A Novel RGB Image Steganography Using Simulated Annealing and LCG via LSB

Mohammed J. Bawaneh[1], Emad Fawzi Al-Shalabi[2] ,Obaida M. Al-Hazaimeh[3]

Information Technology Department, Al-Huson University College,
Al-Balqa Applied University, Irbid, Jordan

Abstract
The enormous prevalence of transferring official confidential digital documents via the Internet shows the urgent need to deliver confidential messages to the recipient without letting any unauthorized person to know contents of the secret messages or detect there existence . Several Steganography techniques such as the least significant Bit (LSB), Secure Cover Selection (SCS), Discrete Cosine Transform (DCT) and Palette Based (PB) were applied to prevent any intruder from analyzing and getting the secret transferred message. The utilized steganography methods should defiance the challenges of Steganalysis techniques in term of analysis and detection. This paper presents a novel and robust framework for color image steganography that combines Linear Congruential Generator (LCG), simulated annealing (SA), Cesar cryptography and LSB substitution method in one system in order to reduce the objection of Steganalysis and deliver data securely to their destination. SA with the support of LCG finds out the optimal minimum sniffing path inside a cover color image (RGB) then the confidential message will be encrypt and embedded within the RGB image path as a host medium by using Cesar and LSB procedures. Embedding and extraction processes of secret message require a common knowledge between sender and receiver; that knowledge are represented by SA initialization parameters, LCG seed, Cesar key agreement and secret message length. Steganalysis intruder will not understand or detect the secret message inside the host image without the correct knowledge about the manipulation process. The constructed system satisfies the main requirements of image steganography in term of robustness against confidential message extraction, high quality visual appearance, little mean square error (MSE) and high peak signal noise ratio (PSNR).

Keywords:
Steganography, RGB, Linear Congruential Generator, LSB, Simulated Annealing.

1. Introduction

The great reliance on the Internet in the field of exchanging information and various digital messages demonstrates a necessary need to deliver the messages to the recipient without allowing any unauthorized person to read, modify, or destroy the contents of the messages. Authorized user aspires **to** transfer data and copyrights securely, thus a several ways have been constructed to achieve this necessity. Steganography, cryptography and watermarking manipulate the confidential data differently, but they share the target of transferring data securely.

The steganography as a science was used in the past and invented based on the principle of hiding secret messages in a medium that provides the optimal cover for the secret message. In simple words, the steganography refers to the process of hiding data into a hard or soft medium that could paper, leather, wood, wax or digital media [20].

The type of digital steganography is determined from the used cover media and data manipulation procedure, the cover media data transporter may be image, audio, text or video. Data injection, substitution, distortion and generation are the common manipulation procedures in the data hiding process [1].

SA is an optimization search method that can be used to solve linear and non-linear problems [6]. The main idea of SA was taken from the metal heating and cooling environment. In initial state, the searcher begins with the worst case until reaching the best case through controlling the temperature and fitness function value. More ever, SA procedure uses the random manner in searching process in aim to maximize or minimize solution, thus the modifications that will change solution to be worse than the current on are rejected. The main demerit of SA procedure arises when manipulating a huge number of random choices that require a huge time of processing.

The LSB is used to hide the digital message in a digital medium by substituting the right most bit of cover medium with a bit from message, thus the data can be analyzed and extracted very easily if applied in its simple form [8],[10].

This paper presents an innovation robust color image steganography system that integrates SA, LCG, Cesar cryptography algorithm and LSB technique in one framework. SA works within an RGB color image as searcher about an optimal solution vector for data hidden in a set of random vectors that were generated via LCG. The host image pixels and data of secret message are

Manuscript received January 5, 2021
Manuscript revised January 20, 2021
https://doi.org/10.22937/IJCSNS.2021.21.1.19

used in SA fitness function in aim to determine the best generated vector. SA procedure uses the three channels of host image (Red, Green and Blue channels) in order to construct three optimal vectors such that each one of them will be utilized with a specific part of secret message.

The constructed system deals with secret message as binary file, thus the message could any type of data. It divides the secret message file into three parts of bytes such that each one is hidden in a color channel. Cesar algorithm is employed to encrypted message bytes before embedding them via LSB and decrypt data bytes after the extraction process. LSB was utilized to substitute the bits of each secret message part in red, green and green computed vector.

