POPULAR DEMOCRATIC REPUBLIC OF ALGERIA
Higher education and scientific research's ministry
University of KASDI MERBAH OUARGLA

Faculty of new information technologies and communication
Department of computer and information technology

LMD Master 2019/2020

Option: Network Administration and Security

Dissertation

For the Master Degree in Network Administration and Security

Title:

Watermarking Algorithm Based on Image Projection and Genetic Algorithm

Presented by:

Anba Amal
Hacini KHadidja

Jury members:

Dr. Zga Adell	University of KASDI MERBAH OUARGLA	Jury-header
Dr. Akram Boukhamla	University of KASDI MERBAH OUARGLA	Examiner
Dr. Belekbir Djalila	University of KASDI MERBAH OUARGLA	Superviser

Academic Year 2019-2020

Acknowledgments

Thank you, ALLAH I always had our aid and our companion, in an accomplishment See the message and thank ALLAH.

To the spirit of our good, adultery parents.

To the spirit of our dearer-religious mothers, you are forgiven for us.

To the spirit of our dear brothers and brothers.

To our teacher Khadmallah Mouhammed Ali for all the efforts he made for us.

To our teacher Dr. Balkebir Djalila who overviews her advice and guidance for the right thing.

I would like to thank all your religious devotion, both old and easy All the means and requirements for our thought to be at best.

Religion and our folkside did not forget their pure preachers.

And finally, we did not forget the university family from the administration and the teachers and the students did not make us clear of any information.

ABSTRACT

This research has included developing the method of concealment in images the projection method. The secret message was first encryption into a binary string, and then the projection method is applied to the original image's cover as it produces a set of pixels determined by the method, which is hidden in the way of the Least Significant Bit (LSB) cell within certain locations, and then after it The genetic algorithm(GA) that works randomly to get the best generation of concealment has been applied, as it ensures the confidentiality of the information and improves the concealment that enables us to get the best value for Peak Signal to Noise Ratio(PNSR).

The PSNR, Bit Error Rate (BER), Mean Square Error (MSE), Average Difference (AD), Maximum Difference (MD), Peak Mean Square Error (PMSE), Normalized Cross-Correlation (NK), Structural Content (SC), Laplacian Mean Square Error (LMSE), Normalized Absolute Error (NAE) and Robustness Test (BER) standards were adopted to achieve results that are effective and efficient.

The application is implemented with NetBeans8.2 software support.

Keywords:

Steganography, Cryptography, Projection, Least Significant Bit (LSB), Genetic Algorithm(GA).

ملخص

اشتمل هذا البحث على تطوير طريقة الاخفاء في الصور طريقة الاسقاط (projection)اذ تم أولا تشفير الرسالة السرية الى سلسلة ثنائية، وبعدها تطبيق طريقة الاسقاط على غلاف الصورة الاصلية، اذ ينتج لنا مجموعة من البكسلات محددة حسب هده الطريقة فيتم الاخفاء فيها بطريقة الخلية الثنائية الاقل اهمية LSB ضمن مواقع معينة، وبعدها تم تطبيق الخوارزمية الجينية (GA) التي تعمل بشكل عشوائي للحصول على أفضل جيل للإخفاء، حيث انها تضمن سرية المعلومة وتحسن الاخفاء وتمكننا من الحصول على أفضل قيمة لل Peak Signal to Noise Ratio (PNSR) .

اعتمدت المقاييس Mean Square Error (MSE) , Bit Error Rate (BER) , PSNR, Average Difference(AD) ,Maximum Difference((MD) ,Peak Mean Square Error (PMSE) ,Normalized Cross-Correlation (NK), Structural Content (SC), Laplacian Robustness ,Mean Square Error (LMSE), Normalized Absolute Error (NAE) Test (BER)لإثبات دقة النتائج و كفاءتها.

أما التطبيق فنفذ باعتماد على البرنامج Netbeans8.2.

الكلمات المفتاحية:

الاخفاء، التشفير، الاسقاط، الخلية الثنائية الأقل أهمية (LSB) ,الخوارزمية الجينية(GA).

Table of content

Table of content

Table of content

Table of content

Figures list

Figures list

Figures list

Tables List

Abbreviation

Abbreviation	Description
IP	Internet Protocol
VIOP	Voix sur Internet Protocol
MIDI	Musical Instrument Digital Interface
ICMP	Internet Control Message Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
OSI	Open Systems Interconnection
MPEG	Moving Picture Experts Group
MIDI	Musical Instrument Digital Interface
AVI	Audio Video Interleave
WAVE	Waveform Audio File Format
RGB	Read Green Blue
LSB	Significant Bit least
GA	Genetic Algorithm
PSNR	Peak Square Noise Ratio
MSE	Mean square error
AD	Average Difference
MD	Maximum Difference
PMSE	Peak Mean Square Error
NK	Normalized Cross-Correlation
SC	Structural Content
LMSE	Laplacian Mean Square Error
NAE	Normalized Absolute Error
BER	Robustness Test

GENERAL INTRODUCTION

The science of confidential information security has become of great interest to researchers who are trying to obtain new and modern solutions and technologies to ensure the protection and confidentiality of the information sent and received via the global network of information (the Internet) without any penetration or detection by the intruders. Therefore, it was necessary to keep pace with the development of information. The establishment of advanced techniques and tools to ensure the confidentiality of information, hence the science of steganography.

Steganography is a protection method that renders the sent and received data invisible by hiding certain messages within a specific jacket. The goal of concealment is not to raise any suspicion about the existence of hidden data, while there are people who have doubts about whether there is a protected data or not, and check it for it. The process in which one of the parties tries to detect, read or change the presence of hidden information is called Steganalysis.[1]

Therefore, the need arose to create a developed method for the purpose of communicating information while ensuring its integrity and rejecting any unauthorized access to this information.

Although messages hiding is a good way to protect information, it is easy to be retrieved manipulated. For that, sophisticated and secretive technology to preserve information is needed, especially with the emergence and development of the global network of information (the Internet). In this thesis, we proposed the use of projection method that is based on projecting each pixel location of the cover image onto new location of a virtual image. The virtual image is obtained by positioning the original image based on the distance and the angle of view. The proposed method ensure the entegrity and the security of the hidden data but it has two main problems that are appeared from using high-low values of distance and angle. Those values result on non-using of certain pixels location, for that, we proposed the use of genetic algorithms (GA) to overcome those problems.

