Image Steganography by LSB Substitution and Optimal Key Permutation Using Genetic Algorithm

Soran Khalid Abdulrahman Al-Dazdaae

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Eastern Mediterranean University
July 2017
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Mustafa Tümer
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Computer Engineering.

Prof. Dr. Işık Aybay
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Computer Engineering.

Asst. Prof. Dr. Cem Ergün
Supervisor

Examining Committee

1. Assoc. Prof. Dr. Önsen Toygar
2. Asst. Prof. Dr. Adnan Acan
3. Asst. Prof. Dr. Cem Ergün

ABSTRACT

Steganography is an important data-hiding technique in which information is secretly passed between a sender and a receiver. In image steganography, a secret-message is embedded into a cover-image to create a stego-image. Furthermore, least significant bit substitution (LSB) is the most common technique used in image steganography. In LSB, either all or some of the pixels in the last four bits are replaced with pixels/a bit of the secret-message. Adding to its security issue, this technique is rather inefficient as it often results in a low-quality stego-image due to the high Mean Square Error (MSE) between the cover and stego-images. Consequently, two methods have been proposed to improve the stego-image's quality: the first method involves using a Genetic Algorithm to uncover the optimal key permutation for embedding, while the second method involves elitism selection. This study adds to these by proposing an innovative method aimed at improving the stego-image's quality.

The method proposed by this study is rooted in both LSB substitution and optimal key permutation by Genetic Algorithm. First, the secret-image and the cover-image are selected, the secret-image is then converted to blocks, which are then encrypted and subsequently shuffled so as to make the embedded secret message meaningless to anyone without the encryption key. The method uses a genetic algorithm to calculate the optimal key permutation to improve the stego-image's quality during LSB substitution. Using the method it proposes, the study found that the stego-image quality was improved substantially following the embedding of the "tiff" secret-image in the following cover-images: Baboon, Lena, Barbara, and Pepper. The study also found that the proposed method significantly reduced the computational complexity in the genetic algorithm.

Keywords: Image Processing, Data Security, Image Steganography, LSB Substitution, Genetic Algorithm, Cross Over, Mutation, Elitism Selection, Tournament Selection.

ÖZ

Steganografi istenilen veriyi gizli bir şekilde göndermeye yarayan veri gizleme ya da saklama tekniğidir. Görüntü steganografisinde, stego resmini elde etmek için gizlenecek resim kapak resmine gömülerek elde edilir. LSB en genel kullanılan görüntü steganografı tekniğidir. Bu teknik her piksel sondan dört bitin ya da bazılarının gizlenecek resmin bitleri ile değiştirilmesi ile olur ve genellikle düşük kaliteli bir stego görüntüsü elde edilir. Sonuç olarak daha kaliteli stego görüntüsü elde etmek için iki yöntem önerilmiştir. İlk yöntemde en doğru permitasyonu ortaya çıkarmak amacıyla eniyileme için tasarlanan (GA) kullanılmıştır, ve genetik algoritmanın içine elitik seçme yöntemi eklenerek algoritmanın yakınsaması hızı artırılmıştır.

Bu çalışmamızda önerilen yöntemler GA destekli LSB yer değiştirme ve en iyi anahtar permutasyonudur. İlk olarak kapak ve gizlenecek resim seçimi yapılır sonra gizli resim küçük bloklara bölünür, saklanacak resim bitleri bu bloklara bölündükden sonra her bir blok şifrelenir, blok sıraları karıştırılır. Bu şekilde elinde anahtar olmayan kişinin gizlenmiş mesajı çözmesi mümkün olmaz. Önerilen yöntem ile stego resmin kalitesinin artırıldığı içine yerleştirilen dört ayrı tift formatlı (gizli) resimde ispatlanmıştır. Bunlar resim işleme deneylerinde sıkça kullanılan, Lena, babuan, barbaro ve pepper resimleridir.

Anahtar kelimeler: Görüntü işleme, Veri güvenliği, Görüntü steganografisi, LSB yer değiştirme, Genetik Algoritma, Çaprazlama, Mutasyon, Elitik seçim, Turnuva seçimi.

DEDICATION

To my Mum and Dad

ACKNOWLEDGMENT

I would like to acknowledge and thank my supervisor Asst. Prof. Dr. Cem Ergün, whose guidance and assistance were pivotal to the success of this thesis. It has been my pleasure working with him, and I recognize his efforts as this projects could not have been done without him.

To my family, my brothers and sisters: Whleed, Tariq, Shaimaa, Alaa and Shelaan, especially my parents, whose financial support and advice enabled me to complete this thesis, I am forever in your debt.

Lastly, I would like to thank all my friends, relative, and colleagues who assisted me in whatever capacity. God bless you all.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

BMP	Microsoft Windows Bitmap File
DSP	Digital Signal Processing
DFT	Discrete Fourier Transformation Technique
DCT	Discrete Cosine Transformation Technique
DWT	Discrete Wavelet Transformation Technique
DES	Data Encryption Standard
DB	Decibel
EA	Evolutionary Algorithm
GIF	Graphical Interchange Format
GA	Genetic Algorithm
GEP	Gene Expression Programming
HAS	Human Analysis Science
IP	Internet Protocol
JPEG	Joint Photographic Experts Group
JQTM	Quantization Table Modification
LSB	Least Significant Bit
LPAP	Local Pixel Adjustment Process
MSE	Mean Square Error
MSB	Moderately Significant Bit
OPAP	Optimal Pixel Adjustment Process
PSNR	Peak Signal to Noise Ratio
PSO	Particle Swarm Optimization Algorithm
PMX	Partial Matched Crossover

TCP	Transmission Control Protocol

Chapter 1

INTRODUCTION

1.1 Overview

Data and information fuel the engines that drive computer communication as well as the global economy in todays's highly competitive and dynamic world. Advances in computer processing power, the development of digital signal processing (DSP), and the Internet have led steganography to go "digital" [1]. In an effort to ensure data security, the concept of data-hiding encouraged researchers to develop creating means of ensuring information does not fall into the wrong hands [2]. This idea is hardly novel and has been used for centuries under different regimes the world over as a way to hide information such that it does not seem to exist [3]. The methods, technologies, and techniques of concealing digital information and covert communication have dramatically increased in the past decade [4].

Electronic data provides a means of easily modifying data so that it can be copied with little to no loss in quality and content. This digital data is easily transferred through computer networks to different locations error-free and usually without any interference. The large-scale distribution of data has increased concern regarding the vulnerability of data to attack and manipulation by other persons [5]. The bulk of modern communication is internet based leading to a desire that such communication be made secret [6]. Consequently, ways of ensuring the safety of secret communication is an important field of research and related techniques increase in volume and sophistication on a daily basis. The digital media utilized for communicating in secret include images, text, audio, and video, which provide superb coverage for hiding information. Data-hiding is the method of embedding secret messages in a medium such that only intended observers should be aware of such messages' existence [8].

Steganography is a widely-used efficient data hiding technique. It is a method whereby secret messages are passed to the knowledge of only the sender and the receiver. It is the practice of invisible communication achieved through concealing the existence of the communicated information by hiding it amongst other information [7]. The secret information cannot be easily extracted from the data-source in which it is embedded without altering said source [10]. Steganography today centers mostly on computers with high speed delivery networks and carrier digital data. A number of techniques have been suggested to embed messages in multimedia objects [11]. Steganography programs permit the user to specify a carrier, an original image or any digital media, that they wish to use in conveying the hidden data [4].

Every steganography system comprises two parts; in the first, the sender inserts the intended message into a cover-object while the receiver extracts the message in the second. The message is converted to a binary message and subsequently embedded in the cover-object (host image) to produce a stego-object similar to the original cover-object. The stego-object is then sent to the receiver via a public channel, who then extracts the binary message. Using a key in the embedding process is optional. If the desired Steganography algorithm uses, it ensures that only a recipient with knowledge of the key can decode the message from the stego-object. Figures 1.1 and 1.2 below show the Embedding and Extraction Algorithms of a Steganography system respectively.

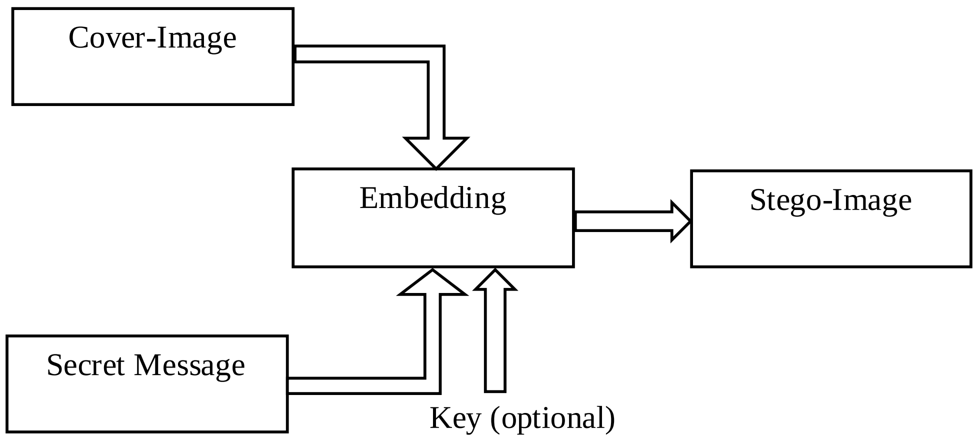


Figure 1.1: Embedding Algorithm

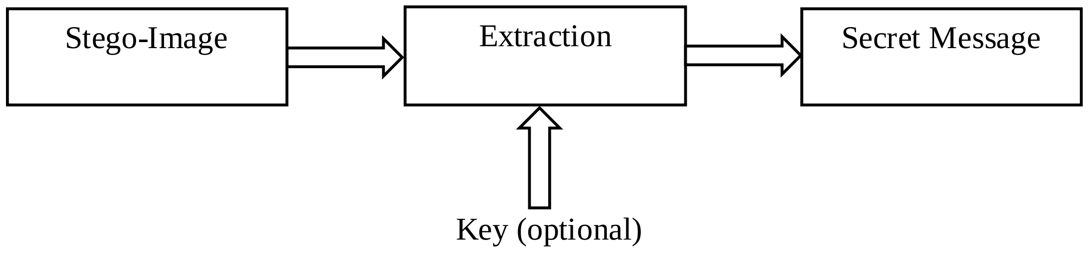


Figure 1.2: Extraction Algorithm of secret-image

1.2 Problem Statement

One problem often faced with Steganography regards the size of the data the user wants to hide in the multimedia file and the quality of the resulting stego-object. Images are one such type of multimedia file. For instance, an attempt to increase the quantity of data hidden in the cover-image, will result in suspicious changes, visible to human eyes, in the original image. This is the problem with which we are concerned in this thesis.

1.3 Thesis Objective

The main objective of this study is exploring how to develop the quality of the stego-object and decrease the complexity of the genetic algorithm. This research will try to create an algorithm grounded in data-hiding techniques for images to increase the amount of hidden data without affecting the cover-image's quality.

1.4 Motivation of the Study

With the rise in the usage of digital communication channels, the protection of signals has become a very important issue of study in the research community. Every second, billions of messages are being transmitted over the internet. Therefore, steganography has become an important issue in the protection of intellectual property and secrecy for the authors of these messages. In this research, the domains of digital steganography, the properties of Human Analysis Science (HAS), the digital representation transmission environments, and its software metric, are discussed and a new method is developed to contribute to the current trend in solving the aforementioned problem. The main aim of this research is to provide a new approach and technique in steganography.

1.5 Structure of Thesis

This thesis adheres to the following structure: following this present chapter, a comprehensive literature review on steganography is offered in chapter 2. Chapter 3 contains the proposed algorithm, as well as the methods used in testing and evaluating the proposed method. Chapter 4 provides the experiment’s results, followed by a discussion section and lastly, Chapter 5 offers a conclusion and future work.

Chapter 2

LITERATURE REVIEW

2.1 Steganography

2.1.1 Historical Context

The term ‘Steganography’ is understood by many researchers to be of Greek origin, and to mean writing that is covered or hidden writing. The term is the result of combining two Greek words: Steganos, meaning covered, and Graphy, meaning writing; Steganography therefore, means Covered Writing [14]. As one of the oldest techniques to be used in hidden communication, there are many examples through history that show a desire to hide either messages or other forms of intelligence from prying eyes. One such example is Mary Queen of Scots, who used both steganography and cryptography to conceal messages. In order to pass messages in and out of her prison, she hide them in the bunghole of a beer barrel. Also, in 5BC, Histaiacus shaved a messenger’s head and wrote a message supporting the mutiny of Aristagoras of Miletus against the King of Persia, sending the messenger when his hair had grown back. This message however, was evidently not time-specific. There have been many more modern examples of stenographic methods such as those used during the Second World War to communicate in secret without enemy interception. Developed by the Nazis, microdots were high magnification microfilm chips; the size of a typewriter period, the dots contained pages of information and drawings amongst others [13].

2.1.2 Current Studies on Steganography

Contemporary steganographic systems employ multimedia objects including text, video, audio, image etc as cover-objects as they are easily transmitted over the internet, as many people do. There are two approaches to Steganography: reversible and irreversible techniques. While the receiver can retrieve both the secret-object and the original cover-object from the stego-object in the reversible technique, in the irreversible technique, the receiver can extract only the secret-object from stego-object, resulting in a distorted original cover-object [19].

The process of hiding information involves the following:

1. Cover-object (hold the secret-object).
2. Secret-object (can be plaintext, digital image or any kind of data).
3. Techniques of Steganography.
4. A stego-key (optional).

Steganography is divided into five branches: 1. Text Steganography 2. Audio Steganography 3. Image Steganography 4. Protocol Steganography 5. Video Steganography [15]. The Figure2.1 below shows categories of Steganography.