LCG is a common random generator that has a set of prerequisite conditions must be met before starting random data generation as shown later on [7]. The main goal of using LCG procedure is to prevent the duplication of pixels in constructed SA vector.

In embedding and extraction process, the user must insert a set of keys to accomplish their task; those keys will be used in LCG (Initial seed) and Creaser algorithm (Cryptography key). The SA parameters as keys are set to be fixed at sender and receiver sites in order to reduce the number of transferring key; however they might be changed according to agreement between two parties. Secret message length and type are combined with message data then embedded within host image, after that the authorized extractor can retrieve them by inserting the correct extraction knowledge.

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

Different steganography techniques tend to increase capacity, including improvement with low visual distortions on a color image as a medium. Consequently, sophisticated combinations of algorithms have been produced that work with LSB and use a medium such as grayscale images [3],[4], other methods, compress a message before hiding it in a digital medium as in [14]. Other studies select video frames using a Linear Congruential Generator LCG to embed a secret message in a digital video or RGB digital image [9][15].

Iranpour and others [11] divides the cover image in block of pixels in which order of pixels were changed according Hamiltonian path. They use Hamiltonian paths in order reduce the distortion of LSB substitution. The distortion rate of embedding process was computed by using a set of image as a cover medium.

Raphel and others [18] developed a visual cryptography schema in which the secret image was encoded into n shares that will be later on propagated to n participants. Next, the shares will be combined and retrieved the whole message.

Parvez [16] uses the actual pixel color of the channel (R, G, or B) to select no data bits to store in each channel. This technique increases the capacity of cover media.

Bawaneh[3],[4] constructed gray scale image system steganography that search about best solution vector in gray host image via SA and Intelligent Water Drop(IWD). The system was robust against detection and analysis as mentioned in results.

Sitompul and others [15] proposed a method to enhance the security of the embedded image by using a Linear Congruential Generator (LCG) to select a random position for inserting each bit of the encrypted character at the LSB of each R, G, and B layer of the cover image. Results showed that the quality of the cover image affected insignificantly by the embedding process.

A. Singh [2] proposed improved LSB for RGB color images by dividing the color image into three planes, and then inserting the message in each one in a way that enhances the quality and produces a high level of embedding capacity.

Xuefeng [22] introduces a palette-based image representation from the perception that digital images in most states use a small part of the available RGB color space by quantizing similar colors in the palette.

Bawaneh [13] built a random LSB image steganography system by using LCG. The system hides the secret message inside the RGB color image according to random locations that were generated by LCG. In constructed system the results show the preference of random LSB over sequential one in term of visual appearance and security.

Bawaneh and others [5] developed a grayscale image steganography system that uses idea of image segmentation. Secret message is distributed between different segments of image according to segmentation key. The system was robust against detection and analysis as mentioned in results.

Kadry and others [12] constructed an image steganography in which the Stego-image is built from the secret text message to be sent. In the proposed method there is no cover image to generate stego image which makes technique novel and more powerful one comparing to other existing ways.

Sharma and others [19] developed an image steganography system that works 8bit or 24 bit. They used the logical operations to hide most significant bit of secret message in least significant bit of cover image.

Cover and stego image have a high comparable visual appearance as mentioned in results.

Raju and others [17] developed a new method that generates a key from the image medium in order to choose the suitable secret message bits to be hidden. PSNR was employed to evaluate the proposed algorithm and quality of result stego image.

3. Materials and Methods

The proposed approach looks to find the optimal solution inside an RGB image (host image) for embedding a secret message by constructing a path using SA and LCG. The goal of this work is to build robust, less sniffing and comparable visual appearance StegoImage. The Optimal solution consists of a set of pixel locations from the host image that will be found randomly using LCG and optimized via SA. Multiple steps should be accomplished to reach the final target; those phases were built based on hidden and extraction processes. Every one of them can be explained as a state machine that has inputs, processing and outputs as shown next subsections.

3.1 Hidden Process

Several prerequisites are required for this stage to carry out the embedding process. The requirements are represented by inputs fetching, data conversion, vectors generation and secret data embedding. Secret message, host image, LCG seeds, encryption key, and SA parameters are the main inputs of the constructed system. Entered data require a manipulation methods and the data format conversion is intended to be compatible with processing stages. SA generates an optimal vector from the host image through its parameters and LCG. In final stage, the result vector will be used to store secret data inside StegoImage. The distribution of tasks between different units in constructed system is shown in Fig. 1**.** The details of carrying out each stage will be illustrated in ulterior subsections.