General Introduction

GA are one of the general search algorithms based on the mechanism of natural selection and the natural gene system, and the idea of working here depends on an increase of the hidden information confidentiality and improve the location of hiding process by applying GA operations (crossover, selection, mutation). The best generation is used to hide the secret message in the best locations, and on this basis, we select some chromosomes from a large number of solutions and apply crossover and mutation between these solutions in order to create better solutions.

The main goal of our application is to create a program that hides confidential information in the form of an image in another image so that no one can extract this information except for the authorized persons who have the key to hide.

This thesis is divided into three chapters:

The first chapter provides a general description of the encryption technology, the methods used in it and its types. A general description of the hiding method, its types, and the most important methods used. We conclude by a comparison between encryption and hiding method.

In the second chapter, we will shed light on the process of LSB hiding, the projection method hiding and the combining projection and genetic algorithms. We will discuss some previous studies in the process of hiding based GA.

In the third chapter, we will present a set of tests done in the context of the use of our method and we will end this manuscript with a conclusion that summarizes the results obtained.

CHAPTER I: STEGANOGRAPHY DEFINITION AND HISTORIC

1 Introduction

Steganography is the art and science of communicating in such a way that it hides the existence of the communication. Thus, steganography hides the existence of data so that no one can detect its presence [2]. In steganography the process of hiding in formation content inside any multimedia content like image, audio, video is referred as "Embedding". For increasing the confidentiality of communicating data. There are different types of steganography techniques each have their strengths and weaknesses. In this chapter, we review the different security and data hiding techniques that are used to implement a steganography [3]and the cryptography techniques.

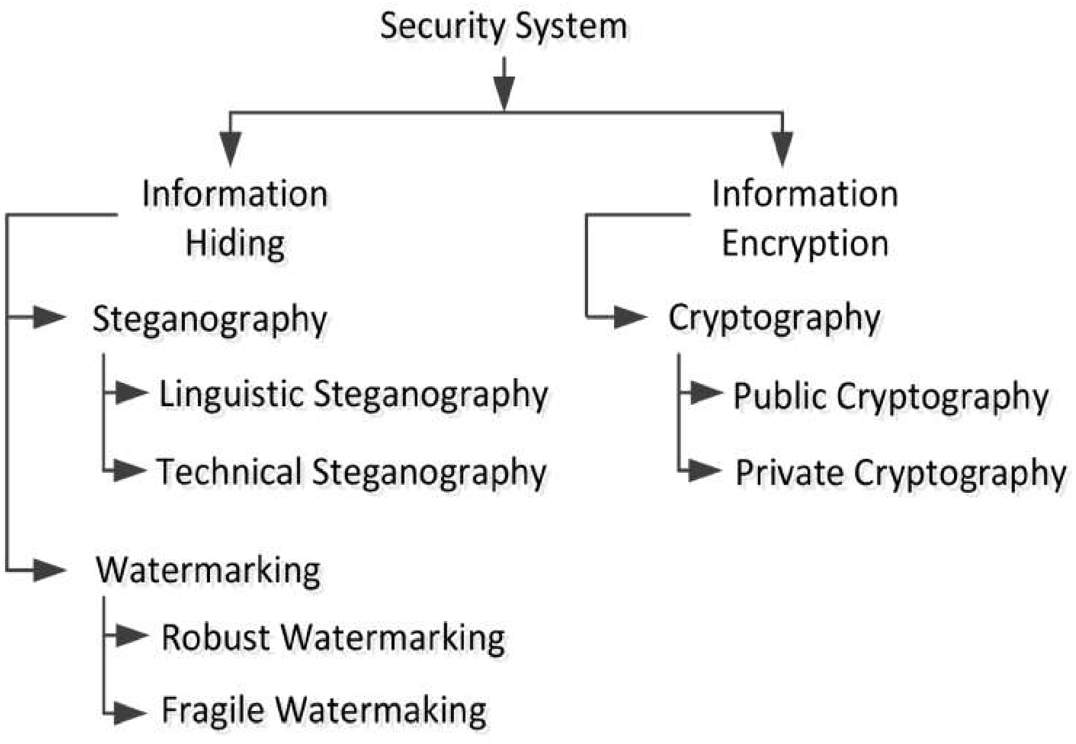


Figure 1.1: Basic Security System Branches [4].

2 Historie:

The earliest use of steganography was documented by greek historian HERODOTUS (484 -445 BC) in his investigative book, the story is that in the second medical war

DEMARATUS transmit the information to Sparta: "he took a double tablet, scratched the wax, then wrote on the wood even XERSES projects; then he covered his message with wax".

In World War I (Petitcolas1999) prisoners of war hid messages in Morse code by using dots and dashes on alphabets.

In world “war II”, The Americans and British used in secret communications (Caldwell 2003) invisible ink that could glow in the heat. While German spies used invisible dots in letters and changed heights of letter-strokes to hide messages. Currently, the focus has been on various forms of digital steganography.

Simmons G J (1984) illustrate the concept of steganography with an example called the “Prisoners’ Problem.” Alice and Bob are the two prisoners. Their goal is to develop an escape plan and bust out of jail, the problem is that the only way to communicate is through Wendy, the warden. If she is active, then she would interfere and try to remove or modify the confidential message or simply stop the communication. Hence Alice and Bob have to communicate with a shared secret key in such a way that Wendy would not be able to decipher their plan without the key. This might be done by steganography[5] .

Recent usage of steganography consists of special inks for hidden messages on currencies, watermarking and fingerprinting for copyright protection.

3 Cryptography:

3.1 Cryptography Definition:

Cryptography is a method of protecting information and communications through the use of codes so that only those for whom the information is intended can read and process it. The pre-fix "crypt" means "hidden" or "vault" and the suffix "graphy" stands for "writing."The other side , is associatedwith the process of converting ordinary plain text into unintelligible text and vice-versa. It is amethod of storing and transmitting data in a particular form so that only those for whom it is intended can read and process it. Cryptography not only protects data from theft or alteration, but can also be used for user authentication[6].

Cryptography is closely related to the disciplines of cryptology and cryptanalysis. It includes techniques such as merging words with images, and other ways to hide information in storage or transit. However, in today's computer-centric world, cryptography is most often associated with scrambling plaintext (referred to as clear text) into cipher text (a process called encryption), then back again (known as decryption)[7].