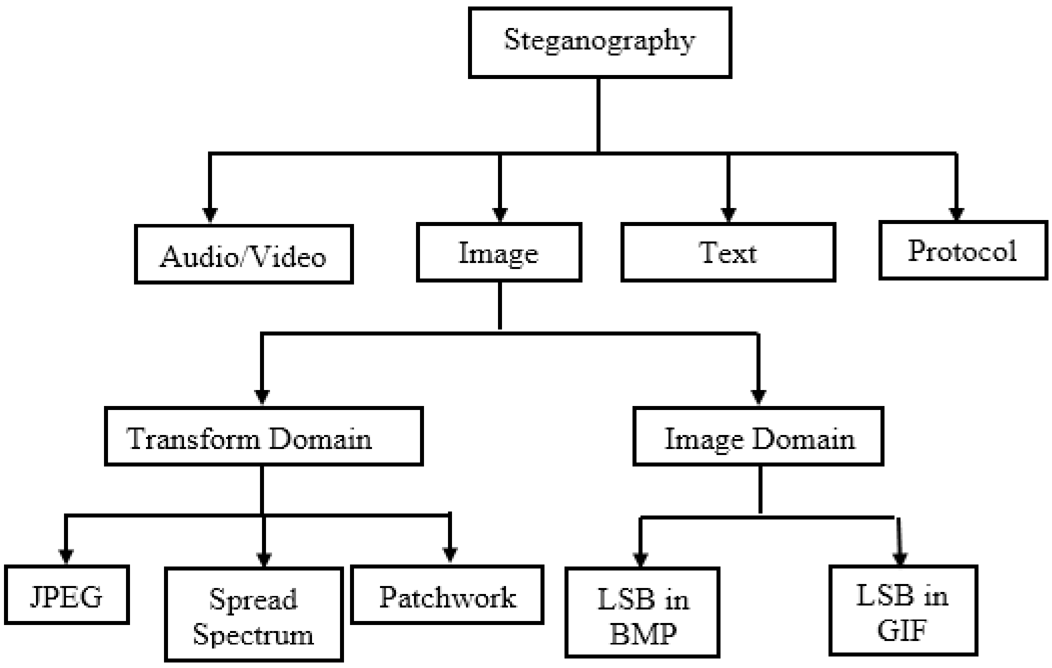


Figure 2.1: Types of Steganography

- **Audio Steganography**: Audio steganography involves embedding a message in a neutral cover speech securely and durably. The methods used in audio steganography include phase coding, LSB coding, spread spectrum, and echo coding.

- **Video Steganography**: This technique involved hiding files of any type in a cover-video file.

- **Image Steganography**: The most commonly used cover-objects for steganography are images, where a secret message is digitally implanted using the secret key.

- **Text Steganography**: This involves embedding the information, as a text file, into the cover-object. Few people use text steganography as the text file has a minute amount of redundant information.

- **Protocol Steganography**: Protocol steganography involves inserting secret information into a network protocol like TCP/IP. Here, the secret message is encoded in the header of a TCP/IP packet I optional/never-used fields [15].

2.1.3 Image Steganography

2.1.3.1 Image Domain

The techniques used in image steganography fall in two categories: image domain and transform domain as shown in Figure 2.1.

The image (spatial) domain technique involves inserting messages into each pixel directly. These “simple systems” involve bit-wise methods, which include bit insertion as well as noise manipulation. The file-formats applicable for this type of steganography are moss less while the technique depends on the image format. The image formats that are the most suitable for image domain steganography are loss less [12].

2.1.3.1.1 Insertion Method for Least Significant Bit (LSB)

LSB insertion is a simple method of embedding secret information in a cover-image. Either all, or some of the bytes in the least significant bit (also called 8th bit) are replaced with a bit/bytes from the secret message [12]. Digital images are split between two variants: 24-bit and 8-bit images. In 24-bit images, each bit of the red, green, and blue color components comprise one pixel, which can be used to store 3 bits. For instance, three pixels of a 24-bit image can be computed as following:

(01011001 01001110 01100101)

(01011001 01001110 01100101)

(01011001 01001110 01100101)

To embed the number 200, which is represented in the binary (11001000), in the LSB of each pixel, the result is shown as:

(0101100**1** 0100111**1** 0110010**0**)

(0101100**0** 0100111**1** 0110010**0**)

(0101100**0** 0100111**0** 01100101)

A single bit of secret information can be encoded in a block with 8-bit image and the stego-image is gotten through the use of an LSB algorithm by inserting the secret-image into the cover-image. A reverse process is used to extract the hidden message from the stego-image.

2.1.3.2 Transform Domain

In the transform (also called frequency) domain techniques, images are first transformed before the message is inserted into the image. The aim of these methods is to embed the secret message in more areas of the cover-image in order to make it more durable [12, 20]. Some of transform domain techniques do not rely on the format of the image and might outrun lossy and lossless format conversions.

Techniques of transform domain come in the following variants [27]:

1. Discrete Fourier Transformation technique (known as DFT).
2. Discrete Cosine Transformation technique (known as DCT).
3. Discrete Wavelet Transformation technique (known as DWT).

2.2 Cryptography

Cryptography is a technique used to generate cypher-text by encrypting plain-text. Data that is easily understood without special procedures in this technique is known as plaintext. The original text is therefore referred to as plaintext and the text resulting from the encryption of a plaintext to guarantee secrecy and information authenticity, as cipher-text. Cipher-text allows the information to be transmitted through insecure networks while preventing anyone but the intended recipient from reading it. Contemporarily, cryptographic techniques are used in either of two scenarios: secret key cryptography and public key cryptography [18].

2.2.1 Secret Key Cryptography

The secret key (also called symmetric key) cryptography scheme involves the use of one key the encryption and decryption processes, of which Data Encryption Standard (DES) is the most widely used technique. Quick recitation, expedited authenticity checks of key recipients, and the use of the same key used during encryption to obtain a plaintext are the advantages of this technique whereas the disadvantages of this technique include the fact that the key sharing mode may be vulnerable to attack. By getting the secret key, any unauthorized person can easily gain access to all the information as this kind of technique does not provide digital signatures that cannot be rejected. The figure 2.2 shows the mechanism of secret (symmetric) key cryptography.

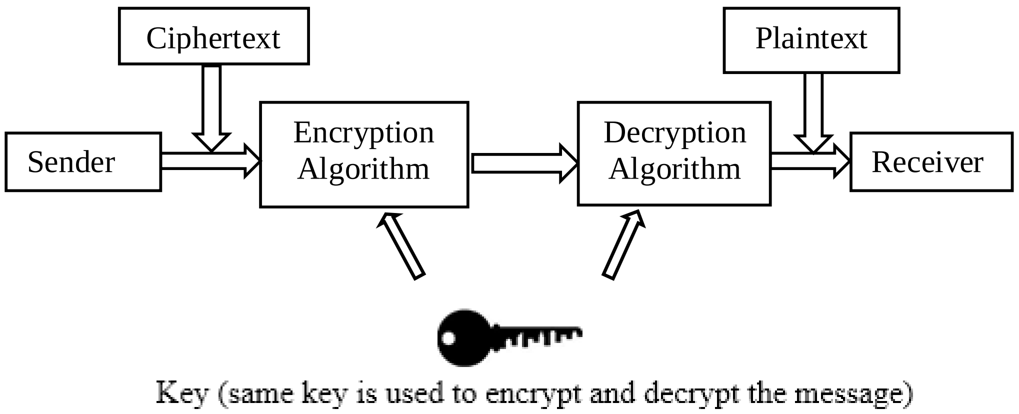


Figure 2.2: Secret (Symmetric) Key Cryptography

2.2.2 Public Key Cryptography

Public key (asymmetric key) cryptography scheme uses two keys: the public key and private key. While the former is used in the encryption process, the latter is used for decryption. Utilizing two keys ensures security-strength as opposed to just one key where anyone with said key can decrypt the message. Figure 2.3 below shows the mechanism of public key cryptography [1, 17, 18].

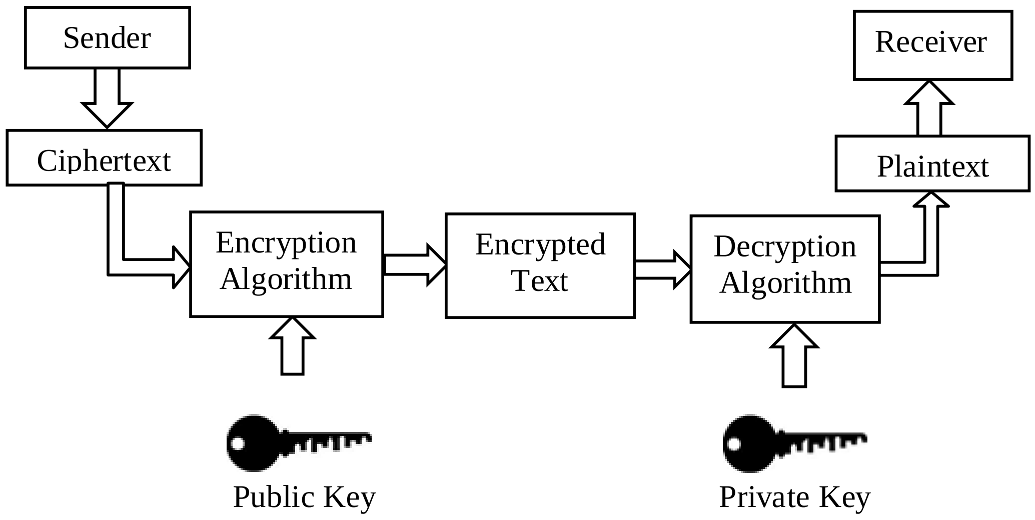


Figure 2.3: Public (Symmetric) Key Cryptography

2.3 Genetic Algorithms (GA)

This is an optimization algorithm that copies Darwinian natural selection. It is employed in solving optimization problems, such as that of combination, where objective equations are not easily computed. The genetic algorithm is a part of the larger evolutionary algorithm, which uses natural selection methods including cross-over, inheritance, mutation etc. to produce suboptimal solutions for any search problems.

2.3.1 History of GA

First, Allan Turing posited the notion of a "learning machine" to duplicate the process of species' evolution back in 1951 [21]. In 1954, Nils Aall Barricelli began computing and simulating evolution over the course of his work on a computer at Princeton's Institute of Advance Study (New Jersey, USA). Alex Fraser, an Australian geneticist, conducted a number of studies covering the "artificial selection of organism with multiple loci controlling measurable traits" back in 1957 and the practice of biologists simulating evolution became widespread by 1960 based on Fraser's work. It is worth mention that the majority of elements in present genetic algorithms may be traced back to Fraser's pioneer simulation. In the 1960s, Hans-Joachin Bremermman proposed "a population of solutions which go over crossover and alteration to solve optimization problems"; embodying most of the characteristics of contemporary genetic algorithms [22]. Other innovators in the field are John Holland, Richard Friedberg, George Friedman and Michael Conrad. Despite the fact that Barricelli is given credit for simulating a simple game of evolution, it was the work of Ingo Rechenberg and Hans-paul Schwefel in the 60's and 70's that adapted artificial evolution as a method of optimization.

2.3.2 GA Steps

We may begin to develop the solutions to the research problems using the steps outlined below:

1. **Initialization:** Although the first population of the candidate solution is generally randomly generated over the search space, it is easy to incorporate domain-specific knowledge or other information.
2. **Evaluation:** The fitness values of each candidate solution are calculated either when the population is generated or when an offspring has been created.
3. **Selection:** A survival-of-the-fittest scenario is imposed on the candidate solutions through selection, which creates duplicates of solutions having fitness values that are above-average. The motivation for selection is a preference for better solutions over worse alternatives and a number of selection procedures seek to materialize this preference, including ranking selection, stochastic universal selection, tournament selection, and roulette selection, partly covered in the proceeding section.
4. **Crossover:** This entails the combination of two or more parent solutions for the creation of novel, possibly superior solutions (offspring). There are a number of ways to accomplish this, some of which are covered in the next section, but their performance is dependent on an adequately-designed mechanism for crossover. The recombined offspring soul combine the traits of its parents rather than be identical to any of them [23].
5. **Mutation:** whereas crossover combines traits from two or more parent sources, mutation involves randomly altering one solution. There are also many variants of this method, but they all involve at least one change in an individual's

trait(s). Put differently, mutation involves a random incursion into the realm of candidate solutions.

6. **Replacement:** The offspring produced using either crossover, selection, or mutation replace the original population. Some of the replacement techniques used in GAs include generation-wise replacement, elitist replacement, and steady-state replacement.
7. Repeat steps 2–6 until the emergence of a terminating condition [23].

From Figure 2.4, it is clear that the process of evolution starts with a randomly generated initial population (individuals or solutions), where individual groups of members in a repetition are called a generation. Each solution in a repetition is evaluated based on how well it resolves the particular problem in question and the evaluation process is known as ‘fitness evaluation’. This fitness evaluation and superior solutions are forwarded to the next generation while new solutions are created via the Crossover and alteration of individuals thus ensuring that new individuals and traits are introduced at every iteration. This process usually ends either when the maximum number of repetitions has been reached, an acceptable fitness level is attained, or the individuals do not improve over generations.

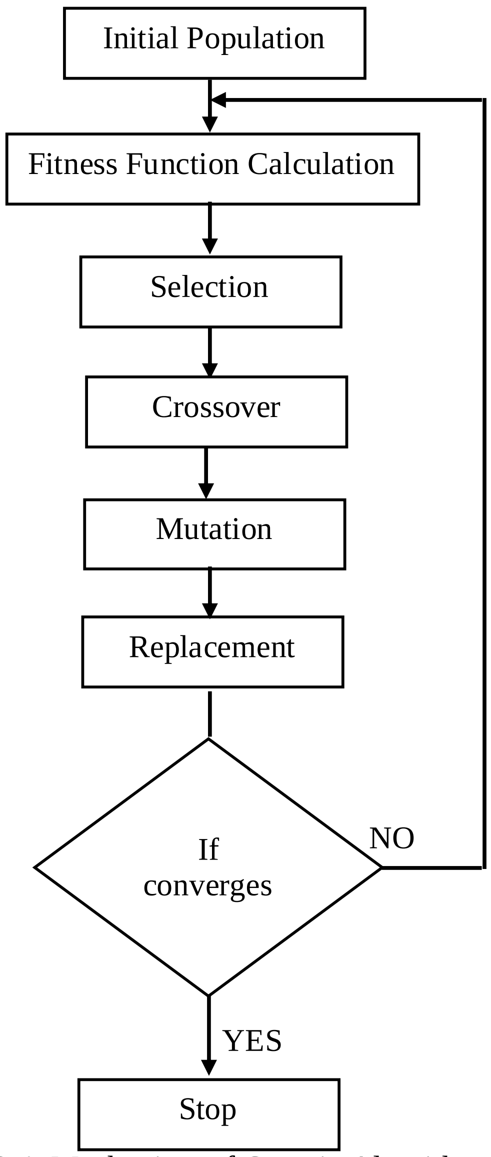


Figure 2.4: Mechanism of Genetic Algorithm

2.3.3 A Basic GA

Algorithm 2.5: A Basic GA [1]

```
INPUT {pop = size of the population, $P_c$ = Crossover Probability, $P_u$ = Mutation
  Probability, Nbits = number of bits per individual, $f()$ = fitness function }.
1. [Begin] Using randomly generated individuals with Nbits alleles, create a
   population size (pop).
2. [Fitness] Using $f()$, calculate the suitability of individuals contained in the
   population.
3. [New population] Until the requisite no. of individuals has been reached,
   new populations are generated as follows:
       1. [Selection] at least two individuals from the population are chosen
          as as parents.
       2. [Crossover] apply crossover to parents with the probability $P_c$ to
          generate new offspring.
       3. [Mutation] using the probability $P_u$, modify the traits of an
          individual, thus resulting in a new individual.
       4. [Accepting] based on certain measurements, new individuals are
          either accepted into, or rejected from the population.
4. [Replace] For the next generation, use the newly generated population.
5. [Test] Test for termination condition.
6. [Loop] Go to 2.
```

2.3.4 Elements of GA

2.3.4.1 Parent Selection

Traits from different individuals are combined to produce new and better individuals in GA. This therefore necessitates a technique through which one might determine what individuals to use as parents for a new generation. The more common selection procedures are:

- **Roulette Wheel:** Here, each individual is given the opportunity to be a parent corresponding to how good a fit it is. A number is selected; the individual with a fitness value greater than the next individual but less than the drawn value is chosen as a parent. The process is repeated until the number of parents required is gotten meaning that individuals with higher fitness values would be dominant as they occupy more space on the roulette wheel and thus are more likely to be selected.