3.1.1 Input Fetching Unit

It gets the required inputs from the interface in order to be used in the next units. Host image, secret message and keys are the main inputs that must be inserted into the system. The host image could be any type of image such as BMP or JPG format because the system will convert it to RGB format then store the resulting image in a frame buffer or bitmap. Each location in the frame buffer has three colors (Red, Green and Blue) and it can store three bits from a secret message. The secret message is manipulated as a stream of bytes thus it could be any type of files such as text, image, audio and video. System input keys are represented by SA parameters, LCG seed for red data, LCG seed for green data, LCG seed for blue data and encryption key.

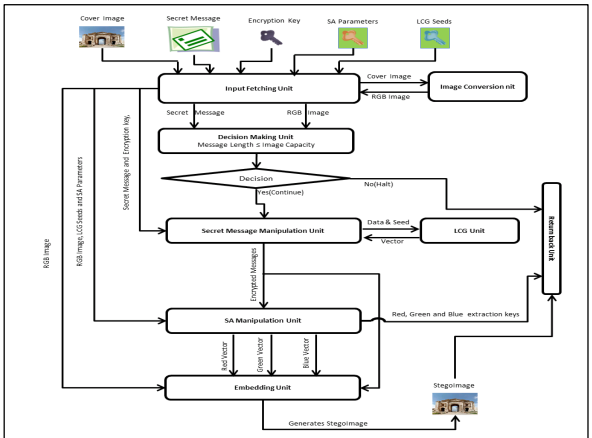

Fig. 1 Hidden process framework

3.1.2 Decision Making Unit

The unit utilizes the width and height of the frame buffer in computing size of the host image and available locations inside it. After that, it retrieves the size and extension of the secret message that will be used in making a decision about hosting compatibility between image and a secret message. As mentioned previously, each pixel can hold three bits, thus the number of available locations is computed by multiplying frame buffer size with value 3 and then dividing result over value 8(one byte). More ever, the length of secret data will be divided between three channels (Red, Green and Blue) such that each one has a part of message data. The size of data between channels may be various due to different sizes of secret messages. After making computation for available locations and message length the decision will be made. Fig.2 shows the main steps that are carried inside the decision making unit.

3.1.3 Secret Message Manipulation Unit

The data of secret messages are manipulated as a stream of bytes according to the size of each channel that was computed in the decision making unit. After reading data of secret messages from their source and before distributing them between channels, they are encrypted using Caesar algorithm. It was employed for the cryptography process in hidden and extraction stages. Caesar algorithm is a simple well-known cryptography technique [21]. It bases on idea of replacing the plain text

letters with other letters through a shifting value and result value has the same length as the original one. The shifting value is considered as a part of the key that must be common between the sender and receiver. The key of Caesar algorithm has two parts; the first one is a predefined list of letters, while the other part is the shifting value inside list. As soon as data read and encrypted, the distribution process of secret data starts their task, it divides data into three byte lists (RedList, GreenList and BlueList) later on each one of them will be converted to a list of bits. Fig. 3 shows how the secret message manipulation unit accomplishes its task internally.

```
Step1: Set MessageLength = SecretMessageFileName.Length;
Step2: Set NumberOfLocations = (H.Height * H.Width * 3) / 8.0;
Step3: Set MessageDistrubtion = MessageLength / 3.0;
Step4: If (NumberOfLocations < MessageDistrubtion * 8) Then
          Exit
       End If
Step5: Set MessageModules = MessageLength % 3;
Step6: Set MessagePartionSize = Integer( MessageLength / 3);
Step7: Set SizeOfRedData = SizeOfGreenData = SizeOfBlueData
       = MessagePartionSize;
Step8: If (MessageModules == 1) Then
          SizeOfRedData = SizeOfRedData + 1;
       Else
          If (MessageModules == 2)
            SizeOfRedData = SizeOfRedData + 1;
            SizeOfGreenData = SizeOfGreenData + 1;
       End If
```

Fig.2 Decision making unit steps

```
Step1: Define RedData = new byte[SizeOfRedData];
Step2: Define GreenData = new byte[SizeOfGreenData];
Step3: Define BlueData = new byte[SizeOfBlueData];
Step4  ByteStream =ReadSecreteMessage(SecretMessage)
Step5: Define EncryptByteStream =EncryptStream(ByteStream)
Step6:DistributeData( EncryptByteStream , RedData,
      GreenData, BlueData);
Step7: RedDataBits = ConvertToBit(RedData);
Step8: GreenDataBits = ConvertToBit(GreenData);
Step9: BlueDataBits = ConvertToBit(BlueData);
```