3.2 Cryptography Techniques:

When transmitting electronic data, the most common use of cryptography is to encrypt and decrypt email and other plain-text messages. The simplest method uses the symmetric or "secret key" system. Here, data is encrypted using a secret key, and then both the encoded message and secret key are sent to the recipient for decryption. The problem is if the message is intercepted, a third party has everything they need to decrypt and read the message. To address this issue, cryptologists devised the asymmetric or "public key" system. In this case, every user has two keys: one public and one private. Senders request the public key of their intended recipient, encrypt the message and send it along. When the message arrives, only the recipient's private key will decode it meaning theft is of no use without the corresponding private key[6].

3.3 Types of Cryptography:

➢ **Symmetric Key Cryptography (Secret Key Cryptography)**:

This type of cryptography uses a key for encrypting and decrypting the plain. The only condition here is that it shares the same key for the encryption and decryption and it also consumes less execution time [7].

➢ **Asymmetric Key Cryptography (Public Key Cryptography)**:

This scheme uses two keys named as a private key and public key. The public key is provided by the receiver to the sender to encrypt the message while the private key is applied by the receiver itself to decrypt the message[6].

4 Steganography:

4.1 Steganography Definition:

The word steganography is derived from the greek word ‘stegos’ meaning ‘to cover’ and ‘graphia’ meaning ‘writing’, thus tanslating to ‘hidden writing’, Steganography is the art and science of writing hidden messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message, a form of security through obscurity, but is not only the art of hiding data but also hiding the fact of transmission of secret data. More generally, is the technique of hiding secret data within an ordinary, file or message in order to avoid detection[3].

Steganography can be used to conceal any type of digital content, including text, image, audio or video, the data to be hidden can be hidden inside almost any other type of digital content.

Steganography is an encryption technique that can be used along with cryptography as an extra-secure method in which to protect data. At any rate, steganography protects from pirating copyrighted materials as well as aiding in unauthorized viewing.

4.2 Steganography Modes

There are two modes of steganography:

4.2.1 Linguistic Steganography:

The literature on linguistic steganography, in which the linguistic properties of a text are modified to hide information, is weak compared to other media. The likely reason is that it is easier to make changes to non-linguistic media in which the secret message will be undetectable by an observer [4].

4.3 Types of Steganography:

- **Text Steganography:**

It consists of hiding informat ion inside the text files. The techniques in text steganography are number of tabs, white spaces, capital letters. In this method, the secret data is hidden behind every nth letter of every words of text message. Numbers of techniques are available for hiding information in text file[1]. These techniques are i) Format Based Method; ii) Random and Statistical Method; iii) Linguistics Method[8].

- **Image Steganography:**

Hiding the data by taking the cover object as image is referred as image steganography. In this technique pixel intensities are used to hide the information. In digital steganography[1] , The 8 bit and 24 bit images are common .The image size is large hides the more information[9] , the methods for hiding data in image are i)LSB ; ii) Masking ; iii) filtering.

- **Audio Steganography:**

It involves hiding data in audio files[1]. It is very important medium due to voice over IP (VOIP) popularity. There are different methods of audio steganography.It is used for digital audio formats such as MIDI, AVI and WAVE for steganography[10]. This method hides the data in AU and MP3 sound files. These techniques are i) Low Bit Encoding; ii) Phase Coding and echo hiding; iii) parity coding.

- **Network Steganography:**

It involves hiding the information by taking cover objects as network protocol such as

ICMP, TCP, UDP, IP etc [9], where protocol is used as carrier is called network protocol steganography. In the OSI model there exist covert channels where steganography can be used[11].

- **Video Steganography:**

It is a method of hiding any data or any type of files into digital video format. video (combination of pictures) is used as carrier for hiding the data[9]. The discrete cosine transform i.e. DCT change the values which is used to hide the information in each of the images in the video, which is not justified by the human eye. The formats used by video steganography are Mp4, MPEG etc[1].

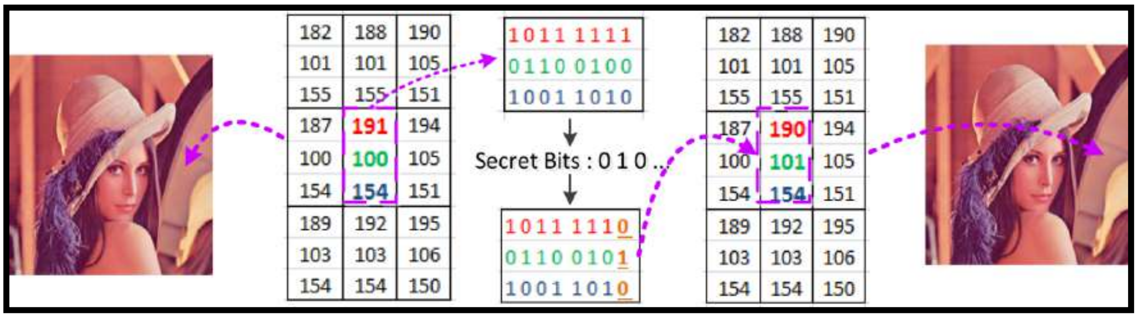
4.4 Techniques of Steganography:

4.4.1 Spatial Domain Methods:

In this method the secret data is embedded directly in the intensity of pixels [7] of cover image in such a way that the effect of message is not visible on the cover image. It means some pixel values of the image are changed directly. Spatial domain techniques are classified into following:

- **Least Significant Bit (LSB):**

LSB is one the method of spatial domain methods. this method is most commonly used for hiding data [7]. By replacing the least significant bits of image pixels with the bits of secret data, the embedding is done. The image obtained after embedding is almost similar to original image because changes in the value of the LSB are imperceptible for human eyes does not bring too much differences in the image. LSB works well where priorities are equal in both images. If one image has more room than the other it results in lower quality. The basic LSB substitution mechanism is shown in Figure 3. This section discusses the major variations and combination of LSB based methods in last 5 years [3].



182	188	190
101	101	105
155	155	151
187	191	194
100	100	105
154	154	151
189	192	195
103	103	106
154	154	150

1011 1111
0110 0100
1001 1010

Secret Bits : 0 1 0 ..

1011 1110
0110 0101
1001 1010

182	188	190
101	101	105
155	155	151
187	190	194
100	101	105
154	154	151
189	192	195
103	103	106
154	154	150

Figure 1.2: Basic 1-bit LSB Embedding Techniques [8].

