and Used Index values are set to 0 due to no segment selection, Head pointer points to first node in non-empty segmentation list and Area Head is set to null value.

Each segment stores a distinguishable index that is used when the user selects the required segmentation for embedding. Also it stores the number of rows and columns in which height and width can be divided to them. The ulterior steps will summarize the forerun description.

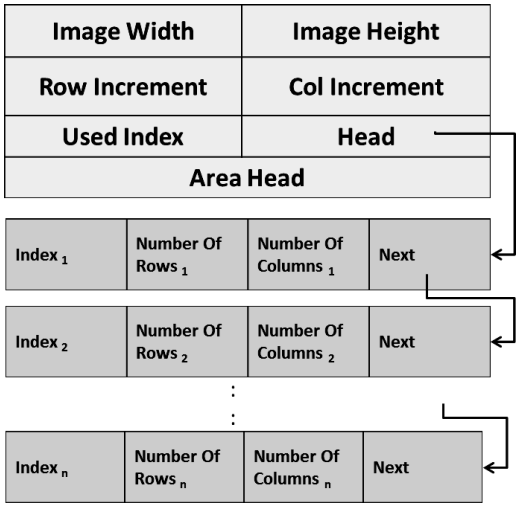

Figure 3. The returned segmentation list.

Step 1: *Set Segmination List. Head = null*
Step 2: *Set Segmination List. Set Image Dimension*
Step 3: *Set Index* = 0, *Used Index* = 0, *Col Increment* = 0, *Row Increment* = 0, *Area Head = null*
Step 4: *For # of Rows* = 2 *To Height*
IF Height Mod # of Rows = 0 *Then*
For # of Cols = 2 *To Width*
IF Width Mod # of Cols = 0 *Then*
Index = Index + 1
Add_Node (*Index*, *# of Rows*, *# of Cols*)
End If
Next For
End If
Next For
Step 5: *Return back the Segmentation list to user*

3.1.3. Preparing Segment Area Boundaries

Here, the user must select a segment form the returned segmentation list to be utilized in the embedding process. According to user selection, the system will find out the areas for number of rows and columns that were determined previously. Each area will hold an index, minimum width, maximum width, minimum height, maximum height, row index and column index, key_X and Key_Y as shown in **Figure 4**.

Index of area is used in the embedding process to determine the selected area from the area list. Minimum width, maximum width, minimum height and maximum height are utilized in the mapping process for finding the correct location inside the area. Col Index and row index are used in the selecting area generator for X and Y locations. Key_X and Key_Y are employed in area X Generator and Y Generator respectively to compute the current and the next values for x and y (x and y are location in the cover image). The next steps show how the values for each area are computed:

Step 1: *Get the index of segment and store it.*
Step 2: *Set Segmination List. Area Head = null*
Step 3: *Set Segment = Find Node* (*Index*)
Step 4: *IF Segment = null Then Return Back to caller*
Step 5: *Set Row Inc = Image Height/Segment. Rows*
Set Col Inc = Image Width/Segment. Cols
Set Min Width = Max Width = 0
Set Min Height = Max Height = 0
Set Segmination List. Used Index = Index
Set Area Index = Row Index =Col Index = 0
Step 6: *While Min Height < Image Height Do*
Begin
Max Height = Min Height + Row Inc − 1
Min Width = 0
Col Index = 0
While Min Width < Image Width Do
Begin
Max Width = Min Width + Col Inc − 1
Add Area Node( )
Area Index = Area Index + 1
Min Width = Max Width + 1;
Col Index = Col Index + 1
End While
Row Index = Row Index + 1
Min Height = Max Height + 1;
End While
Step 7: *Return back the segmentation list to caller.*