Fig.3 Secret message manipulation unit

3.1.4 SA Path Construction Unit

Here, the unit starts its task by fetching data bits of one channel, the initial seed of LCG, and SA parameters. Later on, it builds randomly the current vector and computes the distance between the current vector and fetched data bits. In the initial state, the current vector is set as the best one. In ulterior steps, within a several iterations that may be carried out the unit generates a next random vector and compares it with best-stored one based on a fitness function. Finally, it returns the best vector for embedding fetched data and key that will be used in the extraction process. The illustration of each step will be explained in the next subsections.

3.1.4.1 SA Parameters Initialization

SA procedure utilizes two types of parameters in their labor. The first one is called essential parameters and nominated as maximum starting temperature (**Max_Temperature**), minimum terminating temperature (**Absolute_Temperature**) and cooling rate (**CoolingRate**). Additional parameters are the second ones that are required in SA procedure for vector construction and fitness computation; they are assimilated by the host image frame buffer, channel bits and seed of LCG. However, the values of essential parameters may be set as variables that will be entered via a system interface or as fixed ones. In this work, the essential parameters were set as fixed with values 1000, 0.0001 and 0.99 for Max_Temperature, Absolute_Temperature and CoolingRate respectively.

The seed of LCG will be utilized in constructing current and the next solution vector according to the size frame buffer and the number of bits in the sending channel.

3.1.4.2 Initial Vector Construction

In this step, the solver builds a random vector from the locations of a frame buffer. Each location in the frame buffer has a unique index that is used by LCG to build a list of random locations. LCG generates a random numbers vector that has a length equals to the number of bits sent to the SA procedure. Later on, the index of random location will be converted to X and Y coordinates in the frame buffer. As known, each pixel has three bytes(Red, Green, and Blue), then the selection of color value will be based on the color index, so each channel bits has a unique color index to determine which color value is to be selected from the pixel. According to that, the red channel has color index 0, the green channel has color index 1, and the blue channel has color index 2. The values of **x** and **y** are extracted using random locations that were generated by LCG from the host frame buffer as shown in Fig. 4, and they must be in the range of frame buffer height and width.

```
Step1: Set Y = LCG_RandomLocation / Image.Width
Step2: Set X=  LCG_RandomLocation - Image.Width * Y
```

Fig. 4 X and Y extraction from Host Image size

To ensure the uniqueness of random locations that were generated in aim to select x and y coordinates, the LCG method was used for this task. To carry out a set of random numbers via LCG without any duplication, the preconditions of it must be accrued as shown later on. LCG holds and returns the seed to the caller in aim to obviate the utilization of the same seed another time, which may generate the same sequence of random numbers anther time, also the optimal solution seed will be used in the extraction process as a key.

After constructing the current vector, the procedure sets it as the best one and returns a vector with seed to SA procedure. Fig. 5 shows how the initial vector will be constructed from a host image, seed, secret message, and LCG.

Finally, the result of this stage is a vector which is represented as a list such that each location stores index for a selected location, **x** value coordinates; **y** value coordinates, and color value.

```
Step1: Define Vector = new int[MsgLen];
Step2: For i = 0 To #Bits-1
       Seed = RandomVector.GetRandom(Size, Seed);
            Vector[i] = Seed;
        Next For
Step3: Define CurrentVector = new VectorList();
Step4: For i = 0 To Vector.Length-1
        Y = Vector[i] / Image.Width ;
        X=  Vector[i] - Image.Width * Y;
        Color C = Image.GetPixel(x, y);
        If (ColorIndex == 0) Then
             CurrentVector.AddNode(i, x, y, C.R);
        Else
          If (ColorIndex == 1)
            CurrentVector.AddNode(i, x, y, C.G);
        Else
            CurrentVector.AddNode(i, x, y, C.B);
        End IF
      Next For
```

Fig. 5 SA initial vector construction

3.1.4.3 Fitness Function

Here, to determine the fitness value for each constructed vector, the SA solver computes the distance between fetched data bits and color values of the vector. It takes the least significant bit of each color value, builds a vector from those least significant bits, and then computes the distance between data bits vector and least significant bit vector as shown in Fig. 6.