- **Pixel Value Differencing (PVD):**

In this method the two consecutive pixels are selected. Whether the pixels are determining from smooth area or an edge area. Payload is determined by calculating the difference between two regular pixels and it serves as basis for identifying whether the two pixels belongs to an edge area or smooth area.

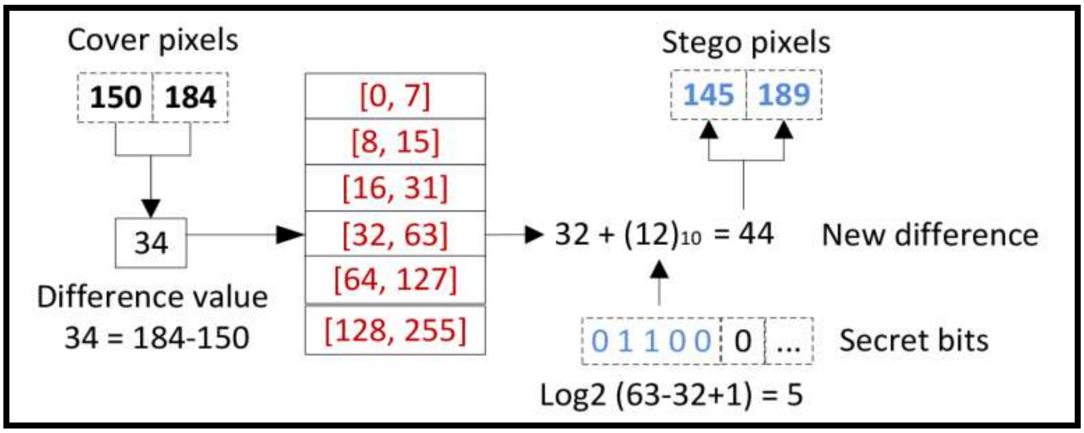


Figure 1.3: General Pixel Values Differencing Embedding Scheme[3].

- **Binary Pattern Complexity (BPC):**

In this segmentation of image are used by measuring the noise factor in the image complexity. Complexity is used to determine the noisy block. The noisy portion is replaced by binary Pattern and it is mapped from the secret data [6].

- **Multi-Base Notation System (MBNS) :**

MBNS is another spatial Domain embedding method based on multiple base notational systems introduced to re-express/transform the secret data into the notational system before embedding process. In MBNS based techniques, secret data is converted into symbols and re-expressed in the multiple-base notational system. These symbols are embeddedinto pixels intensities.

- **Exploiting Modification Direction (EMD) based methods:**

EMD is a well-known embedding method that maintains the high fidelity of stego-images. Generally, the secret digit is transformed by the (2n+1) system during embedding process in the EMD, where is the number of cover pixels. The maximum pixel value of distortion range is just (1). In other words, the EMD utilizes the specific base to determine the local variation of pixel intensity in the image, therefore pixels in high texture areas can embed the more secret message. As results, the EMD can achieve good visual quality when compared with LSB and PVD methods [12].

- **Gray Level Modification (GLM) based methods:**

Another spatial domain strategy is a gray level modification (GLM). These types of embedding methods map the secret data by modifying the gray level of pixels (not hide or embed). Method was based on a mathematical function, where a group of pixels was selected for mapping the secret data inside it. This method utilized the concept of odd and even numbers for mapping. The GLM base methods are simple to implement and have minimum computational complexity. However, these methods were unable to resist steganalysis detection attacks [13].

- **Histogram based methods:**

Histogram based data hiding is another commonly used steganographic technique. The histogram shifting is considered the most efficient histogram based embedding schemes and it has following phases. First, it finds the peak and zero points in a cover image, whereas the bins are shifted with one level between the zero and peak points for emptying peak points. In the second phase, the secret bits are concealed by predefined adjustments in new peak point and the empty point [3].

4.4.2 Transform Domain Technique (Frequency Domain):

It is a more complex way, in this technique; the secret message is embedded in the transform or frequency domain of the cover. The process of embedding data of a signal is much stronger than embedding principles. The different transformations and algorithms are used to hide data in the images. This is a more complex way of hiding message in an image. are broadly classified such as:

i) DFT; ii) DCT; iii) DWT; iv) DCT [11].

- **The Discrete Fourier Transform (DFT):**

DFT is the transform that are purely discrete: discrete-time signals are converted into discrete number of frequencies. DFT converts a finite list of equally spaced samples of a function into the list of coefficients of a finite combination of complex sinusoids ordered by their frequencies [11]. The algorithm for computing the DFT is very fast on modern computers. This algorithm is known as Fast Fourier Transform ‘FFT’.

- **The Discrete Cosine Transform (DCT):**

DCT transform the signal or image from spatial domain to the frequency domain. The mathematical transforms convert the pixels. The DCT is used in Steganography as the Image is broken into 8×8 pixel blocks and transforms these pixel blocks into 64 DCT. Working from left to right, up to down [11].

- **Discrete Wavelet Transform (DWT):**

It is used to transform the image from a spatial domain to the frequency domain. In the process of steganography DWT identifies the low frequency and high frequency information of each pixel of the image. It is mathematical tool used for processing of non-stationary signals. DWT performs in one dimension and in the two-dimensional plane. The DWT is the accurate model than the DFT or the DCT and it is multi resolution description of the image [7].

4.4.3 Spread Spectrum Techniques:

In this method the secret data is spread over a wide frequency bandwidth. The ratio of signal to noise in every frequency band must be so small that it become difficult to detect the presence of the information. Even if parts of data are removed from several bands, there would be still enough information is present in other bands to recover the data. Thus, it is difficult to remove the data completely without entirely destroying the cover. It is a very robust approach used in military communication.

4.4.4 Distortion Techniques:

By technique of distorting the signal, is store the secret data. A sequence of modification is applied to the cover by the encoder. In the decoder phase the decoder calcules the differences between the original and the distorted cover.

4.4.5 Statistical Techniques:

In the technique message is embedded by changing several properties of the cover. It involves the splitting of cover into blocks and then embedding one message bit in each block. The cover block is modified only when the size of message bit is one.

4.4.6 Masking and Filtering:

These techniques are used to hides information by marking an image. These techniques embed the data in the more significant areas rather than hiding it into the noise level. The watermarking techniques are more integrated into the image and it can be applied without the fear of destruction of the image. This method is used for 24-b it and grey scale images.