- **Rank Based:** The roulette wheel method is biased towards individuals with higher fitness levels where there are substantial differences between individuals' fitness levels. In order to avoid this, a rank based method, where the individuals are ranked according to their fitness values, with each fitness rank occupying a position on the roulette wheel so as to allow even weaker individuals a chance at being selected as parents, is more applicable.

- **Tournament:** In this method, a tour size [22], with a minimum of two and a maximum of all individuals in the population, is deduced prior to the selection process. Following this is a random selection of the population, a subset of which, equivalent to the size of the tour, is used as a mating pool. The most fitting individuals in the mating pool are then chosen to be parents.

2.3.4.2 Genetic Operators

2.3.4.2.1 Crossover

In a crossover, genes are selected from different parents to generate fresh children. The alleles gotten from the chosen parents carry the information necessary for the children to resolve an issue. Consequently, children gotten from good parents might receive either all or some of their good traits and their defects.

2.3.4.2.2 Mutation

As the crossover generates children based on information sourced from numerous parents, they often inherit traits contained in the population. So, mutation alleles are altered in a single individual to ensure new traits find their way into the population pool.

2.4 Related Works

Wang et al. [25] proposed two methods to hide each bit of the secret message into the fifth bit of each pixel in the cover image. Using genetic algorithm to obtain the best substitution matrix for embedding the secret data, the second method uses local pixel adjustment process (LPAP) to increase the quality of the stego-image. This method used moderately significant bits (MSB) to hide the important data instead of optimal substitution algorithm and local pixel adjustment, and exemplified a good substitutional choice for storing and transmitting important data.

Chang et al. [24] suggested two hybrid LSB substitution methods to hide the secret data in a cover image: firstly, mixing the optimal pixel adjustment process (OPAP) and optimal LSB substitution to increase the quality of the stego-image, or, using the worst LSB substitution and OPAP. Results demonstrated that the hybrid LSB substitution methods give better results compared to optimal LSB substitution, simple LSB substitution, worst LSB substitution, and OPAP because it has the highest peak signal-to-noise ratio (PSNR).

Hegde et al. [15] suggested a method using LSB and a seemingly random encoding technique to hide the secret message with the master file (carrier file) and sent to the right user. This method selects the pixels of the cover-image randomly when hiding the message (can be plaintext, digital image, audio, video) and uses random key (stego-key) between the sender, to embed the message, and the receiver to extract the message, and makes it very hard for unauthorized users to obtain the embedded message. The result shows that the pseudo random encoding has the highest value in terms of capacity, imperceptibility, and robustness compared with the LSB algorithm. In addition, the PSNR of the proposed algorithm (pseudo random method) is higher than that of the LSB method.

Marghny et al. [26] presented a data-hiding method using simple LSB to improve the capacity of embedding the important data from the secret message and stego-image quality using the LSB technique, which the secret-message is embedded by replacing the rightmost k of the LSBs of the host image's pixels directly with bits of the secret message. The host image is split into two sections: the first section of the host image embeds the secret data of the secret message and applies a change to the bit values that have the secret bits gained by the simple LSB substitution, while the second section of the host image is used to point out which change is applied to every pixel that exists in the first section. The result demonstrates that the proposed method gives a quality stego-image and embeds larger quantities of data.

Wang et al. [34] developed a method to hide secret messages with a large size based on least significant bit substitution to stop unauthorized access of the important data and get improved results. The proposed algorithm used a simple least significant bit (LSB) substitution and optimal least significant bit (LSB) substitution to increase the system performance. This method proved the worst case of embedding results using optimal least significant bit (LSB) substitution. To get the optimal embedding result, each possible substitution should be evaluated, which requires a long computation time. A genetic algorithm can solve this problem and is used to find the best solution to said problem. The genetic algorithm tries to embed the secret message in the rightmost k least significant bits (LSBs) when k is large. The experiment's result shows that using a genetic algorithm requires a lesser computation time compared to simple least significant bit (LSB) substitution.

Chang et al. [35] proposed a method to hide the secret message into the cover image using a dynamic programming strategy to increase the quality of the stego-image and find the optimal embedding result. The experimental result shows that the proposed algorithm obtained a better result compared with simple LSB substitution and optimal LSB substitution, requires less computation time and resulted in an optimal solution.

Chan et al. [39] proposed to embed the secret data into the cover-image using simple LSB substitution, which embeds the bits of the secret message into the least significant bits in the cover-image. The stego-image is obtained, as well as the optimal pixel adjustment process (OPAP), which uses three intervals to measure the error between each pixel of the cover-image, stego-image (obtained by simple LSB), and the stego-image (obtained by applying OPAP). The result shows that applying the OPAP method is better than the simple LSB and optimal LSB because the PSNR value after applying OPAP is high, meaning that the image quality of the stego-image is improved.

Marghny et al. [36] proposed an algorithm to embed the secret message in the cover-image by replacing the least significant bits (LSB) of the cover-image with the secret image bits giving the embedded result (stego-image). This method contains two phases: firstly, using a data-hiding LSB technique with a key permutation method. Secondly, they suggested a novel approach for uncovering the optimal key permutation using gene expression programming (GEP); GEP combines the advantages of both genetic algorithm and genetic programming. The experiment’s result shows that the proposed algorithm increases the quality of the image, and provides high capacity and a low computation time leading to increases in the system security.

Wang et al. [37] proposed a stenographic method using JPEG and practical swarm optimization algorithm (PSO) to increase the stego-image quality. The main goal of this method is to protect the important data from unauthorized users in the communication channels. The two most important gauges in evaluating a stenographic method are the amount of information contained in the secret message and the quality of the stego-image (embedded result). This method embeds large quantities of the secret data while keeping the quality of the stego-image agreeable, which is derived from an optimal substitution matrix using practical swarm optimization algorithm (PSO) to convert the secret data then hiding said data in the host image out of a modified JPEG quantization table. The experimental results show that the proposed method can handle large quantities of data and provide a better stego-image quality than the JQTM method.

Attri et al. [38] presented a new technique based upon optimization in image steganography using a genetic algorithm. The simple LSB technique is easy to bypass by unauthorized users and is less robust. The proposed algorithm used a genetic algorithm to obtain more robustness and increase the time complexity as it provides an approximate optimal solution of the problem. The genetic algorithm in steganography uses four steps in this technique: in the first step, the bits of the secret message replace the bits of the cover-image using simple substitution. In the second step, a genetic algorithm is used in decreasing the amount of error and increasing transparency. The third step involves using a quality controller – the new pixel will be accepted if the original pixel value and the new pixel value are different, else it will be unacceptable and the original cover will be used in reconstructing the new image instead of that. In the last step, the stego-image is created. The result shows that the robustness of the substitution technique is increased while maintaining its data-hiding capacity.

Chapter 3

PROPOSED METHODS

The methods suggested in order to better the stego-image's quality have been twofold. In the first, several experiments are conducted so as to uncover the best values for the embedding process in the genetic algorithm. The second method involves the addition of an elitism selection. Mutation and cross-over have been known to result in offspring that are considerably weaker than their respective parents, leading to the loss of good potential candidates. While it is possible that the Evaluation Algorithm can find these lost traits in a later generation, there is no guarantee that this will be the case for all cases. Elitism, which involves including a tiny percentage of the previous generation's best candidates in the next, is used to mitigate such losses. Additionally, elitism may also significantly boost performance as it prevents the Evaluation Algorithm from having to search for and find lost partial solutions. As a final point, candidate solutions carried over to the next generation unaltered through elitism can become parents in the new generation when breeding begins.

3.1 Embedding Algorithm

The proceeding section contains a detailed introduction of this study's proposed method. LSB substitution and optimal key permutation form the basis of this method, while a genetic algorithm is used to select the most suitable key.

As input, the proposed method need the cover-image, secret image, stego-key and, key permutation to do the embedding. This gives a stego-image as output which is send to the receiver.

3.1.1 Encryption Method

Following is a description of the steps involved in the encryption algorithm.

Algorithm 3.1: Encryption Algorithm

---

1. Produce a key of length L (permutation of integer number 1 to L) L $= \frac{N}{n * n}$

   Where N is the number of pixels in the secret image (S).
2. Split the secret image into n by n blocks.
3. Number blocks starting from the top left corner to the bottom right one.
4. Change the places of the blocks using key to get encrypted secret image (S').

Assume that, the sender wants to send the stego-image to the receiver. Both the sender and the receiver share the same secret key (Sender and the receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure). For encryption, the sender encrypts the stego-image with the shared secret key. For decryption, the receiver decrypts the stego-image with his (the same) secret key.

For example, the secret-image's size is 12×12 pixels (N= 144), the encryption key length L= $\frac{12\times12}{4\times4}$ = $\frac{144}{16} =$ 9 blocks. Then we split the secret-image to 4×4=16 (when n=4) blocks and start numbering the blocks accordingly as shown in Figure 3.1. According to the encryption key which is generated randomly, assume that:

$$\text{Encryption key} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 3, & 5, & 2, & 7, & 9, & 1, & 6, & 4, & 8 \end{pmatrix}$$

means that put the first block of the original secret-image to the third block into the encrypted secret-image and the second block of the secret-image to the fifth block into encrypted secret-image and so on to get the encrypted secret-image (S').

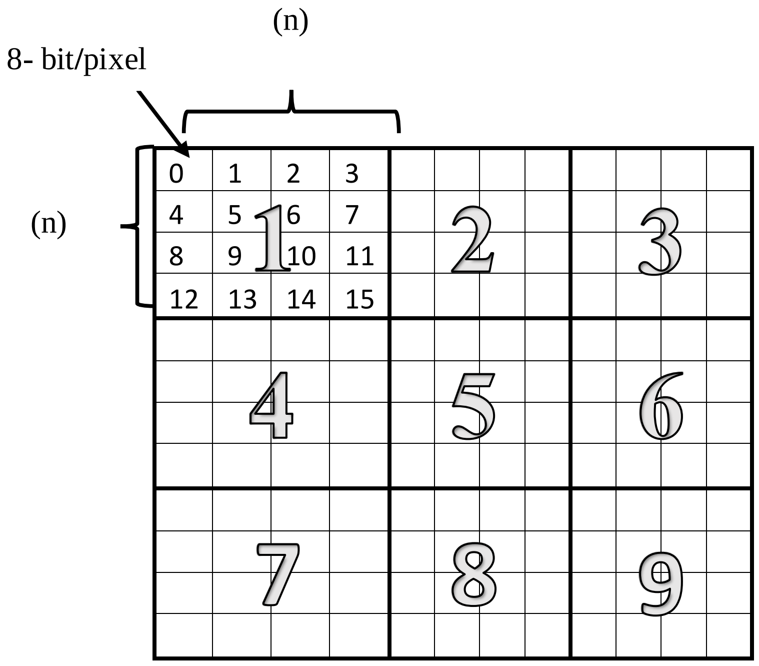


Figure 3.1: Original Secret-Image (S) (12×12 pixels)

From Figure 3.2, the encryption method is used to make the secret-image meaningless for the third party attackers (hackers). Taking the encryption key randomly, we set the seed value; if the second party (recipient) has the encryption key (key= [3, 5, 2, 7, 9, 1, 6, 4, 8]) when k = 4, then the recipient can extract the original secret-image.

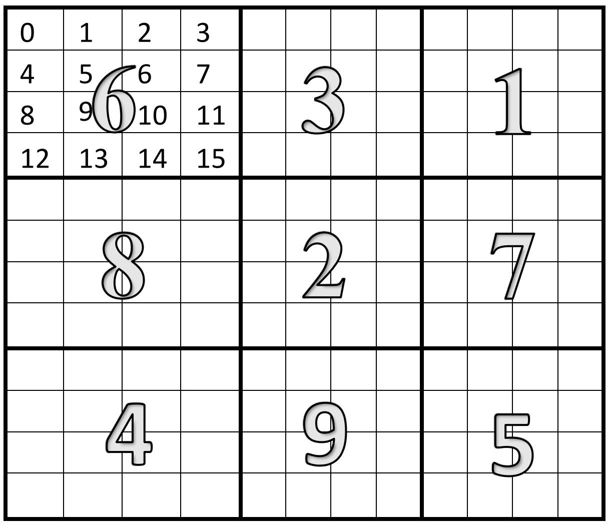


Figure 3.2: Encrypted Secret-Image (S') (12×12 pixels)

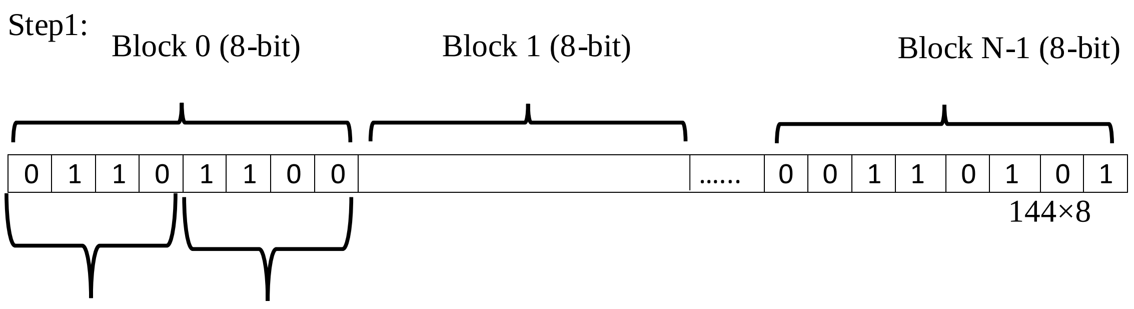
3.1.2 Embedding (k-bit blocks)

The data embedding process' steps are explained below:

Algorithm 3.2: Embedding k-bit blocks Algorithm

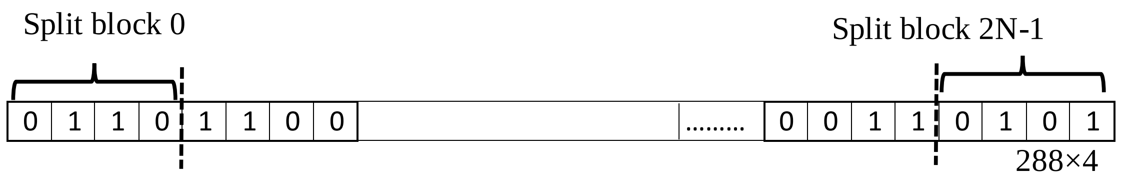
1. Convert encrypted secret image (S') into a row vector of size N × 8.
2. Reshape the vector into a k-bit image (size identical to the cover image). Each pixel is k-bit now (S'_k).
3. Apply key permutation to bit blocks (S''_k) using GA.
4. Replace the k-LSBs in the cover image with S''_k pixels.
5. The stego-image is produced.