The returned distance will be used to compare between current kept vector and newly generated vector; the minimum one distance updates the higher one. Fitness function updates or keeps vectors according to distance difference between new and best vector as shown in Fig. 7.

As mentioned previously, the seed value that generates the best new vector must be kept for future use in the extraction process. RD is a generated random number that belongs to interval [0, 1]; the value of RD will be utilized to make a comparison process.

```
Step1: Set Distance = 0;
Step2: Define Temp = Vector.Head;
Step3: Set Counter=0;
Step4: While Temp <> null Do
         LSBit = Temp.ColorValue & 1;
          Distance = Distance + (LSBit-MessageData[i],2)^2;
          Counter = Counter + 1;
          Temp = Temp.Next;
        End While
Step5: Return SquareRoot(Distance);
```

Fig. 6 Distance computation between vectors

```
Step1: Set deltaDistance = New Distance - Best Distance
Step2: IF  deltaDistance < 0  Or  e^(- deltaDistance/Max_Temperature) > RD
       Then
          BestVector= NewVector
          Best Distance= New Distance
          SAKey=Seed
       End IF
```

Fig. 7 Fitness function computation

3.1.4.4 Optimal Vector Construction

The process of finding an optimal vector of embedding secret data begins after the initialization of SA parameters and initial vector construction. It accomplishes processing via a set of steps that are carried out within a loop which starts from Max_Temperature and halts at Absolute_Temperature. Inside the loop, it creates a new vector called the next vector, computes the distance of data bits from the next vector, and decides about keeping or updating to the best vector. The optimal vector and extraction key for data are the result when the loop terminates, they are stored in list and returned to the embedding unit. Fig. 8 shows the main steps of building and storing optimal vector.

3.1.5 Embedding Unit

Data of the secret message is divided into three parts in order to distribute them between red, green, and blue channels of RGB image. The embedding unit requires three optimal lists for embedding red, green, and blue data, thus it calls the SA unit three times in aim to accomplish that. After obtaining the three embedding lists, the unit starts their task of embodiment data inside the host image. It uses the least significant bit (LSB)

method to embed bits inside an image in order to reduce the effect of visual appearance modification [3].

Each one of the lists that were returned from SA has a set of locations to store secret message bits. The location in any list can store only one bit at a time, so the unit must manipulate each list distinctly. The embedding unit starts using RedSA list that will be utilized to hide the red portion of the message, after completing RedSA list, it continues with GreenSA and then BlueSA. The pixel in the host image is selected according to the value of X and Y in the current manipulating list. The modification in a selected pixel is done according to the list name (RedSA, GreenSA, and BlueSA), so only one color in pixel must be modified at a time.

```
Step1: Build Initial Vector
Step2: Set Best=Initial Vector
Step3: Compute Best Vector Distance
Step4: While Max_Temperature > Absolute_Temperature Do
        Build Next Vector
        Compute Next Vector Distance
        Set deltaDistance = New Distance - Bestr Distance
        IF (deltaDistance < 0 Or e - deltaDistance/Max_Temperature > RD )
           BestVector= NewVector
           Best Distance= New Distance
           SAKey=Seed
        End IF
         Max_Temperature = Max_Temperature* Cooling_Rate
       End While
Step5: Set Optimal Vector= Best Vector
Step6: Return Optimal Vector and SAKey
```

Fig. 8 Steps of building and storing optimal vector

Target pixels for hiding secret message bits will be selected accruing to their order in each list. Substitution process continues working until hiding all bits in frame buffer. In final stage, the frame buffer data will be transferred to Stego-Image. Fig. 9 shows the main steps of calling the SA procedure, embedding secret data inside frame buffer, and storing results inside Stego-Image.