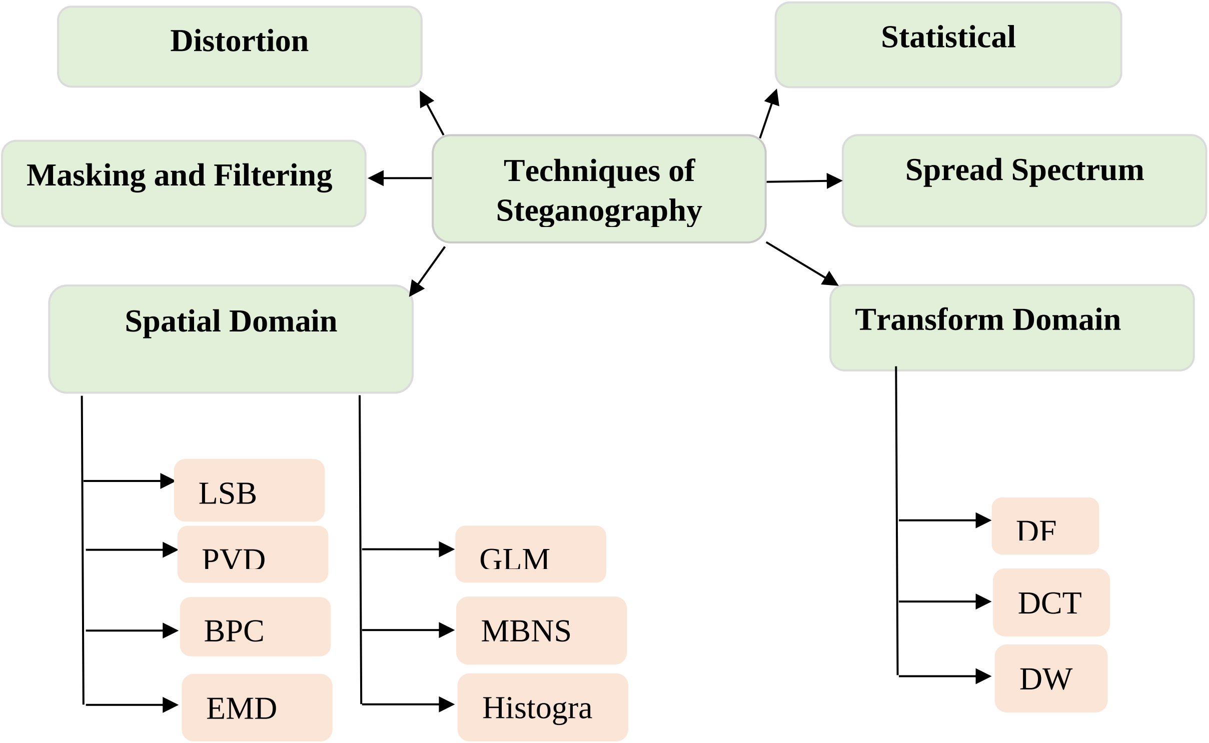


Figure 1.4: Techniques of Steganography.

4.5 Comparison of Spatial and Transform Domains:

Spatial domain directly exploits the cover image data/pixels to conceal the secret information, For example: substitution of secret bits inside pixel value. In contrast transform domain, the data of the cover image is first converted into other signal/form before applying the embedding process. In this table below is a comparison between Spatial and Transform Domains:

Characteristics	**Spatial Domain**	**Transform Domain**
System type	Simple	Complex
Format dependency	Dependent	Independent
Pixel Manipulation	Direct	Indirect
Computational complexity	Less computation times	High computational time
Embedding Capacity	High	Limited

Visual Quality	High	Less controllable
Integrity of visual features	Maintainable	Less maintainable
Robust	Highly prone	Less prone
Security	Vulnerable to geometric attacks	Resistant to geometric attacks
Statistical detection attacks	Easy to expose/detect	Hard to expose/unsuccessful
Non-Structural detection attacks analysis	Easily detectable	Easily detectable
Target (Capacity)	High	Moderate

Table 1.1: Comparison Between Spatial and Transform Domains [3] .

5 Comparison of Steganography and Cryptography:

Steganography and cryptography are both methods used to hide or protect secret data.

1) The meaning of the cryptography signifies “secret writing” while steganography is Signifies “hidden writing”.
2) In cryptography imposes a change on the secret message before transferring it , while in steganography, the main structure of the message is not changed .
3) the cryptography provides confidentiality, integrity, authentication, and non-repudiation. On the contrary, the Steganography provides only confidentiality and authentication.
4) The steganography can be employed on any medium such as text, audio, video and image while cryptography is implemented only on the text file.
5) The degree of the security of the secret data is measured by the key length which makes the algorithm strong and unbreakable. Conversely, there is no such thing in steganography [5].

in this table below is the discord between Steganography and Cryptography:

		Cryptography	**Steganography**
Objective		Protection of content	Hiding the existence of communication and secret data
Characteristics	Perceptual Security	Visible	Invisible
	Security of Communication	Depends on confidentiality of Key	Depends on confidentially of embedding technique
	Robustness	Against complexity of ciphering algorithm	Against detection of existence of secret information
	Key requirement	Mandatory	Application dependent
	Output type	Ciphertext/ plain text	Medium dependent Image/Text/Video
	Detection and Extraction complexity	Detection easy and complex extraction	Both complex
Challenges		Key management, Complexity of encryption algorithm	Embedding capacity High Imperceptibility, Robustness

Table1.2: Comparison Between Information Encryption and Information Hiding.