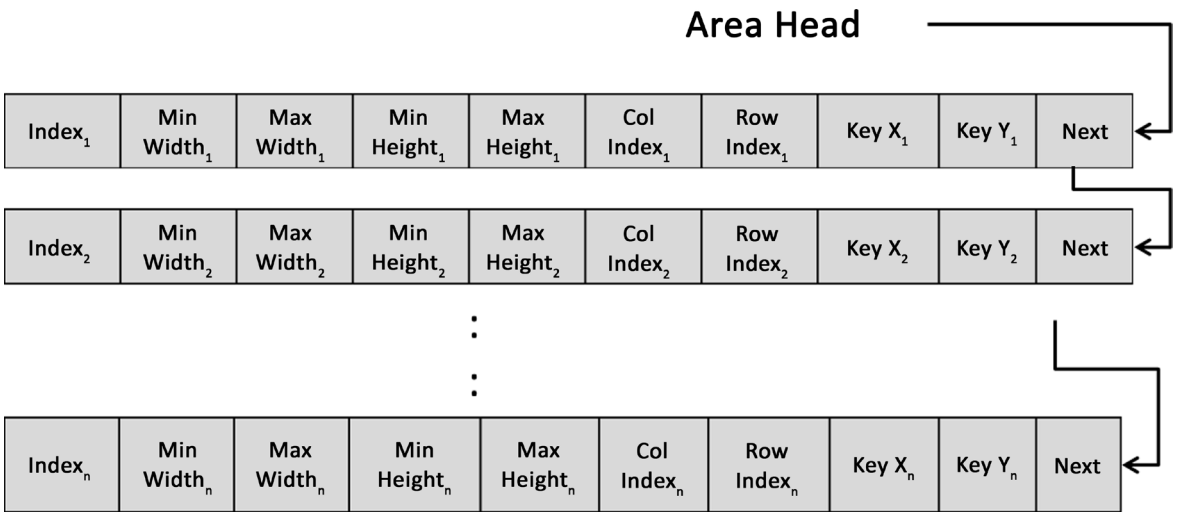

Figure 4. Linked list of segment areas.

3.1.4. Preparing the Secret Message

The secret message could be any type of files such as text, image, PDF, DOC…etc. The message file is converted to a stream of bytes. The system will compute size of input secret message, find out their extension and display the results to user in aim to keep them for the future employment in the extraction process.

3.1.5. Preparing the Number of Areas

Number of areas defines the total number of areas in the selected segment. It is computed as follow:

Step 1: *Segment* = *Find Node* (*Used Index*)
Step 2: *# of Areas* = *Segment. Rows* * *Segment. Cols*

3.1.6. Preparing the Key Value

The inserted master key value is utilized in multiple operations. It is used for data encryption, x & y generation and area selection as ulterior:

Step 1: *Set Segment Area* = *Area Head*;
Step 2: *Set Encryption Key* = *Key* * 23
Step 3: *Set Selection Key* = *Key* * 31
Step 4: *Set Generator Key* = *Key*
Step 5: *While Segment Area*! = *null Do*
Segment Area. Key X = *Generator Key* * 3
Segment Area. Key Y = *Generator Key* * 7
Generator Key = *Generator Key* * 5
Segment Area = *Segment Area. Next*
End While

3.1.7. Preparing the Area Generators

Each area in the segment requires two generators one for x and the other for y, so the number of X Generator will be equals to number of columns in segment, while the number of Y Generator equals to number of rows in the same segment. They are defined as ulterior:

Step 1: *X Generator* = *new LCG* [*Segment. Cols*]
Step 2: *Y Generator* = *new LCG* [*Segment. Rows*]

Linear Congruent Generator (LCG) method is used to generate a set of random number sequence over the interval [0, M − 1] without redundancy until completing the cycle. LCG has a set of preconditions that must be satisfied to generate the set of numbers without any redundancy [16]. Those conditions are: C and M has no common factors other than the value 1, (A-1) is multiple of every prime number that divides M and (A-1) is multiple of 4 if M Multiple of 4. The general formula of LCG is shown in **Figure 5**.

$$X_{i+1} = (A X_i + C) \bmod M$$

- X_{i+1} *refers to next random number*
- X_i *refers to current random number*
- *C refers to non-common factor*
- *A refers to multiplier*
- *M refers to cycle of generator*