When dealing with the encrypted secret-image, each pixel in the encrypted secret-image has 8-bit blocks (S') as shown in Figure 3.2. Firstly, we should split the encrypted secret-image to 4-bit blocks (S'_k) (when k=4) because we have k-bit spaces into cover-image. Then, the encrypted secret-image using optimal key permutation (S''$_k$) is obtained (generated form GA) when (i) is all possibilities of k-bit block of the encrypted secret-image (S'$_k$), (ii) is the optimal key permutation obtained from GA (In decimal), and (iii) is the optimal key permutation obtained from GA (In binary). After that, replacing the bit-blocks of the encrypted secret-image (S''$_k$) into k-LSBs of the cover-image. For example, the first pixel of the cover-image is 25 (in binary; 00011001) then we replace the 4-bit blocks of the encrypted secret-image (0101) into k-LSBs of the cover-image and so on for all pixels of cover-image as shown in step 4 Figure 3.3. Finally, after replacing, each pixel of the stego-image has the amount of information of the encrypted secret-image as shown in step 5 Figure 3.3.



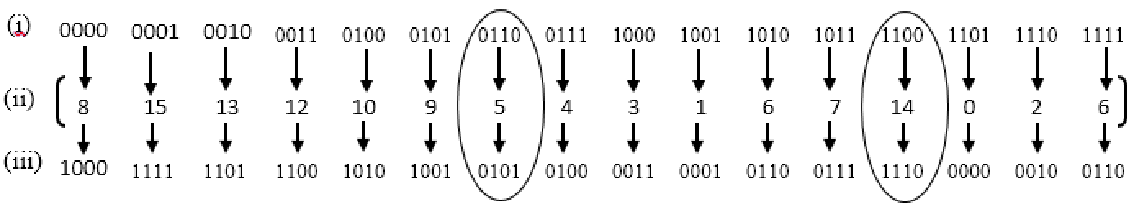
(a) Convert encrypted secret-image (S') to row vector

Step 2:



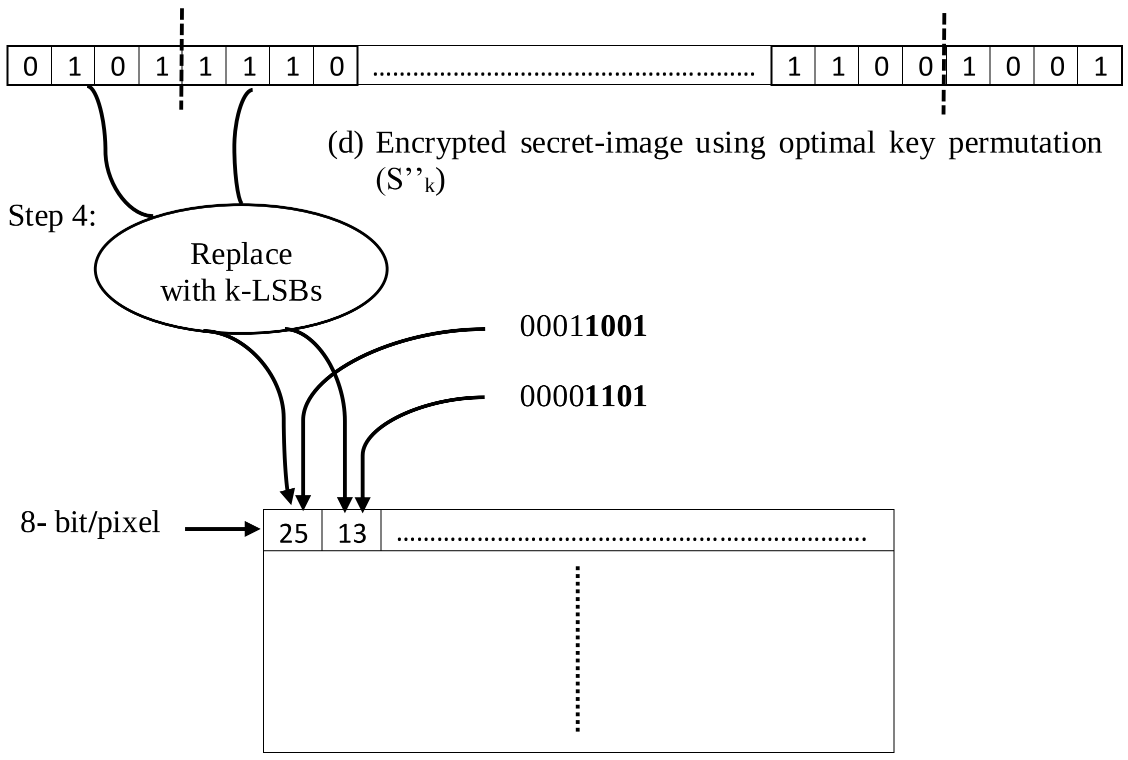
(b) Split each pixel of encrypted secret-image to k-bit blocks (S'$_k$)

Step 3: Assume that the optimal key permutation is obtained from GA is follow:


(i) 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
(ii) 8 15 13 12 10 9 5 4 3 1 6 7 14 0 2 6
(iii) 1000 1111 1101 1100 1010 1001 0101 0100 0011 0001 0110 0111 1110 0000 0010 0110

(c) key permutation obtained from GA

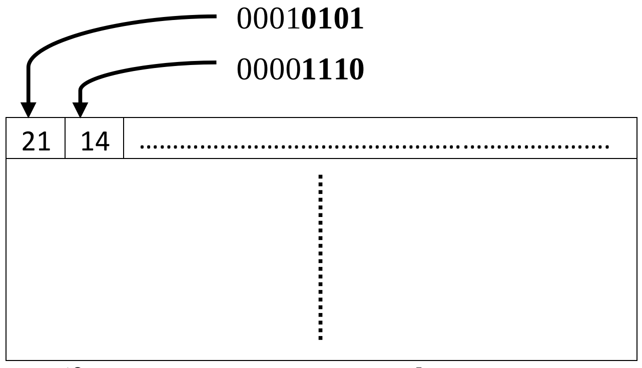
Step 4:



(d) Encrypted secret-image using optimal key permutation (S’’$_k$)

(e) Cover-image 12×24 pixel

Step 5: the stego-image.



(f) stego-image 12×24 pixel

Figure 3.3: An Example of Embedding Encrypted Secret-Image into Cover Image

3.1.2.1 Bit permutation on S’$_k$

The bit permutation in the proposed method is described below:

Algorithm 3.3: Bit Permutation Algorithm

```
1. P_k is a vector of length $2^k$ consisted of random permutation of numbers
   between {0 to $2^k$-1}.
2. For each original pixel value $v = 0 : 2^k-1$
                $v(i) \leftarrow P_k(i)$      when $i = 1 : 2^k$
3. The new image is $S''_k$.
```

According to the key permutation, after we change the 4-bit block. Assume that the key permutation (generated from GA) (when k = 4) is obtained as shown in step 3 (i), (ii), and (iii) Figure 3.3, after searching the whole image, all block number positions are matched by using GA in encrypted secret-image (S'_k). For example, if k-bit block (0000) is considered for the first search, according to GA (1000) is found as the best match for first 4-bit block (which yields lowest MSE value) then they are replaced in step 3 (iii) Figure 3.3, and search the whole image again and find matching block number via MSE, if k-bit block (0001) is considered for the second search, according to GA (1111) is found as the best match for second 4-bit block, then they are replaced in step 3 (iii) Figure 3.3 and so on for other pattern.

3.1.2.1.1 Genetic Algorithm

This data hiding technique performs sub-optimally when the cover-image's k-LSBs are the selected location for embedding. This is particularly true when **k** > 3 as the amount of possible key permutation increases exponentially and in correlation with increases in **k** increases. To illustrate this, if we assume that k is 4, the aggregate number of prospective key permutations for embedding the data is 16! (approximately 20,000 billion key permutations). The simplest way to find the best possible embedding result is to compute the PSNR for every substitution and choose the substitution with the highest PSNR value. Unfortunately, the process of calculating the PSNR of each substitution is time-consuming and impractical. Instead, a genetic algorithm, which randomizes the search process, is often used to solve the optimization problem. Every potential solution matches an individual in the GA that is represented by a chromosome comprised of various genes. The fitness function, which is an objective function, allows us determine the quality of each individual chromosome. The GA process begins by defining an initial population within the first generation. This population is chosen to undergo crossover and mutation processes to produce offspring for subsequent generations. The quality of each resulting offspring is then determined using the fitness function and the offspring with the highest qualities are allowed to survive and shape the next generation. The entire process is consistently repeated until a certain amount of iterations has been reached, or a predetermined condition has been satisfied. In our study, in a GA with 2^k genes, an individual G' is described using a key permutation as:

$$G' = g_0 g_1 \ldots\ldots g_{2^k-1}$$

Where $g_0$is the first element of the key, $g_1$the second etc.

For instance, the chromosome length is $2^4 = 16$ if we take k as 4, random chromosome can be generated to obtain sample population as:

$$\left.\begin{array}{c} G'_1 = 13\ 15\ 3\ 5\ 8\ 0\ 10\ 11\ 6\ 12\ 2\ 7\ 1\ 9\ 14\ 4 \\ G'_2 = 5\ 7\ 9\ 4\ 12\ 1\ 14\ 6\ 3\ 10\ 2\ 8\ 11\ 13\ 0\ 15 \\ \vdots \\ G'_{50} = 15\ 2\ 3\ 5\ 8\ 11\ 10\ 9\ 7\ 4\ 13\ 0\ 1\ 14\ 12\ 4\ 6 \end{array}\right\} \text{Population}$$

One method used in selecting an individual from the population in a GA is tournament selection. Here, multiple 'tournaments' are carried out between a random selection of individuals (chromosomes) taken from the population. First the winners of each tournament are chosen for the cross-over operation to find most fitted member. Selection-related pressures are easily mitigate by altering the tournament size as weak individuals have a smaller chance of selection in larger tournament sizes [9].

Algorithm 3.4: Tournament Selection Algorithm

```
1. Tournament size ($t_s = t_r \times P_s$)
   Where $t_r$ : tournament rate, and $P_s$ : population size.
2. Select $t_s$ chromosomes from the old population randomly.
3. Evaluate then cost (fitness) function (MSE).
4. Sort the chromosomes according to cost function (which should be
   minimized).
5. Select the best two of $t_s$ as a parents.
```

The genetic operators are defined as follows:

- **Partially matched crossover (PMX):** here, offspring are produced using two crossover points in the parents' chromosomes selected at random [29]. For example, if the number of embedded data for each pixel is equal to 4 (k=4), the chromosome's length is equal to 2^k ($2^4 = 16$), and the genes are in range from 0 to 15.

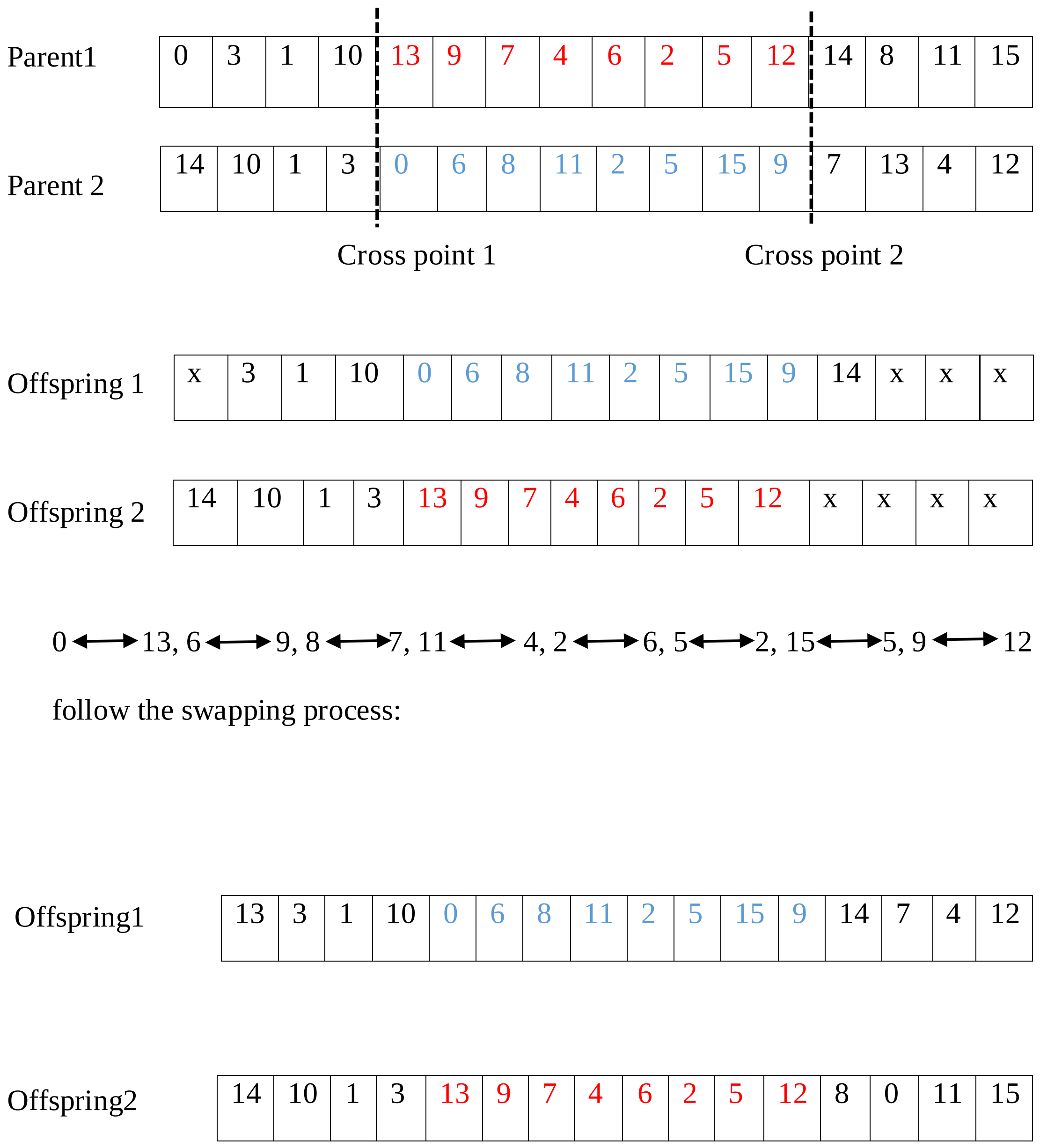


Figure 3.4: Example of Partial Matched Crossover (PMX)

Algorithm 3.5: Partially matched crossover (PMX) Algorithm

---

1. Select two random cut points for parent 1 and parent 2 [31].
2. Copy the segment between the cut points from parent 2 to offspring 1.