6 Comparisons of Method Steganography:

In this table a description of the algorithm used to hide information and its advantages and the value of visual quality 'PNSR', Through the table, we can compare the values of 'PNSR', where we notice a rise in the value of 'PNSR' at the authors did that Roy and Changder (2014):

Algorithm	**Reference**	**Approach**	**Advantages**	**Visual Quality (PSNR)**
LSB X-OR	Chakraborty et Al (2013)	LSB-based	Replacement of encryption in stego-system	NA
LSB	That SarreshtedAri and Akhaee (2014)	LSB-based	Simple implementation	52.90 dB
GLSB++	Qazanfari and Safabakhsh, (2014)	LSB-based	Reduced probability of change per pixel.	> 50 dB
Operation LSB	Yuan (2014	LSB-based	Less modification per pixel (mpp).	≈ 50 dB
ALSB-MLEA	Muhammad and Al (2016)	LSB-based	Light-weightagainstencryption.	> 45 dB
LSB-WH	Tavares and Junior (2016)	LSB-based	Based on Word hunt puzzle approach.	NA
MPBDH	Nguyen and Al (2015)	LSB-based	Block based multi-bit plane adaptive LSB embedding.	≈ 46 dB

PVD-TPVD	Lee and Al (2012)	PVD-based	Secret image communication.	≈ 40 dB
Octonary PVD	Balasubramanian and Al (2014)	PVD-based	Adaptivelyregion based embedding.	≈ 40.20 dB
MF-PVD	S. Shen and Al (2015)	PVD-based	Utilized the correlation of R G B channels.	≈ 36 dB
PVD-TPVD	Hernández-Servin and Al (2015)	PVD-based	Eliminate the location map of overflow/underflow for TPVD.	≈ 36.25 dB
Ad-PVD	Swain (2015)	PVD-based	Application based adaptive solutions.	≈ 46.65 dB
PVD-TPVD	Grajeda-Marín and Al (2016)	PVD-based	Skip overflow/underflowproblems.	≈ 38.33 dB
FEMD	Kieu and Chang (2011)	EMD-based	Massive improvement in capacity.	≈ 52 to 31 dB
Ad-EMD	H.-M. Sun and Al (2011)	EMD-based	Improvedembeddingcapacity.	≈ 43 to 34 dB
FFEMD	Wen-Chung and Ming-Chih (2013)	EMD-based	Resolve the overflowproblem.	NA
GEMD	Kuo and Al (2013)	EMD-based	Extraction function: lookup & formal form.	≈ 50.17 dB

MSD	Kuo, Wang and Al (2016)	EMD-based	Reduce the pixel modification ratio (n/2).	> 52 dB
MBEF	Kuo and Al (2016)	EMD-based	Adaptive embedding capacity	≈ 51 to 30 dB
VRNS	Geetha and Al (2011)	MBNS-based	Renowned numerical model.	≈ 41 dB
GMB	Chen and Al (2016)	MBNS-based	Adaptive capacity-based solution.	≈ 50 to 35 dB
GLM-MLE	Muhammad and Al (2015)	GLM-based	High imperceptibility.	≈ 57 dB
APPM	Hong (2013)	PPM-based	Adaptive to visual quality vs payload.	≈ 52 to 35 dB
PPM-PVD	J. Chen (2014)	PPM-based	Random embedding characteristics.	≈ 50 to 42 dB
MP-MPE	Jafar and Al (2015)	Prediction-based	Improvedembeddingcapacity.	≈ 46 dB
RDH-HS-ME	Z. Pan and Al (2015).	Histogram-based	Adaptive approach.	≈ 30-50 dB
RDH-HS	K. Chen and Al (2016)	Histogram-based	Reduce the location map size.	NA
Hybrid-Edge-ALSB	Ioannidouand Al (2012)	Edge-based	High imperceptibility and capacity.	≈ 44 dB

ALSB	W. Tseng and Leng (2014)	Edge-based	Efficient edge detection by the hybrid fuzzy edge detector.	≈ 38.18 dB
Edge-Interpolation	Jung & Yoo (2014)	Edge-based	Improvedembeddingcapacity.	> 35 dB
E-XoR coding	Al-Dmour and Al-Ani (2016)	Edge-based	Simple implementation	> 40 dB
Canny-Huffman	S. Sun (2016)	Edge-based	Improved visual quality and capacity	≈ 60 dB
LCS	Roy and Changder (2014)	Mapping-based	Limited modification in the cover image	NA
Fibonacci	Alan Abdulla and Al (2014)	Mapping-based	Fibonacci embedding reduces the visual distortion effects.	> 50 dB
BI	Swainand Lenka (2012)	Pixel/ Block indicator-based	Simple to implement	> 42.75 dB
PI-zigzag	Mahima hand Kurinji (2013)	Pixel/ Block	Multi-mode of indicators.	> 50 dB
HIS-M-LSB-SM	Khan and Al (2015)	Color model	Reduces processing time.	~47.93 dB

ALSB	Khan and Al (2016)	Color model	Image scrambling using a light-weighted image scrambler.	~52.45 dB
GA	Kanan and Nazeri (2014)	ML-based	Adaptive embedding.	≈ 34-55 dB
ANN-MPSO	El-Emam (2015)	ML-based	Improved capacity and visual quality.	> 55 dB
ANN-PSO-GA	El-Emam and Al-Diabat (2015)	ML-based	Hybrid utilization of ANN with GA	> 50 dB
ANN-GA GA-Chaotic	Doğan (2016)	ML-based	Chaotic map improved GA-based hiding.	> 52 dB
PVD-ALSB	Jung (2010)	Hybrid	Simple implementation.	≈ 36.28 dB
OPAP-FPVD	Liao, Wen and Zhang (2011)	Hybrid	Four pixels difference employing.	≈ 39.11 dB
SPVD-OPAP	M Khodaeiand Faez (2012)	Hybrid	Improved embedding payload.	≈ 38 dB
LSB-PVD	Y. Tsai, Chen and Chan (2014)	Hybrid	Resolve the overflowproblem.	≈ 35.64 dB
RDH-DE-IN-HS	Lu and Al (2014)	Hybrid	No peak point searching.	≈ 33 dB
IEMD-PVD	S. Shen and Huang (2015)	Hybrid	Resolve the overflow/underflow problem.	≈ 42.46 dB

MPE-LSB-EMD	K. Wu and Al (2015)	Hybrid	Application adaptive solution.	> 35 dB
PI-LSB-PVD	Das and Kar (2015)	Hybrid	Color channel-based indicator.	> 39 dB
ALSB-RMDR	Hussain and Al (2016)	Hybrid	Simple implementation.	39 dB
Directional-PVD-ALSB	Swain (2016)	Hybrid	Integration of ALSB with PVD.	≈ 40.44 to 39.29 dB

Table 1.3: Comparison Between Method Steganography [3].

7 Genetic Algorithms (GA):

7.1 Definition of GA:

The genetic algorithm is defined as an intelligent algorithm that can be used to find and solve complex problems and improve them, as it is one of the efficient research methods based on the principles of natural selection and genetics, invented by the Netherlands in 1975 at the University of Michigan, where he published many researches in this field, and the main goal was Including building and improving many algorithms, software, and systems using these algorithms.[15]

Genetic algorithms are a type of optimization algorithm, meaning they are used to find the optimal solution(s) to a given computational problem that maximizes or minimizes a particular function. Genetic algorithms represent one branch of the field of study called evolutionary computation [48], in that they imitate the biological processes of reproduction and natural selection to solve for the test' solutions [16].