Figure 5. LCG formula.

$$X_A = \frac{(X_R - X_{RMin})}{(X_{RMax} - X_{RMin})} * (MaxAreaWdith - MinAreaWdith) + MinAreaWdith$$

$$Y_A = \frac{(Y_R - Y_{RMin})}{(Y_{RMax} - Y_{RMin})} * (MaxAreaHeight - MinAreaHeight) + MinAreaHeight$$

- MaxAreaWdith, MinAreaWdith, MaxAreaHeight , MinAreaHeight are determined by the selected area
- X_{RMax} = Value of column increment, X_{RMin} =0
- Y_{RMax} = Value of row increment, Y_{RMin} =0

Figure 6. Mapping function.

3.1.8 Caesar Cipher

It is one of the most common and simplex cryptography techniques [17]. It bases on the idea of substitution for plain text letters. It must have a predefined list of letters that represents the first part of key, while the second part is the shifting value within the list. In this work the selection of substitution byte (shifting value) is done through the LCG random generator from a predefined list that is common between sender and receiver.

3.1.9. Mapping Function

The mapping function that converts a value from one range to another is employed in the computer graphics viewport mapping [18]. The function bases on the boundaries of selected areas. **Figure 6** shows how the values of XR and YR will be utilized to find out the area pixel XA and YA. However, the values of XR and YR are generated by the area random generator.

3.1.10. Embedding Process

After completing the main requirements for embedding process (Image conversion process, Secret message selection, Key setting, Number of areas determination, Area boundaries and Segment selection), the system will start data manipulation if the size of secret message compatible with the size of cover image. First of all, it takes the first byte of secret message stream, encrypts it by using the Caesar algorithm then the result will be converted to bits. For each bit in the selected byte, the system will select an area from the segment according to area index. Next the system will generate x and y through X Generator and Y Generator respectively, it's also update the values for Key_x and Key_y in the selected area. The values of x and y must be mapped to the boundaries of selected area. The pixel in (x, y) location will be fetched for manipulation and replacement. After completing that, the new color is set on bitmap buffer of cover image and the area index updated to next area. The process stills working until completing all bits of secret message bytes. The previous description can be summarized as follows:

Step 1: *Secret Stream. Position*=0
Step 2: *Area Index = Selection Key*
Step 3: *Load cover image in a bitmap buffer*
Step 4: *Set Capacity* = (*Image Width* * *Image. Height*)/8
Step 5: *IF Secret Length* > *Capacity Then*
    Return back to caller
Step 6: *while Stream. Position* < *Stream. Length Do*
    W = *Stream. Read Byte* ( )
    W = *Encryption* (*Encryption key*, *W*)
    Bits = *Convert To Bits* (*W*)

```
        For i = 0 To Bits.Length-1
Area Index = Random LCG (Area Index)
Segment Area = Find Area Node (Area Index)
        With Segment Area Do
Get XY (KeyX, KeyY)
        KeyX = X
        KeyY = Y
X = Map Value (X, Min Width, Max Width, 0, Col Inc)
  Y = Map Value (Y, Min Height, Max Height, 0, Row Inc)
     End With
     Color C = Get Pixel (X, Y)
IF C. Red [0]! = Bits[i] Then C. Red [0] = Bits[i]
     C = Color. From Argb (C. Red, C. Red, C. Red);
Set Pixel(X, Y, C);
     Next For
End While
```