3. Starting from the first gene, copy the elements of parent 1 that do not exist in offspring 1.
4. For the duplicates, follow the swapping process of the second stage and fill the remaining places.
5. Apply the same process to produce offspring 2.

- **Mutation operation:** After getting offspring1 and offspring2, the are each mutated in the next step as follows:

Algorithm 3.6: Mutation Algorithm

```
For each gene in the chromosome:
1. Randomly select a number between 0 and 1 [32].
2. If the random number < mutation probability, do
3.  Select two indexes randomly (Ind1 and Indx2), suppose that Indx1< Indx2,
    otherwise flip them.
4. Temporary = offspring (Indx1).
5. Shift the other genes in range of Indx1:Indx2 to the left.
6. Insert temporary into offspring (Indx2).
```

For example, if the number of embedded data for each pixel is equal to 4 (k=4) and the chromosome’s length is denoted as 2^k ($2^4 = 16$), the resulting genes are in range of 0 to 15.

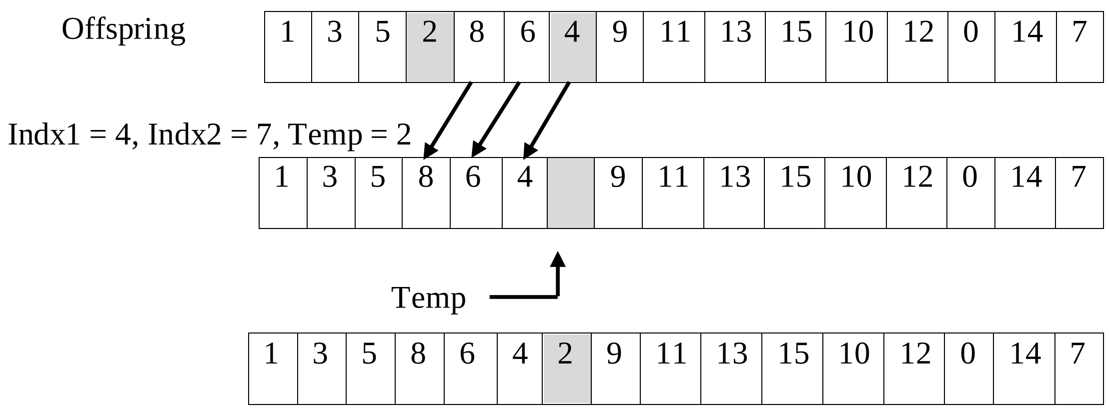


Figure 3.5: Figure 3.5: Example of Mutation

The Figure 3.6 shows a flowchart for the proposed algorithm. As can be seen, the process begins first with (Preprocessing steps (part a) for Steganography) the selection of the cover-image and the intended secret-image. The secret image is then converted into blocks, which are subsequently encrypted using the chosen encryption key. The genetic algorithm (part b) is used to find the optimal key for key permutation after which the stego-image is produced.

In Figure 3.6, the arrows shows that GA process (part b) and k-bit permutation process (part c) are connected each other. Here, first produce initial key population, which is followed by the production of a stego-image for each of the chromosomes. Then these chromosomes are used to produce a k-bit image, the blocks of which are permuted based on the key permutation followed by the embedding process of the k-LSB bits to have a stego-image, which is then inputted back into the genetic algorithm where the fitness function of this stego-image is calculated, followed by the processes of selection, crossover and mutation. These processes are repeated to find the optimal key value, according to converging criteria (will be explained in chapter 4). If the convergence is satisfactory (Yes), the final stego-image is produced. However, if the convergence is not satisfactory (No), the k-bit permutation is repeated, starting with the production of a k-bit image.

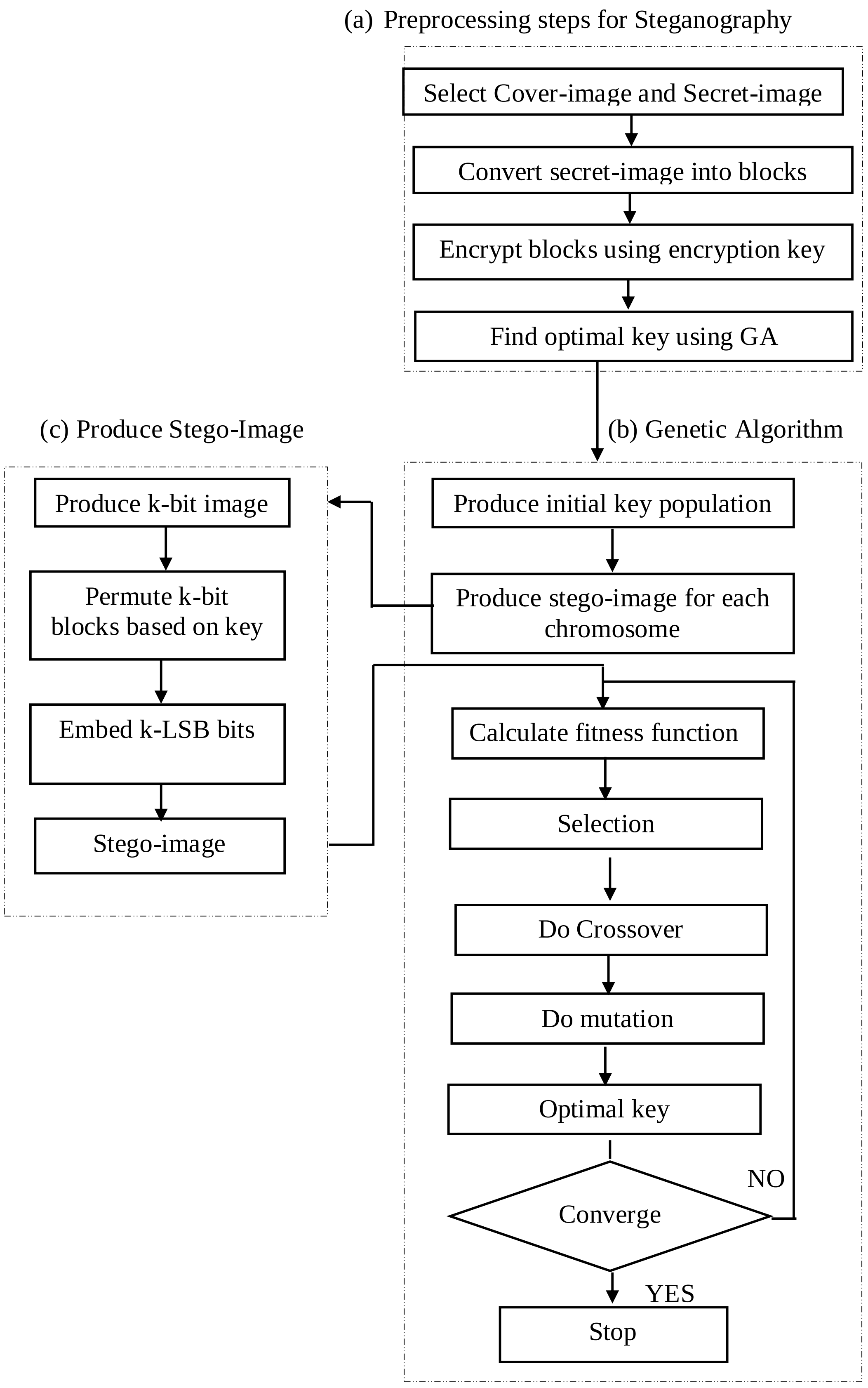

(a) Preprocessing steps for Steganography
Select Cover-image and Secret-image
Convert secret-image into blocks
Encrypt blocks using encryption key
Find optimal key using GA
(b) Genetic Algorithm
Produce initial key population
Produce stego-image for each chromosome
Calculate fitness function
Selection
Do Crossover
Do mutation
Optimal key
Converge
NO
YES
Stop
(c) Produce Stego-Image
Produce k-bit image
Permute k-bit blocks based on key
Embed k-LSB bits
Stego-image

Figure 3.6: Flowchart of the Proposed Method

3.2 Extracting Algorithm

The following procedure is carried out on the other side of the communication to extract the hidden secret image:

Input: stego-image.

Output: secret image.

1. Extract decimal k-bit image from stego-image by

   $$I_k = I \, Mod \, 2^k$$

   Where I: the pixel value.

2. Convert the extracted image to k-bit binary image I'_k(encrypted secret-image).
3. Apply reverse key permutation.
4. Reshape the extracted encrypted secret-image to a row vector of length (L = k×R×C).

   Where **R** represents the number of rows in the secret image and **C**, the number of columns.

5. Reshape the vector into 8-bit, convert the extracted encrypt secret-image to decimal image I''_k.
6. Split I''_k into blocks of n-by-n pixels.
7. Apply inverse encryption on blocks and get I_s(Extracted original secret image).

3.3 Measurement Metrics

In order to measure the level of image distortion resulting from hiding messages in the proposed method, the image quality, in regards to the peak signal-to-noise ratio (PSNR), the more widely used measurement of steganography performance in image-processing, is evaluated. The PSNR, expressed as a logarithmic decibel (dB) scale, is represented by the following Equation when 8-bit blocks are considered:

$$\text{PSNR} = 10\times\log_{10}\frac{255^2}{MSE}\ \ (dB) \tag{1}$$

MSE, the mean square error between the stego-image and the cover-image, is calculated as:

$$\text{MSE} = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(x_{ij} - y_{ij}\right)^2 \tag{2}$$

Where the pixel value of the cover-image is denoted by x_{ij}, and that of the stego-image is represented by y_{ij}, while n and m respectively represent the height and width of the cover-image [26].

It is noteworthy that the original image and the stego-image are most similar in larger PSNR values. Generally speaking, it is difficult for the human eye to notice stego-image distortion when the PSNR value is larger than 30dB. In contrast, the distortion is easily detected when the PSNR is below 30dB, meaning that the stego-image's quality is low and the distortion is high.

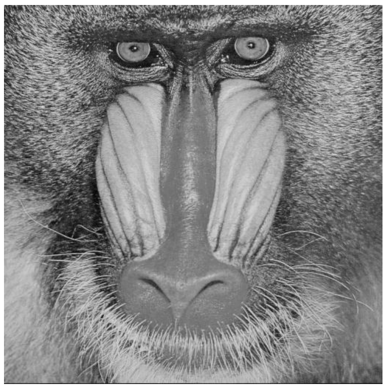
Chapter 4

RESULTS AND DISCUSSIONS

4.1 Results and Discussions

The goal of these experiments has been to simultaneously increase the stego-image’s quality and reduce its complexity. To do this experiment, four standard 8-bit per pixel cover images were used. The cover images (grayscale images) are: "Lena", "Baboon", "Barbara", and "Pepper", seen below in that order; the size of each image is 512 x 512 pixels as shown in Figure 4.1.

(A) Lena


(B) Baboon

(C)Barbara

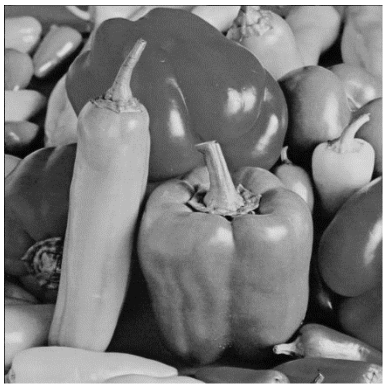

(D) Pepper

Figure 4.1: Cover-Images

The secret image (known as "tiff" image) is a grayscale image, 512 × 256 pixels in size for 4-LSB insertion when k-LSB is large as shown in Figure 4.2. Additionally, it is assumed that the sizes of the secret images are 384 × 256 pixels for 3-LSB insertion, 256 × 256 pixels for 2-LSB insertion, and 256 × 128 pixels for 1-LSB insertion.

Figure 4.2: Secret-Image

Assuming that k is the number of embedding bit from the secret image into the LSBs of the cover image and the chromosome in the GA contains 2^k genes, when k = 4, 2^4 = 16 genes in one chromosome. Meaning that the total number of possible key permutations employable for the purpose of embedding the data = 16! (20,000 billion key permutations), whereas when k = 3, 2^3 = 8 genes in one chromosome, and the total number of possible key permutations= 8! (40320 key permutations), when k = 2, 2^2 = 4 genes in one chromosome, the total of possible key permutations = 4! (24 key permutations), when k = 1, 2^1 = 2 genes in one chromosome, the total of possible key permutations = 2! (2 key permutations) that can be used in embedding the data. To find the number of embedding bits from the secret-image in the cover-image, the following formula is used:

$$\frac{m1\times \mathrm{n}1}{m2\times \mathrm{n}2} \qquad (3)$$

Where m_1 and n_1 respectively represent the width and height of the cover-image and m_2 and n_2 denote the width and height of the secret image respectively:

$\frac{512\times 512}{512\times 256} = \frac{8}{4}$ (embed 4 bits from the secret-image in LSBs of the cover-image).

$\frac{512\times 512}{384\times 256} = \frac{8}{3}$ (embed 3 bits from the secret-image in LSBs of the cover-image).

$\frac{512\times 512}{256\times 256} = \frac{8}{2}$ (embed 2 bits from the secret-image in LSBs of the cover-image).

$\frac{512\times 512}{256\times 128} = \frac{8}{1}$ (embed 1 bits from the secret-image in LSBs of the cover-image).