7.2 Characteristics of Genetic Algorithm:

- ❖ Genetic Algorithm is based on the concept of natural selection and natural genetics.
- ❖ These are easy to understand and implement.
- ❖ These are extensively used in optimization problems.
- ❖ These work on population of points rather than an individual point.

- These use probabilistic transition rule instead of deterministic rules.
- Genetic Algorithms can effectively deal with large number of variables.
- These algorithms do not require any derivative information.
- hese are more effective for complex problems than simple problems.
- These provide solution quickly as compared to other traditional optimization techniques.

  [17]

7.3 Genetic parameters:

Genetic parameters are the basic step of the genetic algorithm, and it is a fixed step that differs in the way it is formulated and applied according to the issue or its field of application, as this step is linked with each other, and the algorithm cannot be applied if all these steps are not applied or the algorithm is not lost. Genetic value and usefulness in finding and calculating the solution.

A genetic algorithm consists of three principal operations:

7.3.1 The Selection Operation:

Selection is one of the important processes of „GA". It is used to select an individual on the basis of its fitness value. The individual chromosomes that have higher fitness values are most probable candidates to be selected for reproduction. The individuals having low values will get little chance of their selection. In simple words the probability of a chromosome to be selected for reproduction is proportional to its fitness value [17].

7.3.1.1 Type of Selection:

- Tournament selection.
- Roulette wheel selection.
- Proportionate selection.
- Rank selection.
- Steady state selection.[18]

7.3.2 The Crossover Operation:

The crossover operation generates the off spring from two chosen individuals in the population, by exchanging some bits of the two individuals. The offspring will then inherit some characteristics of their parents.[19]

It is a technique used to combine the individual chromosomes which produces a new chromosome. The offspring generated by crossover is supposed to have the features of both of the parents chromosomes. The objective of crossover operation is to find better solution from good solution. It is also called recombination operator. It selects a pair of individual chromosomes (Parents) for mating. First of all, a location for crossover is selected, and then bits or characters of the selected parents following marked location are swapped. Crossover operation is used to avoid duplication of parents used in recombination. The off springs generated by crossover operation may have either higher fitness or in some cases lower fitness than their parents. However a high fitness offspring is desirable from crossover operation. Crossover operator can be implemented by using several techniques.[17]

7.3.2.1 Type of Crossover:

- Single-point crossover.
- Two-point crossover.
- Uniform crossover.
- Flat crossover.

[20.] see the figure (4):

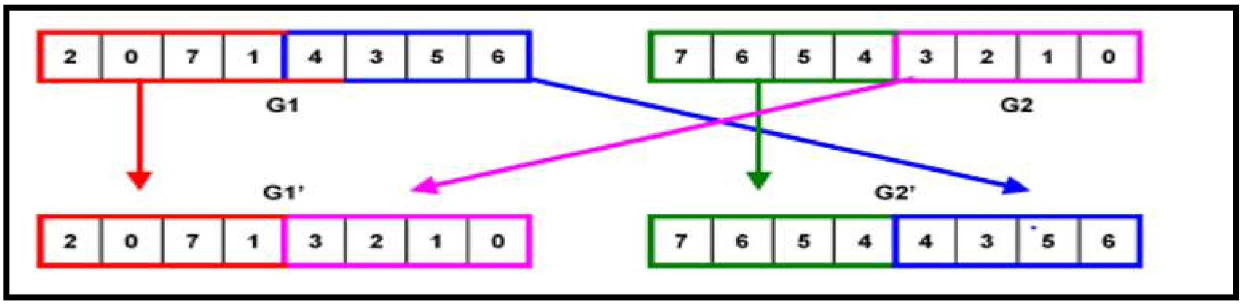


Figure 1.5: Crossover Uniform Generating the Gl' and G2 [21]

7.3.3 The Mutation Operation:

The mutation operation generates the off spring by randomly changing one or several bits of individuals the offspring may then possesses different characteristics from their ascendants. Mutation can then avoid local search in the searching space and increase the probability of finding the global optimum.[17]

7.3.3.1 Type of Mutation:

- Insert Mutation.
- Inversion Mutation.

- Scramble Mutation.
- Swap Mutation.
- Flip Mutation.
- Interchanging Mutation.
- Uniform Mutation.
- Creep Mutation.[22]

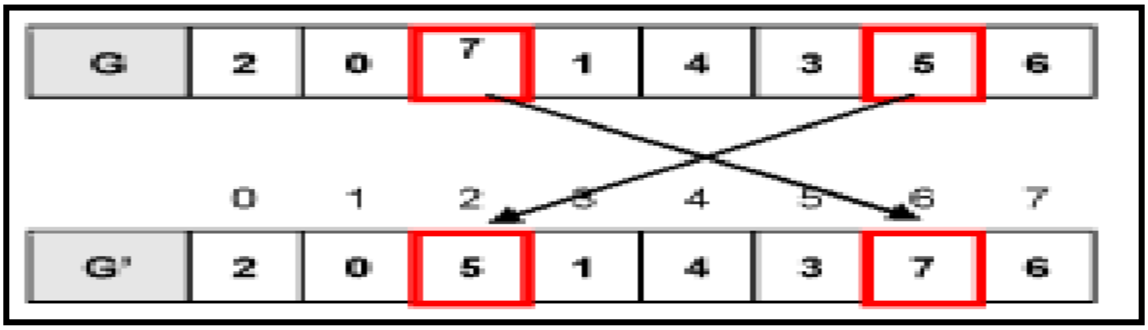


Figure 1.6: Mutation of an Individual G [21].

8 Previous Works of Genetic Algorithms (GA) in Hide Data:

Authors	Use of GA
[22]	In 2010Use the genetic algorithm for modified the pixel values of the steg-image and to keep their statistic characters. Thus, the existence of the secret message is hard to be detected by the RS analysis. Meanwhile, better visual quality can be achieved by the proposed algorithm. The experimental results demonstrate the proposed algorithm's effectiveness in resistance to Steganalysis with better visual quality.
[23]	In 2011Use the genetic algorithm in hiding, which is that which contains the optimum value for the largest PSNR (one of its values), which means a small standard deviation
[1]	In 2013Use the genetic algorithm for to apply the idea to hide image message, using the least significant bit algorithm inside an image and encrypt it in a new way for encryption. For the purpose of increasing security of the access of the letter it is being encrypted to hide the message before using the genetic algorithm to generate random numbers employed in the process of concealment, for the highest extent of randomness. This in turn increases the strength of encryption and concealment.