3.1.4 LCG formula and Conditions

LCG is a well-known random generator that has a set of preconditions to generate a sequence of random numbers over an interval [0, **M**-1] without any duplication until completing the cycle **M** [7]. Those conditions can be summarized as follow:

- C and M have no common factors other than value 1.
- (A-1) is a multiple of every prime number that divides M.
- (A-1) is a multiple of 4 if M Multiple of 4.

```
Step1: Set RedSA= RunSA(HostImage, RedDataBits)
Step2: Set GreenSA= RunSA(HostImage, GreenDataBits)
Step3: Set BlueSA = RunSA(HostImage, BlueDataBits)
Step4: Set Temp = RedSA.Head;
Step5: Set Counter = 0;
Step6: While (Temp <> null) Do
       Color C = HostImage.GetPixel(Temp.X, Temp.Y);
       byte[] Bits = BN.ConvertToBits(C.R);
       Bits[0]= RedBits[i];
       byte NewColor = BN.ConvertToByte(Bits);
       Color NC = Color.FromArgb(NewColor, C.G, C.B);
       HostImage.SetPixel(Temp.X, Temp.Y, NC);
       Temp = Temp.Next;
       Counter=Counter+1
      End While
Step7: Repeat Steps from 4 to 6 for Green and Blue data
     Make sure to modify code to be compatible with each one
Step8: HostImage.Save(StegoImage, ImageFormat.Bmp);
```

Fig. 9 Steps of Hidden process

Fig. 10 shows the general formula of LCG that was used in the constructed system. The initial seed is given by solver. A and C must satisfy the previous preconditions. The value of X_{i+1} and the value of X_i respectively represent next and current random values.

$$X_{i+1} = (AX_i + C) \bmod M,$$

Fig. 10 LCG Formula

3.2 Extraction Process

The extraction process requires a set of keys to extract the secret message from Stego-Image. Those keys are represented by:

- Length of secret message that was embedded,
- RedSAKey, GreenSAKey, and BlueSAKey were utilized in building vectors for red, green, and blue data.
- Key for decrypting secret data
- Extension of the secret message (a type of message)

If the previous information is provided, the extraction process can start and distribute tasks between internal subunits. The length of the message must be known for the system to determine the portion for each channel, also it's used in building a solution vector from Stego-Image. RedSAKey, GreenSAKey, and BlueSAKey are used to determine the initial seed of LCG and staring pixel for each channel inside Stego-Image. After

collecting bits from channels and combining them into bytes, they must be decrypted by using the correct key. Finally, the decrypted bytes will be written to a file that has a correct entered extension. Fig. 11 shows the main steps carried out to extract the message from Stego-Image.

```
Step1:  Get the required keys from the interface.
Step2:  Determine the portion of each channel according to
        message length key.
Step3: RedVector =BuildLCGVector(StegoImage, RedDataSize,
       RedSAKey)
Step4:  GreenVector =BuildLCGVector(StegoImage,
       GreenDataSize, GreenSAKey)
Step5: BlueVector =BuildLCGVector(StegoImage, BlueDataSize,
       BlueSAKey)
Step6: Temp = RedVector
Step7: While Temp <> null Do
          byte[] Bits = new byte[8];
          For j = 7 To 0 Step -1
            Bits[j] = (byte)(Temp.ColorValue & 1);
            Temp = Temp.Next;
          Next For
         byte B = BN.ConvertToByte(Bits);
         WriteToStreamByte(B);
       End While
Step8: Repeat Steps 6 and 7 for Green and Blue data
Step9: Decrypt data of stream byte
Step10: Copy data from stream byte to the secret message file
```

Fig. 11 Steps of the extraction process

4. Results and Analysis

The system was evaluated using a data set of images and binary file of different size with a view to check the security criteria's against detection and modification as shown in Table 1.

Table 1: Evaluation Data Set

Secret Message	Size	Cover Image	Size	Dimension [W x H]
Message1	41Bytes	Jarash	717KB	700 x 350
Message2	508Bytes	Lenna	768KB	512 x 512
Message3	1868Bytes			

According to images that were utilized data set, Jarash and Lenna as host images consist of 73500bytes and 786432bytes respectively, each byte in the host image can store an only bit so that Jarash and Lenna in maximum state store a secret message of length 91875bytes and 98304 respectively. The maximum length of secret message to be hidden inside the host image is computed, as shown in Fig. 12.

Length of Message= (Image Width * Image Height*3)/8

Fig. 12 Equation of Maximum Length for Secret Message

MSE, PSNR, visual appearance, and robustness are the main criteria that were used in evaluating the constructed system [3],[4]. MSE and PSNR are two renowned quality parameters that will be employed to compare

$$MSE= \frac{1}{(W*H)}\sum_{I=1}^{W}\sum_{J=1}^{H}(\text{HostImagePixel(I,J)} - \text{StegoImagePixel(I,J)})^2$$

$$PSNR=20* \log_{10}(\frac{255}{\sqrt{MSE}})$$

- W and H are image width and height respectively

Fig. 13 MSE and PSNR equations

between HostImage and StegoImage. The equations of MSE and PSNR are shown in Fig. 13.