3.2. Extraction Process

In order to extract the secret message from the stego-image, you should be aware of four important things; first you should know the length of secret message, this means you should know the number of bytes that were hidden. Secondly, you should know the key that was used to hide the data. Thirdly, you should know the segmentation that was utilized in hiding the data. And finally, you should know the extension of secret message (type of message). Once you have these information you can extract the message from the stego-image using the ulterior steps as shown in **Figure 7**.

```
Step 1: Determine segment to be used on stego-image
Step 2: Compute segment area boundaries
Step 3: Compute number of segment areas
Step 4: Initialize the keys
Step 5: Initialize the area generators
Step 6: Load stego-image in a bitmap buffer
Step 7: Set Capacity= (Image Width * Image Height)/8
Step 8: IF Given Length > Image Capacity Then
Return back to caller
Step 9: Set Area Index= Selection Key
Step10: Set Byte Counter = 0
Step 11: While Byte Counter < Message Length Do
          Bits = new byte [8]
          For j = 0 To j < Bits. Length-1
Area Index = Random LCG (Area Index)
Segment Area = Find Area Node (Area Index)
          With Segment Area Do
Get XY (Key X, Key Y)
Key X = X
Key Y = Y
  X = Map Value (Min Width, Max Width)
                Y=Map Value (Min Height, Max Height)
             End With
  Color C = Get Pixel (X, Y)
Bits [j] = (byte) (C.R & 1)
  Next For
B = Convert To Byte (Bits)
B = Decryption (Decryption key, B)
Write Byte (Stream Buffer, B)
```

Byte Counter = *Byte Counter* + 1
End While
Step 12: *Copy Buffer* (*Stream Buffer*, *Filename*, *Extension*)
Step 13: *Return back to caller*

The extraction procedure carries out the steps from one to nine as in the hidden procedure, so no need to re-explain them, also X, Y generation and mapping are done as in the hidden process. Decryption process uses the Caesar algorithm that was used to encrypt the secret message bytes.

4. Results and Analysis

The proposed system was tested using a lot of images and secret messages. **Table 1** displays the data set that were utilized in the evaluation process.

Table 2 shows the number of segments, possible segments and total number of areas for each segment in Lenna and Jarash as a cover image.

The main issues that were taken in our consideration for evaluating the system are: visual appearance of ste-