To determine the effectiveness of the proposed method in cases where the k-LSB value is large (4-LSB insertion), the genetic parameters used for the optimal key selection in our proposed method are listed as follows:

- Population size = 50
- Cross over ratio = 0.7
- Mutation ratio = 0.1
- Elitism size = 0.3
- Tournament selection = 5
- NO. of generation = 13

MSE and PSNR are used in the present study to calculate the stego-image's quality. Figure 4.3 shows the fitness function, which is used here to mean the Mean Square Error (MSE), on the x-axis while the y-axis represents multiple population sizes (10-100). The variables are combined so as to see the effect of multiple population sizes on the fitness function (MSE) shown in Figure 4.3. It is evident from the graph in Figure 4.3 that the MSE gradually decreased to an optimal solution (50) that is the MSE equivalent of 27.72. Subsequent population size increases caused the MSE to become stable because the elitism selection ensures that the best members of the present generation are included in the next generation.

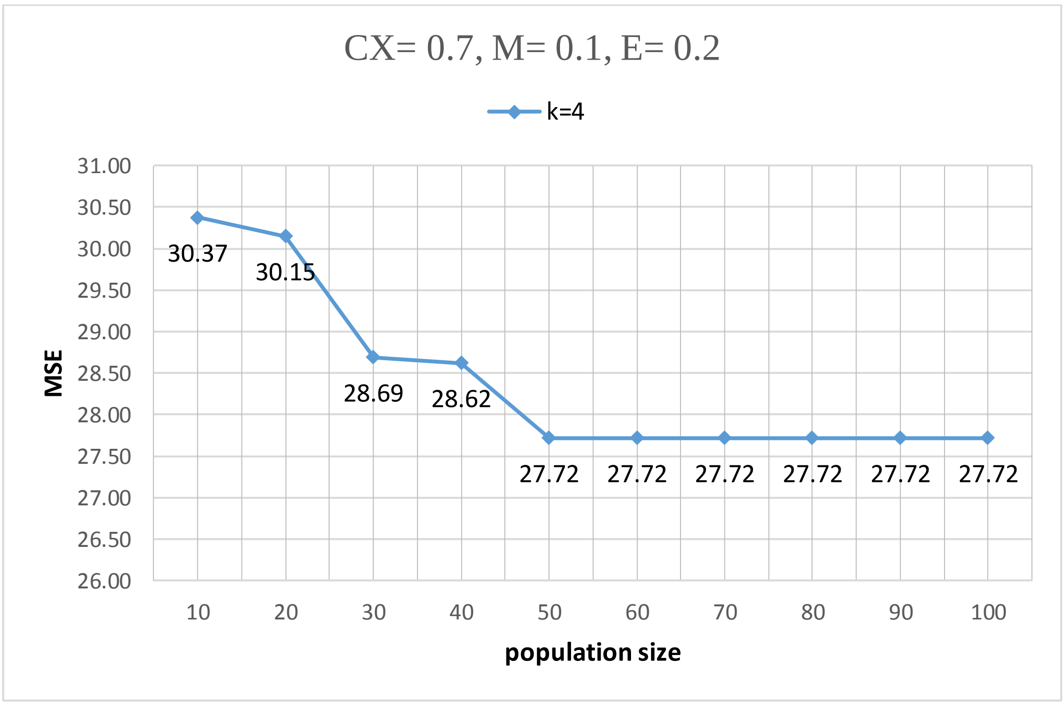


Figure 4.3: MSE vs. Optimal Population Size

The results below show the effects of multiple crossover rates (0.6-0.9) when the population size (Pop) = 50, and the mutation rate (M) = 0.1. The x-axis represents the MSE as a fitness function while the y-axis represents the different crossover rates. It is evident from the graph in Figure 4.4 that the MSE dropped from a crossover rate of 0.6, for which the MSE is equal to 29.46, to an optimal crossover rate of 0.7, which has the least MSE value at 27.80 but increased slightly to 27.83 when the crossover rate was 0.8 and even further to 28.64 with a crossover rate of 0.9. From the Figure below, the optimal crossover rate appears to be 0.7.

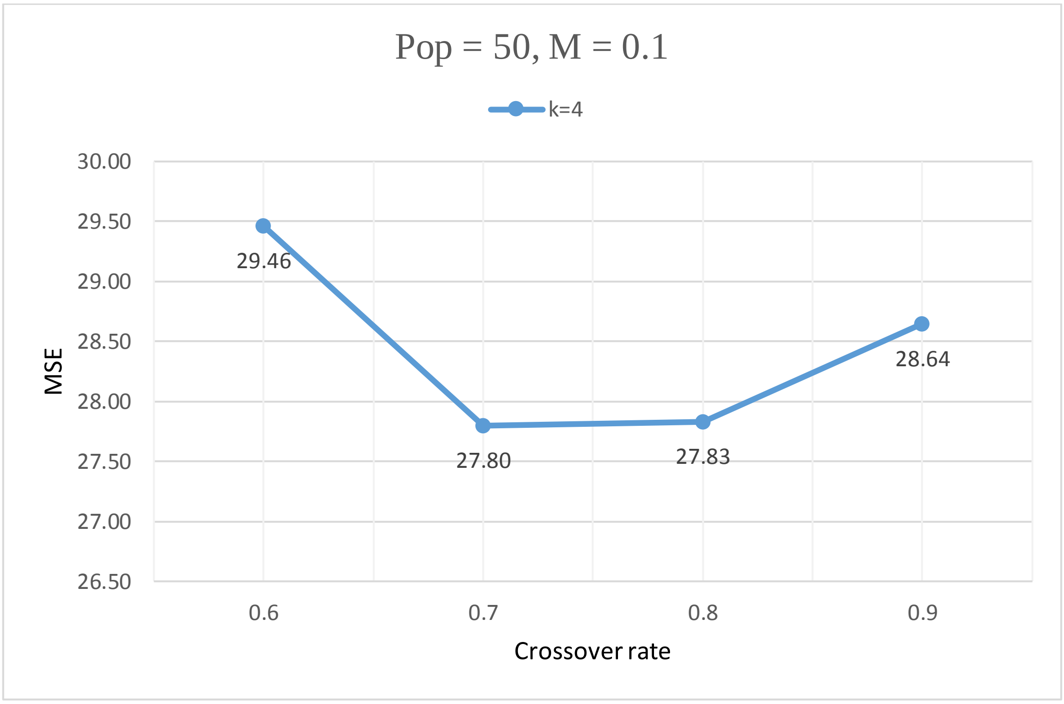


Figure 4.4: MSE vs. Optimal Crossover Rate

Figure 4.5 demonstrates the effect of multiple elitism sizes (0 – 0.5) after fixing the previous parameters – population size (Pop) = 50, and crossover rate (CX) = 0.7, and mutation rate (M) = 0.1. The Figure below shows: on the x-axis, the MSE as a fitness function, and on the y-axis, different elitism rate. It is evident from the graph that the MSE decreased slightly between elitism rate 0, which has MSE equal to 27.80, and elitism rate 0.2, which has an MSE equal to 27.72, then decreased dramatically at elitism rate 0.3 with an MSE equal to 27.15. Although elitism rate 0.4 and 0.5 respectively were more stable, according the graph, the optimal elitism rate is 0.3, when the MSE is equal to 27.15.

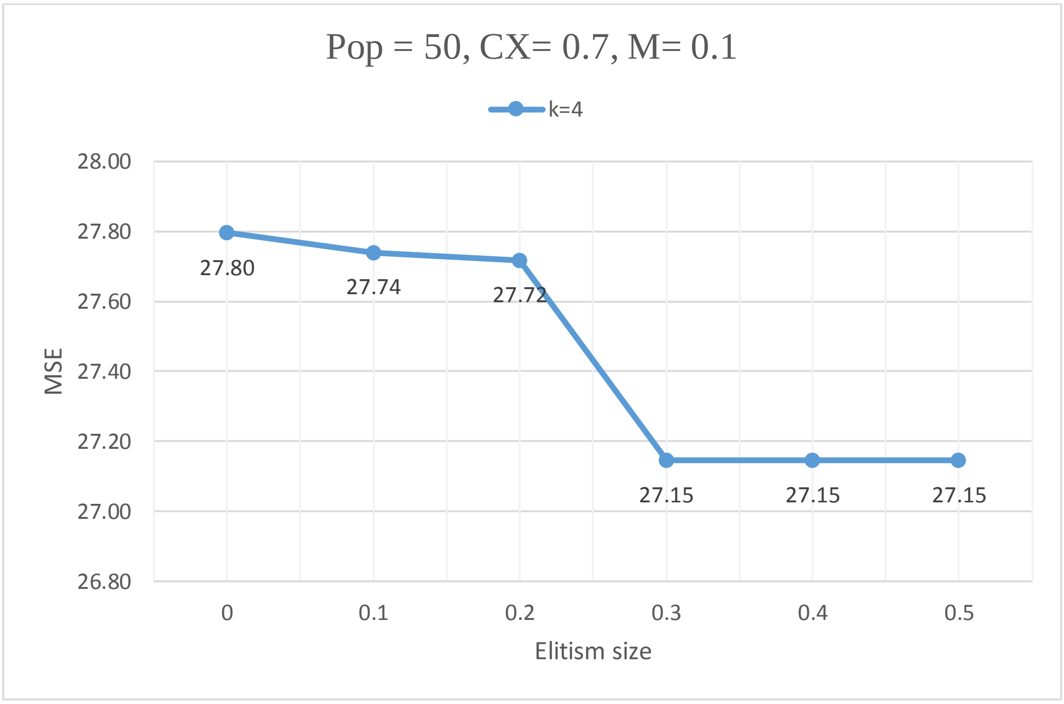


Figure 4.5: MSE vs. Optimal Elitism Size

Figure 4.6 shows the effects of multiple tournament selections (2-8) on a population size (Pop) = 50, crossover rate (CX) = 0.7, mutation rate (M) = 0.1, and elitism rate (E) = 0.3. The x-axis on the graph represents the MSE as a fitness function while the y-axis shows different tournament selections. It can be seen from the graph in Figure 4.6 that the MSE decreased slightly from a tournament selection equal to 2 with MSE equal to 27.15, to a tournament selection equal to 5 which has the least MSE at 26.93. The fitness function was subsequently increased gradually till the tournament selection was equal to 6, and MSE was equal to 28.80, after which the tournament selections were increased to 7 and 8 respectively. According to the graph, the optimal tournament selection is 5, which has the least MSE at 26.93, meaning that 5 chromosomes are picked at random then sorted according to their respective fitness functions (MSE) and the best two chromosomes are taken as parents.

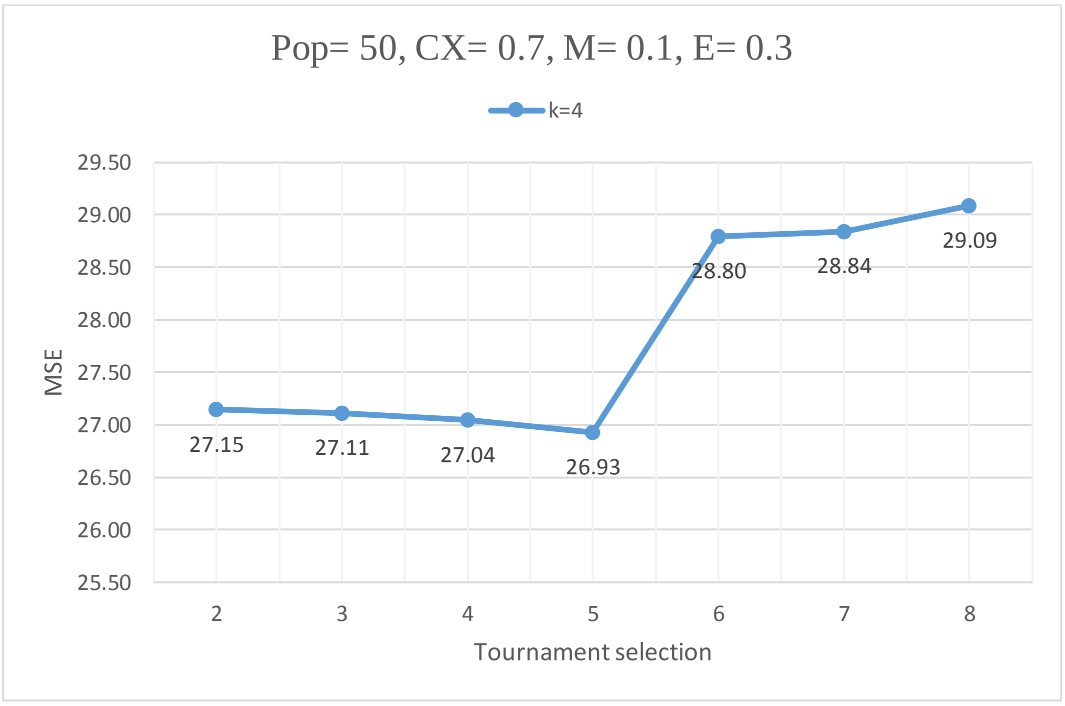


Figure 4.6: MSE vs. Optimal Tournament Selection

Figure 4.7 show the effects of multiple mutation rates (0.00125-0.3) ($\frac{1}{800}$ = 0.00125; where **800** is the number population size multiply with the chromosome length which is equal to 16 genes because this experiments use global optimization, and **1** represent a gene in the chromosome) when population size (Pop) = 50, crossover rate (CX) =0.7, mutation rate (M) = 0.1, elitism rate (E) = 0.3, and tournament selection (T_s) = 5. The Figure shows: on its x-axis, the range of fitness functions, while the y-axis demonstrates different mutation rates starting from 0.00125 up to 0.3. According to the fitness function, the MSE dropped slightly from a mutation rate of 0.00125 down to 0.1. After that, as can be seen from the plot, the fitness function starts rising considerably when the mutation rate increases between 0.1 and 0.3. Consistent with the fitness function, the optimal mutation rate is equal to 0.1 because it has the least Mean Square Error (MSE) at 26.93. Increasing mutation rate to some extent can be tolerable in large population but in limited population size like in Figure 4.7 (after 0.1 mutation). Search may become random.