MSE measures the amount of errors or modifications inside the StegoImage that may result from embedding secret message inside the host image to produce the StegoImage. The preference of MSE is measured from lowest value highest one, so minimum MSE is considered as the best one. On the hand, PSNR deals with quality of resulting StegoImage and generates an opposite result to MSE, thus maximum PSNR is classified as eclecticism one. The values of MSE and PSNR are displayed in Table 2 when Massage1, Message2 and Message3 were hidden in Lenna and Jarash as a host images

Table 2: MSE and PSNR at Different cases

Image	Message	Computed MSE	Maximum MSE	PSNR
Jarash	Secret1	0.000249	0.000446	84.1691
Jarash	Secret2	0.002698	0.005529	73.8204
Jarash	Secret3	0.010388	0.020310	67.9655
Lenna	Secret1	0.000221	0.000417	84.6819
Lenna	Secret2	0.002716	0.005168	73.7914
Lenna	Secret3	0.009476	0.018982	68.3646

The value of modification rate is computed from computed MSE over Maximum MSE and it gives the percentage of modification bytes in StegoImage

compared with the original host image. The quality of resulting StegoImage according to value of PSNR is very good and also in terms of computed MSE because the constructed system searches via SA about the path that generates a little noise in image then employees in it in embedding process. Furthermore, the data of secret message, pixels and size of host image play an important role in values of MSE and PSNR as shown in Table 2. Modification rate in utilized data set indicates that on overage 52% of used bytes from the host image were modified, thus it's a good result compared to the size of an image. The rate of MSE for cover image and secret message works in linear fixed form as shown in Fig. 14, thus it can be predicted easily by computing the percentage of secret message bytes inside the cover image.

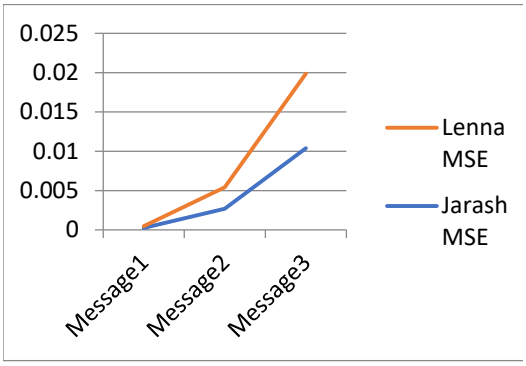

Fig. 14 MSE curve of resulting StegoImages

Visual appearance refers to human view in comparing host image and resulting StegoImage by using the human eyes or the magnifying classes. The resulting StegoImages of constructed system were similar to cover ones; due to SA minimum path that reduced the noise may result from embedding process and also employing LSB substitution process that has a little modification in original data. Fig. 15 shows the cover images and StegoImages that store Message3 with lack of similarity between them in term of visual appearance.

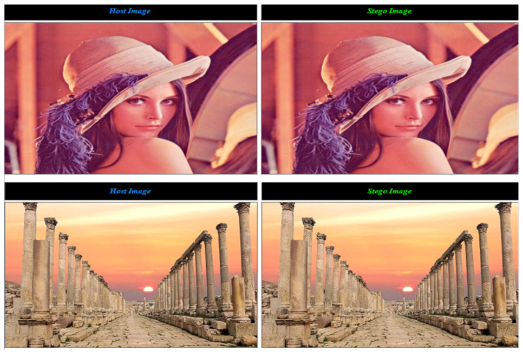

Fig. 15 Visual appearnce of cover and

The robustness against secret message extraction of constructed system resulted from the used keys, employed cryptography and random distribution of data within the image. To extract the secret message, a full knowledge about the utilized keys must be satisfied. Those keys are represented hidden message length, an extension of hidden message, channels (red, green, and blue) builder key, SA parameters, LCG parameters, secret message distribution method, and substitution or embedding method.

Most of image steganography systems encounter a problem with modifications in StegoImage attributes such as size or format, and the constructed system suffers from this problem, so it's not robust against image attributes modification.