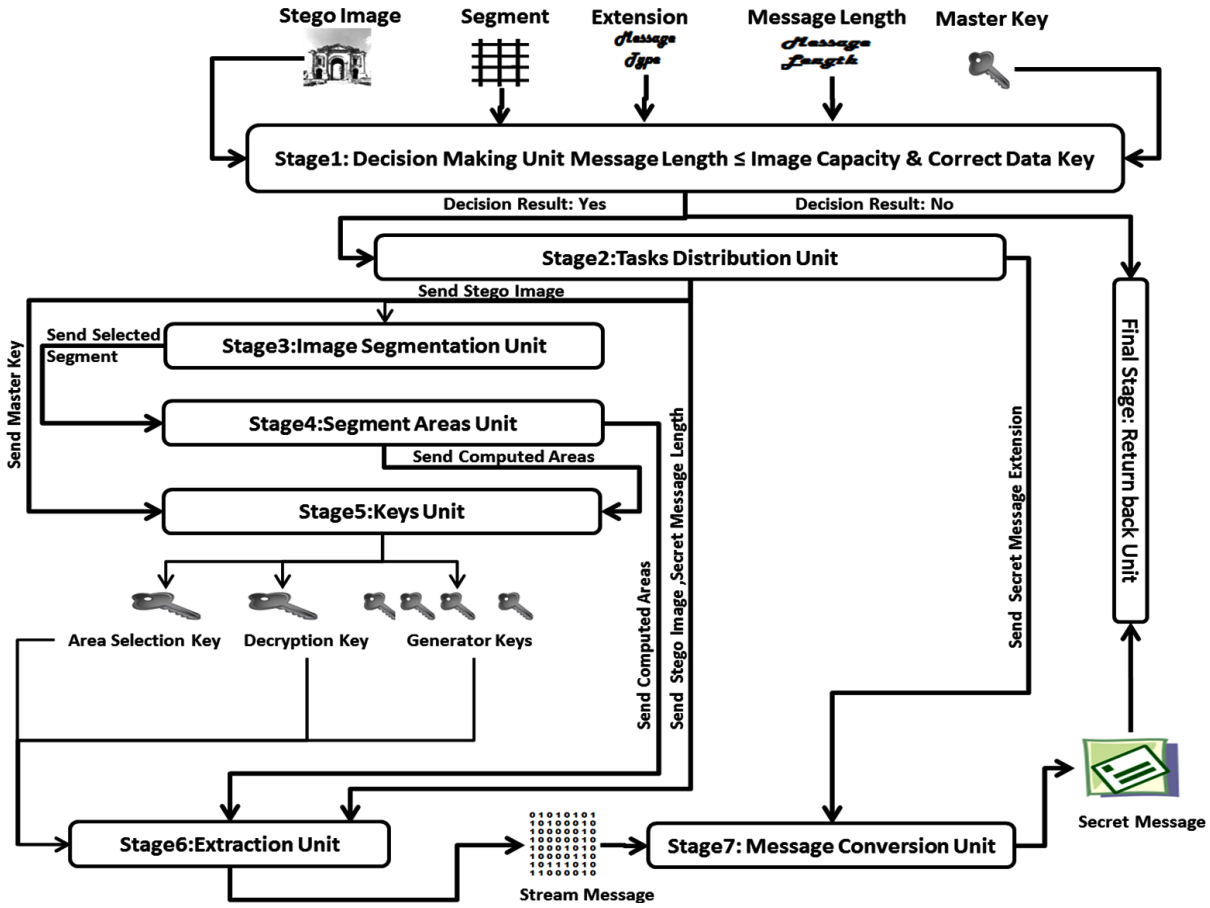

Figure 7. Frame work of extraction process.

Table 1. Evaluation data set.

Secret Message	Size	Cover Image	Size	Dimension [W × H]
Secret 1	508 Bytes	Jarash	717 KB	700 × 350
Secret 2	41 Bytes	Lenna	768 KB	512 × 512
Secret 3	1 KB			
Secret 4	13.7 KB			

go-image compared to the cover one, modification rate, capacity of cover image, robustness against modification, detecting ability and security.

Visual appearance is evaluated through the human eyes or the magnifying classes. The stego-image is compared with cover image in order to detect a noise or irregularity in the final form of image. To test the appearance, we embedded secret 1, secret 2, secret 3 and secret 4 in the images of Lenna and Jarash with two segment area for each one. The stego-images in the different cases gave a result similar to the cover one as shown in **Figure 8**. The main reasons for that were the equality distribution of bits within cover image and the features of image itself (gray scale) will hide the noise.

One criteria can be utilized to select the best segment is the modification rate. It is computed by dividing the number of modified pixels over the cover image length. The MR value bases on the user selected image. It helps in selecting the best segment for hiding the effect of visual appearance in the stego-image. MR varies between one segment and another for the same secret message as shown in **Table 3**, so the sender should select the minimum MR segment. Variation in MR results from different distribution of pixels. Huge segment areas may give

Table 2. Possible segments and areas.

Lenna Image		Jarash Image	
64 Segments		160 Segments	
Segment	# of Areas	Segment	# of Areas
2 × 2	4	2 × 2	4
2 × 4	8	2 × 4	8
2 × 8	16	2 × 5	10
:	:	:	:
:	:	:	:
256 × 256	65,536	175 × 350	61,250

Table 3. Modification rate for segment.

Segment	Lenna Image	Segment	Jarash Image
[2 × 2]	0.0076	[2 × 2]	0.0078
[64 × 32]	0.0070	[70 × 20]	0.0062