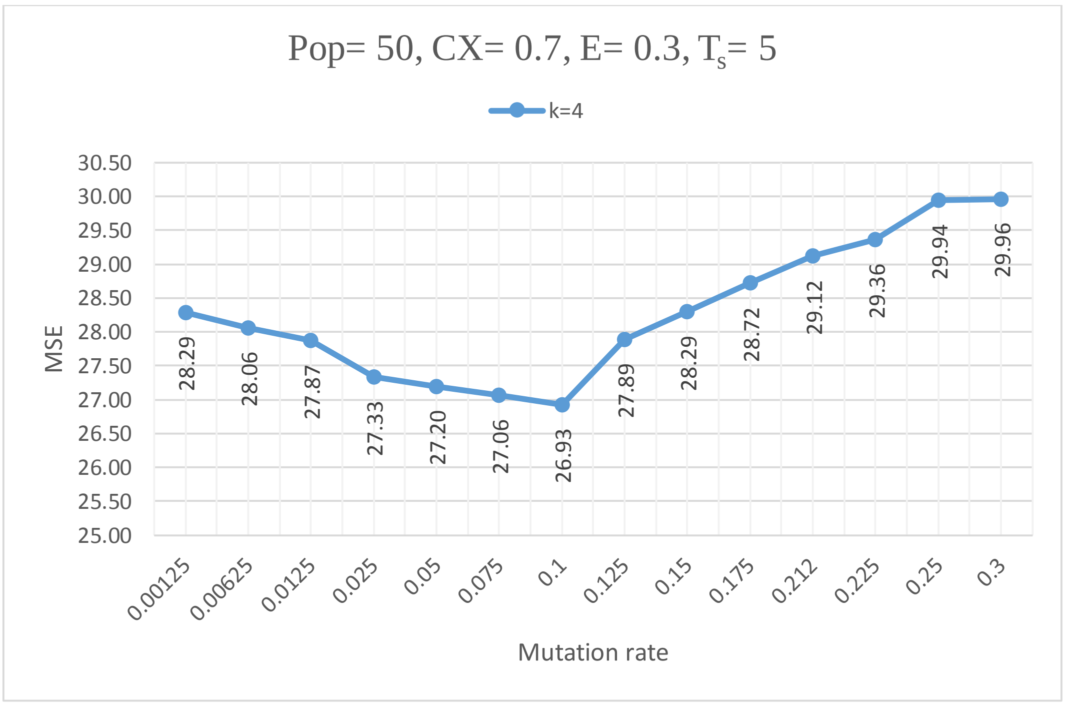


Figure 4.7: MSE vs. Optimal Mutation Rate

Figure 4.8 shows the result of embedding the encrypted secret image (known as the "tiff" image), as shown in the Figure 4.2, into the cover image "Lena", as shown in Figure 4.1 (A). The experiment compares the embedding results obtained using the methods employed by Wang et al. [33], for which the PSNR value is equal to 32.565 dB, Marghny et al. [29], for which the PSNR value was 32.931 dB, and the proposed method, which has the highest PSNR value at 33.410 dB, when the number of embedding bits from the secret image into the cover image is large (k = 4-LSBs insertion). Furthermore, it can be seen from the Figure 4.8 that using the proposed method improves the stego-image’s quality.

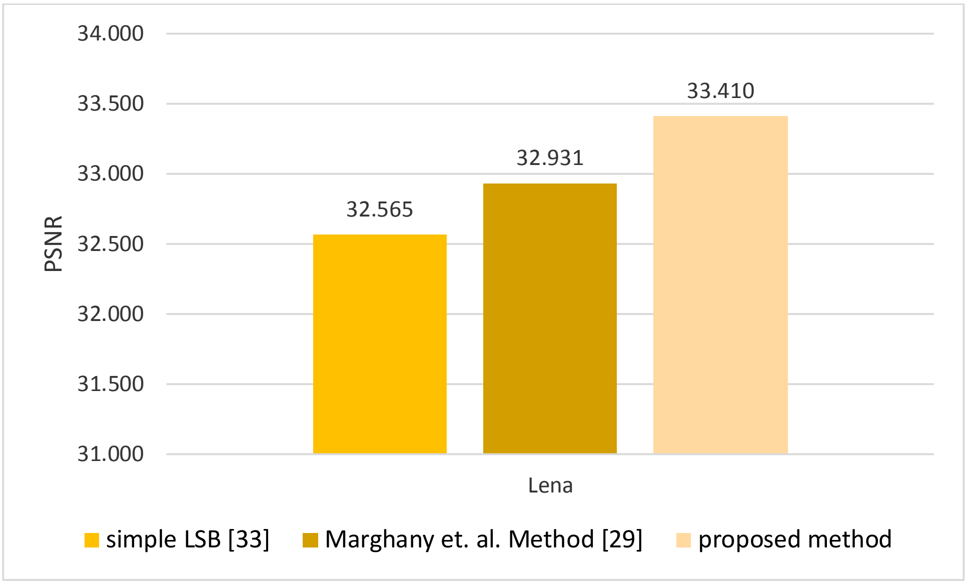


Figure 4.8: Result of Embedding the Encrypted Secret-Image into the 4-LSB of the Cover-Image (Lena)

Figure 4.9 illustrates the result of embedding the encrypted secret image ("tiff" image) shown in Figure 4.2 into the cover image "Baboon", shown in Figure 4.1 (B). The experiment compared the results obtained by embedding using the Wang et al. method [33], which had a PSNR value of 31.936 dB, the Marghny et al. method [29], with a PSNR value equal to 32.619 dB, and the proposed method, which has the highest PSNR value at 33.080 dB, when the number of embedding bits from the secret image to the cover image is large (k = 4-LSBs insertion). As with the previous example, it can also be seen from the Figure 4.9 that the stego-image's quality is improved by the proposed method.

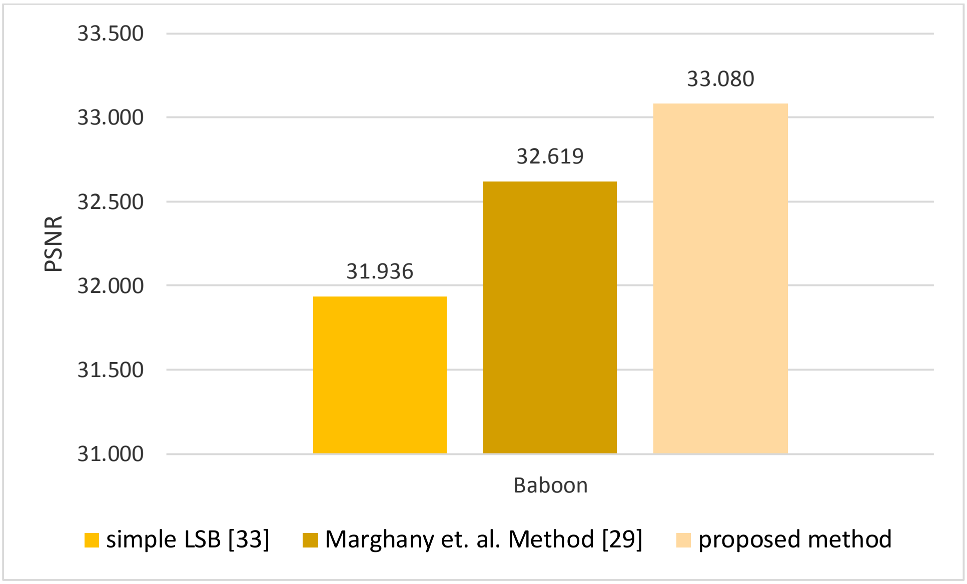


Figure 4.9: Result of Embedding the Encrypted Secret-Image into the 4-LSB of the Cover Image (Baboon)

Figure 4.10 shows the result of embedding the encrypted secret image in Figure 4.2 in the cover image "Barbara", as shown in the figure 4.1 (C). It provides a comparative overview of the embedding results gotten through the method posited by Wang et al. [33], with a PSNR value of 32.941 dB, Marghny et al. method [29], with a PSNR value equal to 33.843 dB, and the proposed method with the highest PSNR value at 34.208 dB when the number embedding bits from the secret image in the cover image is large (k = 4-LSBs insertion). This stego-image’s quality is also enhanced by using the proposed method.

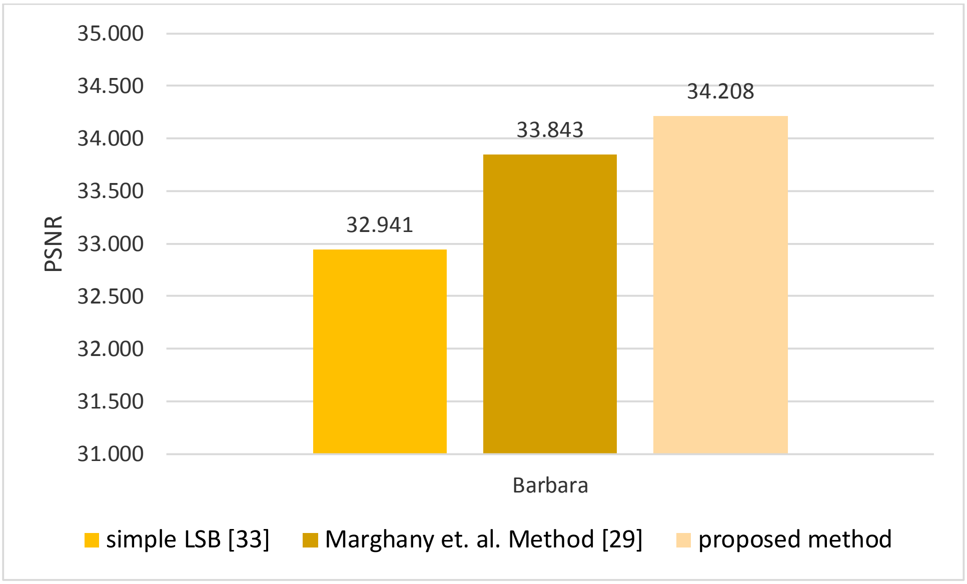


Figure 4.10: Result of Embedding the Encrypted Secret-Image into the 4-LSB of the Cover-Image (Barbara)

Figure 4.11 shows the results of embedding the encrypted secret image (Figure 4.2) into the cover images "Pepper", shown in the Figure 4.1 (D). The experiment compares the results gotten using the Wang et al. method [33], PSNR value is equal to 32.066 dB, Marghny et al. method [29], PSNR value is equal to 32.882 dB, and the proposed method, which has the highest PSNR value equal to 34.208 dB when the number of embedding bits from the secret image into cover image is large (k = 4-LSBs insertion). The proposed method evidently improves the quality of the stego-image.

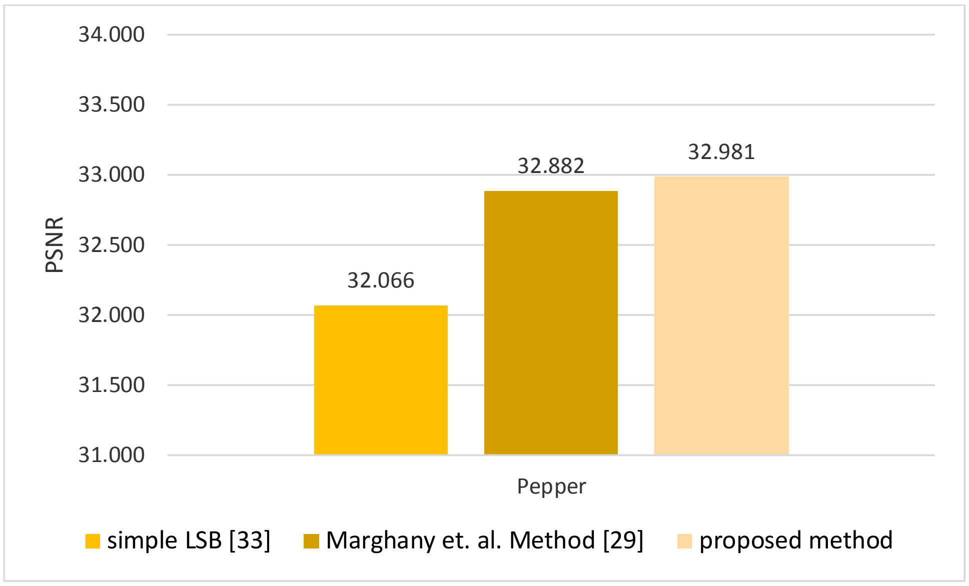


Figure 4.11: Result of Embedding the Encrypted Secret-Image into the 4-LSB of the Cover-Image (Pepper)

4.1.1 Performance Evaluation Comparison of Proposed Method

The GA has population size Np and runs for Ng generations since each block requires L times computations to determine its fitness value, a total of Ng×Np×L computations are performed. The computational complexity / block is then O(Ng×Np) [30].

Other factors, including elitism, crossover, mutation, and selection function are not considered as they exert less of an influence on complexity [9].

Figure 4.12 demonstrates the effect of 100 generations using "Lena" as a cover image and "tiff" as a secret image with the optimal genetic algorithm parameters: population size (Pop) = 50, crossover rate (CX) = 0.7, mutation rate (M) = 0.1, elitism size (E) = 15, and tournament selection (T_S) = 5. Figure 4.8 shows the MSE, which represents the fitness function in this study, on the x-axis, and the effect of 100 generations on the y-axis. It is evident from the plot in Figure 4.12 that the fitness function (MSE) decreased gradually from the first generation, which has an MSE equal to 33.30, to the thirteenth generation, with an MSE equal to 26.93, becoming stable after the thirteenth generation. The time complexity calculated using the Marghny et al. method [29] is 1000 (10×100 = 1000; where 10 is the number of the population size and 100 is the number of generation in GA), whereas the complexity in the proposed method is equal to 650 (50×13 = 650; where 50 is the number of population size and 13 is the number of generation in GA). The proposed method also resulted in a minimum number of complexity, less Mean Square Error (MSE), and achieved less computational complexity than the Marghny et al. method.