5. Conclusion

The idea of combining in a one system SA, LCG, RGB image, secret message distribution between three channels, multi SA vectors, and is considered a unique one. It can be classified as an improvement for sequential and random LSB method. MSE, PSNR, visual appearance, and robustness against extraction are considered the main essential requirements of security for steganography system; the constructed system was satisfied them as shown in the result and analysis section. It was robust against the attack (several keys), effective in embedding data(simple LSB), highly visual appearance(minimum modified paths), use all image pixels(search for an optimal path in all pixels) and simple data extraction by an authorized user(one master key holds all other keys) and suffers from image attributes modification (Resizing or formatting). In the future, the SA may be replaced with another optimization algorithm such as Bat or IWD to check improvement or disimprovement in the new proposed framework compared with the current one.

References

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

[2] A. Singh and H. Singh, "An improved LSB based image steganography technique for RGB images," 2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, 2015, pp. 1-4, doi: 10.1109/ICECCT.2015.7226122. (2015)

[3] Bawaneh, Mohammed J. ,"An Adaptive Virtual Gray Scale Image Steganography Using Simulated Annealing." International Journal of Computer Science and Information Security 14.9 (2016): 612 (2016).

[4] Bawaneh, Mohammed J. "A Preferential Virtual Gray Scale Image Steganography Using Intelligent Water Drop." International Journal of Computer Science and Information Security 14.11 (2016):538 (2016).

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

[6] Bertslmas.D and Tsitsklls.J "Simulated Annealing, statistical science", 8,1, 10-15 (1993).

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

[8] C. Irawan, D. R. I. M. Setiadi, C. A. Sari and E. H. Rachmawanto, "Hiding and securing message on edge areas of image using LSB steganography and OTP encryption," 2017 1st International Conference on Informatics and Computational Sciences (ICICoS), Semarang, 2017, pp. 1-6, doi: 10.1109/ICICOS.2017.8276328 (2017).

[9] E. H. Rachmawanto, K. Prasetyo, C. A. Sari, I. M. S. De Rosal and N. Rijati, "Secured PVD Video Steganography Method based on AES and Linear Congruential Generator," 2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 2018, pp. 163-167, doi: 10.1109/ISRITI.2018.8864466 (2018).

[10] E. J. Kusuma, O. R. Indriani, C. A. Sari, E. H. Rachmawanto and D. R. I. M. Setiadi, "An imperceptible LSB image hiding on edge region using DES encryption," 2017 International Conference on Innovative and Creative Information Technology (ICITech), Salatiga, 2017, pp. 1-6, doi: 10.1109/INNOCIT.2017.8319132 (2017).

[11] Iranpour, M. and Safabakhsh, R., "Reducing the embedding impact in steganography using Hamiltonian paths and writing on wet paper", Multimedia Tools and Applications, 74,17, 6657–6670, doi:10.1007/s11042-014-1921-6 (2015).

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

[13] Mohmmad J. Bawaneh, Atef A. Obeidat. "A Secure Robust Gray Scale Image Steganography Using Image Segmentation", Journal of Information Security(JIS),7,1,152-164 (2016).

[14] N. Akhtar, V. Ahamad and H. Javed, "A compressed LSB steganography method, 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT), Ghaziabad, 2017, pp. 1-7, doi: 10.1109/CIACT.2017.7977371 (2017).

[15] O. S. Sitompul, Z. Situmorang, F. R. Naibaho and E. B. Nababan,, "Steganography with Highly Random Linear Congruential Generator for Security Enhancement," 2018 Third International Conference on Informatics and Computing (ICIC), Palembang, Indonesia, 2018, pp. 1-6, doi: 10.1109/IAC.2018.8780445 (2018).

[16] Parvez, Mohammad Tanvir, and Adnan Abdul-Aziz Gutub. , "RGB intensity based variable-bits image steganography." 2008 IEEE Asia-Pacific Services Computing Conference. IEEE, (2008).

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

[18] Raphel. A and Jacob. E., "DATA HIDING FOR EXTENDED COLOR VISUAL CRYPTOGRAPHY IN GENERAL ACCESS STRUCTURES", International Journal of Emerging Trends in Engineering and Development,2,4,539-544, ISSN 2249-6149 (2014).

[19] Sharma, V.K. and Shrivastava, V. ,"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 (2012).

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

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

[22] Xuefeng Wang, Then Yao and Chang-Tsun Li, "A palette-based image steganographic method using colour quantisation," IEEE International Conference on Image Processing 2005, Genova, 2005, pp. II-1090, doi: 10.1109/ICIP.2005.1530249 (2005).