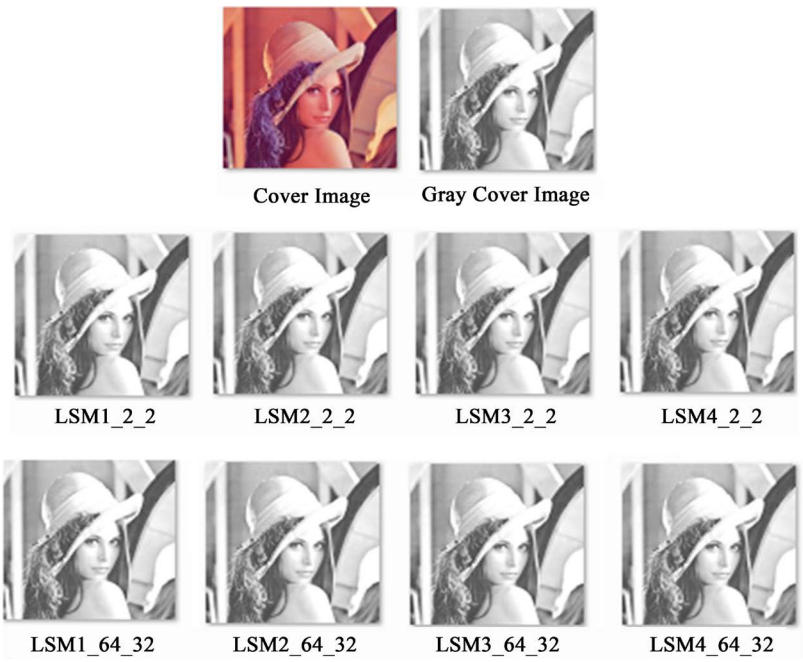

Figure 8. Different cases of Lenna stego-image.

Figure 9. Lenna with maximum size secret message.

a low MR due to regular and equal distribution of pixels within those areas.

Capacity of cover image is determined by the dimensions of image. As we mentioned previously, each secret message bit requires one pixel (3 bytes) from cover image, so the maximum capacity is computed by dividing cover image length on the value 8 (one byte). Lenna image can store in the maximum state 3264 bytes. To evaluate the proposed method at the maximum capacity state, we got a secret message of size 3 KB and embedded it in Lenna image; the result stego-image was very similar in visual appearance to cover one with a modification rate only 0.04 as shown in **Figure 9**.

Information is considered a robust when it is embedded, hidden inside an image and encountered any modification to image. In general, most of image steganography techniques are not robust against the modification and our proposed technique is one of non-robust methods.

Undetectability as a feature was applied in the proposed method because the used pixel consists of three colors (Red, Green and Blue), each one stores the same value and the bit is distributed randomly within a random area, according to this the hidden bits under an image are not doubtable or suspicious.

The proposed method is considered a secure one because no attacks view the hidden message unless they have a full knowledge about the keys. The extraction process requires multi keys for extracting the secret message. Those keys are: master key, length of message, used segment and type of secret message. Master key is utilized to build other keys for cryptography, key of area selection and keys of pixel location. For example to extract the message inside a given image, we will require a set of keys as follow:

- Image consists of N segments (N keys) and the user must select one of them.
- The Selected segment holds a number of areas, selection of random area needs a key and the initial key will be regenerated in next area selection.
- Each area requires two keys to select the pixel and the initial keys will be regenerated in next pixel selection.
- Decryption of extracted bit requires a key that is regenerated in next decryption process.
- Secret message length and extension are also considered two keys
- According to that, the extraction process is very hard, complex and secures one.

5. Conclusion

This paper presented a novel gray scale steganographic method for information security. It based on the idea of image segmentation to give an improved steganography method for embedding secret message bit in least significant bytes of random pixel in a random area within the grayscale cover image. The main goal of this paper is to construct a solution that is robust against the attack, effective in generating stego-image and very hard for visual appearance to predict and detect the existence of secret message. Experimental results show that, the proposed method satisfied most of the security requirements (visual appearance, security, undetectability), explained adaptability of grayscale cover image as a host to hide the secret messages and improved the data hiding capacity of host image by utilizing all the pixels. The future work of this paper proceeds to extract the secret message from stego without a master key. We look to distribute the master key inside in the stego-image in a way that allows the receiver extracting the secret message without any pre-knowledge about it.