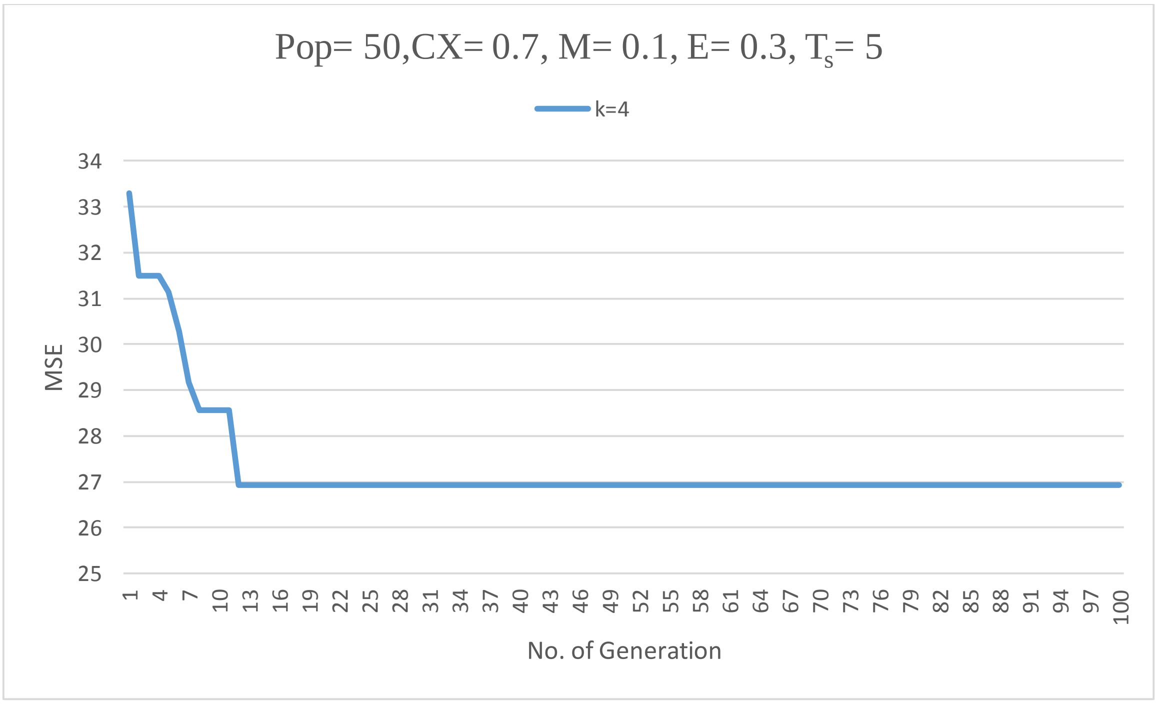


Figure 4.12: MSE vs. No. of Generation

The Figure 4.13 shows the CPU Time on different population size (10-100) using "Lena" as a cover image and "tiff" as a secret image with the optimal genetic algorithm parameters. The x-axis represents the CPU Time (in seconds) while the y-axis represents the different population size. It is evident from the graph in Figure 4.13 that the CPU Time increases gradually with increase in population size. The program was written in Matlab 2015, and run on a Dell Laptop of Core i3 intel processor, with 8 G RAM running 64bits Windows 7 operating system.

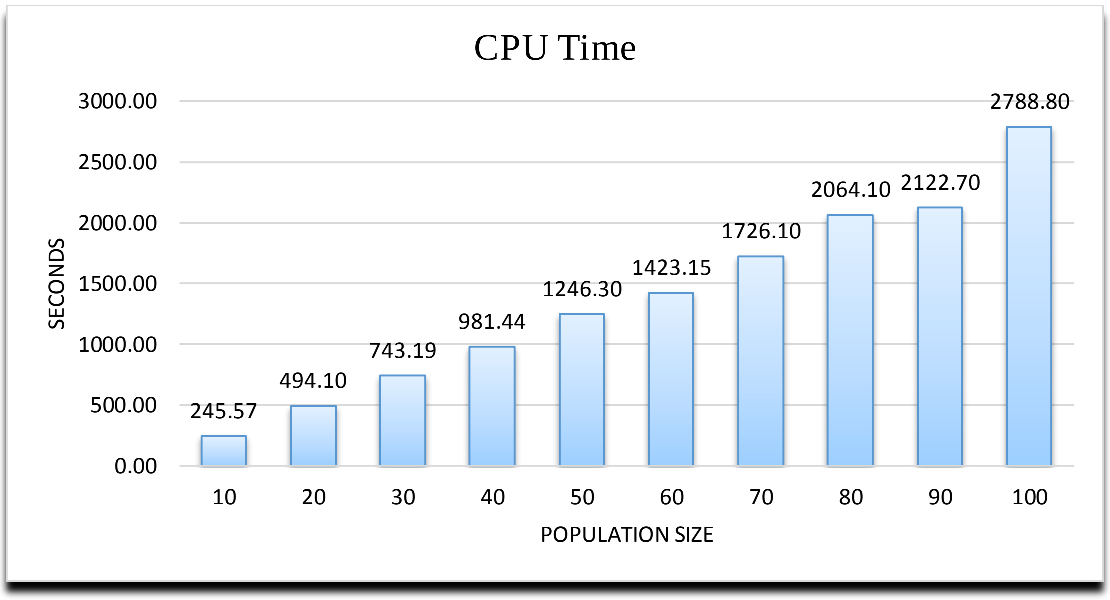


Figure 4.13 CPU Time vs. Population size

Chapter 5

CONCLUSION AND SCOPE FOR FUTURE WORK

5.1 Conclusion

As one of the more common data-hiding techniques, steganography is a method of discreetly passing messages that involves hiding the intended secret message inside other information. Steganography generally involves two processes: in the first, the sender inserts the secret message into a cover-object for the receiver to extract in the second process. The result of the sender embedding a secret message into the cover-object is known as the stego-object and the process of embedding involves the use of a key known only to the sender and receiver so as to protect their communication from unintended audiences.

Steganographic systems generally use multimedia objects as their cover-objects. Based on the type of steganography utilized, the cover-object could be either text, audio, video, protocol, or an image. Also, depending on the particular steganographic technique used, the cover-image could either remain intact after the receiver extracts the secret message (reversible techniques) or be distorted (irreversible techniques).

This particular study is concerned with image steganography i.e. embedding a secret-image in another cover-image. The LSB method is the most common method used in image steganography. It involves a process whereby some or all of the bytes in the least significant bit are replaced with bits from the secret-image. LSB however, often results in a low quality stego-image. As such, two methods have been suggested to improve the quality of the resulting stego-image. In the first method, a genetic algorithm is used to find the optimal key permutation, while the second method involves elitism selection.

The method improves upon the conventional Genetic Algorithm by adding elitism selection to it. As the Genetic Algorithm is used to find the substitution with the highest PSNR value in a generation, the addition of elitism selection ensures that this optimal PSNR value is transferred when subsequent generations are formed through cross-over and mutation. Furthermore, the proposed method improves on that of Marghany et al. by using optimal parameters – for population size, cross-over rate, elitism, tournament selection, and mutation – as opposed to fixed parameters.

The proposed method was used to embed the 'tiff' image (secret-image) in Lena, Baboon, Barbara, and Pepper (cover-images). It was found that proposed method improved the quality of the resulting stego-images and also reduced the computational complexity of the Genetic Algorithm by 35% from 1000 to 650 generations.

5.2 Scope for Future Work

The LSB substitution technique allows for a large amount of information to be embedded in the cover-object. The LSB technique used in this study allowed us to embed four bits of the secret-image in a single pixel of the cover-image. Future studies can go further to explore the possibility of hiding five bits in a single pixel of the cover-image by replacing the LSBs of each pixel. Furthermore, subsequent studies could also explore the possibility of changing the value for every pixel in the cover-image so as to conceal the hidden data.

REFERENCES

[1] Sharma, V. K. & 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*, Vol. 36, No.1, pp. 1-8.

[2] Abraham, A. & Paprzycki, M. (2004). Significance of steganography on data Security. *In Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference*, Vol. 2, pp. 347-351.

[3] Pavani, M., Naganjaneyulu, S. & Nagaraju, C. (2013). A Survey on LSB Based Steganography Methods. *International Journal of Engineering and Computer Science*, Vol. 2, Issue 8, pp. 2464-2467.

[4] Hosmer, C. (2006). Discovering hidden evidence. *Journal of Digital Forensic Practice*, Taylor & Francis Group, Vol.1, No.1, pp.47–56.

[5] Reddy, V. L., Subramanyam, A. & Reddy, P. C. (2011). Implementation of LSB Steganography and its Evaluation for Various File Formats. *International Journal of Advanced Networking and Applications*, Vol. 2, Issue 5, pp. 868-872.

[6] Mandal, P. C. (2012). Modern Steganographic technique: A Survey. *International Journal of Computer Science & Engineering Technology (IJCSET)*, Vol. 3, No. 9, pp. 444-448.

[7] Kanzariya, Nitin K., Nimavat, & Ashish V. (2013). Comparison of Various Images Steganography Techniques. *International Journal of Computer Science and Management Research*, Vol. 2, Issue 1, 2013, pp. 1213-1217.

[8] Chan, C. K. & Cheng, L. M. (2004). Hiding data in images by simple LSB substitution. *Pattern Recognition*, Elsevier, Vol. 37, No. 3, pp. 469 – 474.

[9] Cem, E., & Kadri, H. (2000). Multiuser Detection Using a Genetic Algorithm in CDMA Communications Systems. IEEE transactions on communications, Vol. 48, No. 8, pp. 1374-1383.

[10] Jain, V.; Kumar, L.; Sharma, M. M.; Sadiq, M. & Rastogi, K. (2012a). Public-Key Steganography Based On Modified LSB Method. *Journal of Global Research in Computer Science,* Vol. 3, No. 4, pp. 26-29.

[11] Chandramouli, R. & Memon, N. (2001). Analysis of LSB based image steganography techniques. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, Vol. 3, pp. 1019-1022.

[12] Morkel, T., Eloff, J.H.P. & Olivier, M.S. (2005). An Overview of Image Steganography. *Information and Computer Security Architecture (ICSA) Research Group*.

[13] Govinda, B., Vijay, A. & Kailash, P. (2013). Steganography: Exploring an ancient art of Hiding Information from Past to the Future. *International Journal*

of Engineering and Innovative Technology (IJEIT). Volume 3, Issue 4, pp. 192-194.

[14] Katzenbeisser, S. & Petitcolas, F.A.P. (2000). *Information Hiding Techniques forSteganography and Digital Watermarking,* Artech house.

[15] Ramakrishna, H. & Jagadeesha, S. (2015). Design and Implementation of Image Steganography by using LSB Replacement Algorithm and Pseudo Random Encoding Technique. *IJRIT in Computing and Communication*, Vol. 3 Issue 7, pp. 4415-4420.

[16] Nagham, H., Abid, Y., Badlishah, A. & Osamah, M. (2012). Image Steganography Techniques: An Overview. *International Journal of Computer Science and Security (IJCSS),* Vol. 6, Issue 3, pp. 168-187.

[17] Lokhande, U. & Gulve, A.K. (2014). Steganography using Cryptography and Pseudo Random Numbers. *International Journal of Computer Applications*, Vol.96, No.19, pp. 40-45.

[18] Aarti, M. (2015). Data Hiding System Using Cryptography & Steganography:A Comprehensive Modern Investigation. *International Research Journal of Engineering and Technology (IRJET),* Vol. 02, Issue 01, pp. 397-403.

[19] Tanmoy, S. & Sugata, S. (2014). Steganalysis: Detecting LSB Steganographic Techniques. *IJESM*, Vol.4, Issue 2, Page 34.

[20] Marghny, H. M., Naziha, M. & Mohamed, A. B. (2012). Data Hiding Technique Based on LSB Matching towards High Imperceptibility. MIS Review 2012 Department of Management Information Systems, College of Commerce National Chengchi University and Airiti Press Inc, Vol. 18, No. 1, September (2012), pp. 57-69.

[21] Haleh, V. & Kenneth, D. J. (2012). Genetic Algorithms as a Tool for Feature Selection in Machine Learning, Springer.

[22] Bir, B. & Yingqiang, L. (2003). Genetic Algorithm Based Feature Selection for Target Detection in SAR Images, *Image and Vision Computing Journal 21*, 591–608.

[23] Goldberg, D. E. (2002). Design of Innovation: Lessons From and For Competent Genetic Algorithms, Kluwer, Boston, MA.

[24] Chang, C. C. & Tseng, H. W. (2009). Data Hiding in Images by Hybrid LSB Substitution. Third International Conference on Multimedia and Ubiquitous Engineering, *IEEE,* pp. 360-363.

[25] Wang, R.Z., Lin, C.F. & Lin, J.C. (2000). Hiding data in images by optimal moderately significant-bit replacement. *IEE Electron. Lett.*, vol. 36, no. 25, pp. 2069–2070.

[26] Marghny, H. M. & Loay, M. M.(2016). High Capacity Image Steganography Technique based on LSB Substitution Method. *An International Journal Applied*

Mathematics & Information Sciences Appl. Math. Inf. Sci. 10, No. 1, pp-259-266.

[27] Naveen, K., Karambir, & Rajiv, K. (2012). A Comparative Analysis of PMX, CX and OX Crossover operators for solving Travelling Salesman Problem. *International Journal of Latest Research in Science and Technology*, Vol.1, Issue 2, pp. 98-101.

[28] Tournament selection (2017, May) Wikipedia.[Online]Retrieved form https://en.wikipedia.org/wiki/Tournament_ selection.

[29] Marghny, M, Fadwa, A, & Mohamed, B (2011). Data Hiding by LSB Substitution Using Genetic Optimal Key-Permutation. *International Arab Journal of e-Technology*, Vol. 2, No. 1, pp. 11-17.

[30] Rylander, B., (2001). Computational Complexity and the Genetic Algorithm. Ph.D. Dissertation, University of Idaho, USA, June, 2001.

[31] Umbarkar, A., & Sheth, P., (2015). CROSSOVER OPERATORS IN GENETIC ALGORITHMS: A REVIEW. *ICTACT journal on soft computing*, VOL: 06, ISSUE: 01, pp. 1083-1092.

[32] Abdoun, O., Abouchabaka, J., & Tajani, C., (2012). Analyzing the Performance of Mutation Operators to Solve the Traveling Salesman Problem. *International Journal of Emerging Sciences (IJES)*. Vol. 2. No.1.

[33] Wang, R. Z.,Chi-Fang Lin,C. F. & Lin, C. J.(2001). Image hiding by optimal LSB substitution and genetic algorithm. Pattern Recognition, pp. 671-683.

[34] Wang, R. Z.,Chi-Fang Lin,C. F. and Lin, C. J.(2001). Image hiding by optimal LSB substitution and genetic algorithm. Pattern Recognition, pp. 671-683.

[35] Chang, C. C., Hsiao, J. Y. and Chan, C. S. (2003). Finding optimal least significant-bit substitution in image hiding by dynamic programming strategy. *Pattern Recognition*, pp. 1583 – 1595.

[36] Marghny, H. M. and Hussein, I. B. K. (2012). Data Hiding by LSB Substitution using Gene Expression Programming. *International Journal of Computer Applications*, Vol. 45, No.14, pp. 13-20.

[37] Li, X. and Wang, J. (2007). A steganographic method based upon JPEG and particle swarm optimization algorithm. Information Sciences, vol. 177, no. 15, pp. 3099–3109, 2007.

[38] Gangeshawar, and Attri, J. (2015). Optimizing Image Steganography using Genetic Algorithm. *International Journal of Engineering Trends and Technology (IJETT)*, Vol. 24, No. 1, pp. 32-28.

[39] Chan, C.K. and Cheng, L.M. (2004). Hiding data in images by simple LSB substitution. *Pattern Recognition*, vol. 37, no. 3, pp. 469–474.