References

[1] Sneha, B. and Gunjan, B (2014) Data Encryption by Image Steganography. *International Journal of Information and Computation Technology*, 4, 453-458. http://www.irphouse.com/ijict.htm

[2] Abdelwahab, A.A. and Hassaan, L.A. (2008) A Discrete Wavelet Transform Based Technique for Image Data Hiding. *Radio Science Conference*, Egypt, March 2008, 1-9.

[3] Nilizadeh, A.F. (2013) Steganography on RGB Images Based on a Matrix Pattern Using Random Blocks. *International Journal of Modern Education and Computer Science*, **4**, 8-18. http://dx.doi.org/10.5815/ijmecs.2013.04.02

[4] Chang, C.-C., Lin, P.-Y. and Chuang, J.-C. (2010) A Grayscale Image Steganography Based upon Discrete Cosine Transformation. *Journal of Digital Information Management*, **8**, 88-94.

[5] Kadry, S. and Nasr, S. (2013) New Generating Technique for Image Steganography. *Lecture Notes on Software Engineering*, **1**, 190-193. http://dx.doi.org/10.7763/LNSE.2013.V1.43

[6] Bawaneh, M.J. (2014) A Novel Approach for Image Steganography Using LCG. *International Journal of Computer Applications*, **102**, 34-38.

[7] Juneja, M. and Sandhu, P.S. (2013) An Improved LSB Based Steganography Technique for RGB Color Images. *International Journal of Computer and Communication Engineering*, **2**, 513-517. http://dx.doi.org/10.7763/ijcce

[8] Asthana, A. and Johri, S. (2012) An Adaptive Steganography Technique for Gray and Colored Images. *International Journal of Advanced Research in Computer Science and Software Engineering*, **2**, 340-345.

[9] Farn, E.-J. and Chen, C.-C. (2009) Novel Steganographic Method Based on Jig Swap Puzzle Images. *Journal of Electronic Imaging*, **18**, Article ID: 013003. http://dx.doi.org/10.1117/1.3073979

[10] Sharma, V.K. and Shrivastava, V. (2012) A Steganography Algorithm for Hiding Image in Image by Improved Lsb Substitution by Minimize Detection. *Journal of Theoretical and Applied Information Technology*, **36**, 1-8.

[11] Udhayavene, S., Dev, A.T. and Chandrasekaran, K. (2015) New Data Hiding Technique in Encrypted Image: DKL Algorithm (Differing Key Length). *Eleventh International Multi-Conference on Information Processing*-2015 (*IMCIP*-2015). www.sciencedirect.com
http://dx.doi.org/10.1016/j.procs.2015.06.093

[12] SandipSadashiv, S. and Rao, P.V. (2015) A Steganographic Method by Using Adaptive Algorithm for Gray Scale Images. *International Journal of Electrical*, *Electronics and Computer Systems* (*IJEECS*), **3**, 9-13.

[13] Ballesteros, D.M., Renza, L.D. and Rincon, R. (2015) Gray-Scale Images within Color Images Using Similarity Histogram-Based Selection and Replacement Algorithm. *Journal of Information Hiding and Multimedia Signal Processing*, **6**, 1156-1166.

[14] Raju and Dhanda, M. (2015) An Improved LSB Based Image Steganography for Grayscale and Color Images. *International Journal of Current Engineering and Technology*, **5**, 3295-3297.

[15] Prajapati, H.A. and Chitaliya, N.G. (2015) Secured and Robust Dual Image Steganography: A Survey. *International Journal of Innovative Research in Computer and Communication Engineering*, **3**, 30-37.

[16] Byron, M.J.T. (1984) Elements of Simulation. Chapman and Hall, USA, 57-64.

[17] Stallings, W. (2005) Cryptography and Network Security Principles and Practices. 4th Edition, Prentice Hall, Upper Saddle River, 36-38.

[18] Hearn, D.D. and Baker, M.P. (1994) Computer Graphics. 2th Edition, Pearson, A Paramount Commercial Company Englewood Cliffs New Jersey, 219-220.