[24]	In 2016Use the genetic algorithm for pixel assortment of image where data is to be concealed so that detection of clandestine information become multifarious. And using AES cryptographic algorithm for compressed data is transformed into cipher text and then the encrypted data is concealed in the image.
[25]	In 2016Use the genitic algorithm for the best way used to propose system is achieved by Least Significant Bit (LSB) with Visual Cryptography (VC) and. Genetic Algorithm basic function is to modify the pixel location of stego image and the detection of this message is complicated.
[26]	In 2017 Use the genitic algorithm for secure data transmission this work presented a system which uses genetic algorithm, steganography along with visual cryptography. for encryption where plain text is converted into cipher text then it is hidden in LSB pixels of an image using Steganography technique.
[27]	In 2018GA is used to generate random key that represents the best ordering of secret (image/text) blocks to be hiding in the cover images.
[28]	In 2019 Use the genitic algorithm to hide the data for evaluation the performance of different orders for LSB matching and searches for a near-optimum solution among all the permutation orders.
[18]	In 2020Use the genitic algorithm to hide the data and working at the expense of Peak Signal to Noise Ratio (PSNR) for each section after the steganography and then get the best PSNR value of the optimal section (ie, a better distribution of the random sites).

Table 1.4: Table Showing Previous Studies of Genetic Algorithms.

9 Conclusion:

Steganography is not intended to replace cryptography but rather to supplement it. If a message is encrypted and hidden with a steganographic method it provides an additional layer of protection and reduces the chance of the hidden message being detected.

Steganography is still a fairly new concept to the general public although this is likely not true in the world of secrecy and espionage. With continuous advancements in technology it is expected that in the near future more efficient and advanced techniques in steganalysis will merge that will help to better detect illicit materials transmitted through the Internet.

CHAPTER II: DIRECT IMAGE PROJECTION BASED STEGANOGRAPHY TECHNIQUE USING GENETIC ALGORITHM

1 Introduction:

Due to the disadvantages of the LSB method, this chapter developed the hiding method in the images where a new method of hiding is proposed: Projection and image hiding ideally by the genetic algorithm, in this chapter we discuss to the algorithms, the benefits, and the schemes of the LSB, Projection, and GA method.

2 LSB Method:

The most common and most common LSB method is used in Steganography.

2.1 LSB Overview:

Least significant bits (LSB) insertion is a simple approach to embedding information in image file. The simplest steganography techniques embed the bits of the message directly into least significant bit plane of the cover-image in a deterministic sequence. Modulating the least significant bit does not result in human-perceptible difference because the amplitude of the change is small [29]**.**

The LSB based technique is mainly uncomplicated and simple approach through which message bits are embedded within the least significant bits of cover image. In the LSB steganography method and for the purpose of covering the secret messages, the least significant bits of the cover-image are exploited. Thus, this method is considered one of the most common techniques that include the standard LSB replacement [30].

Consider the following cover photo and the secret letter in parts. LSB substitution alternates the last bits of the jacket image as each bit belongs to the messages to be hidden. The following example is to show the method for replacing the standard LSB.

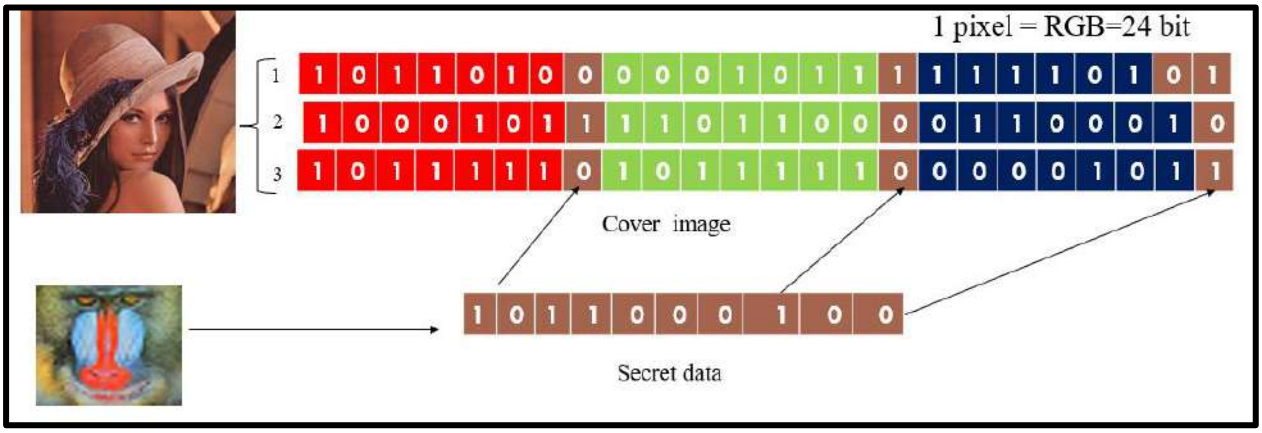


Figure2.1: Hiding with LSB Method.

2.2 Implementation of Hiding Process:

To implement this method, we must go through a set of stages:

1. Read the cover image(C).
2. Converting the cover image into an matrix containing the pixels in binary.
3. Read the secret message (S).
4. Converting a secret message (S) into a binary message ('S).
5. Hide the secret message in the cover image by using the hide method in The least significant bit for LSB.
6. Display the image after hidin(stego_image).

2.3 LSB Hiding Algorithm:

This algorithm shows how to work with LSB:

Algorithm: ‘Image-HideLSB’

Input: Cover: Image(C), Secret: Image(S);

Matrix: Extract pixels from the cover image and convert to binary and position in a matrix(M).

Vector: Extract pixels from the secret information and convert them into binary And put them in a beam(V).

```
Begin
For (i=0 Until   length of V, Do)
    For (k=0   Until Hight of M, Do)
             For (j=0   Until    Width of M, Do)
                    M[k][j] ←——— V[i] ;
```

```
            Fin pour ;
        Fin pour
Fin pour ;
End.
```

2.4 Graph of LSB Function in Hide:

This diagram explains how the algorithm works from beginning of hiding to Get a picture after hiding:

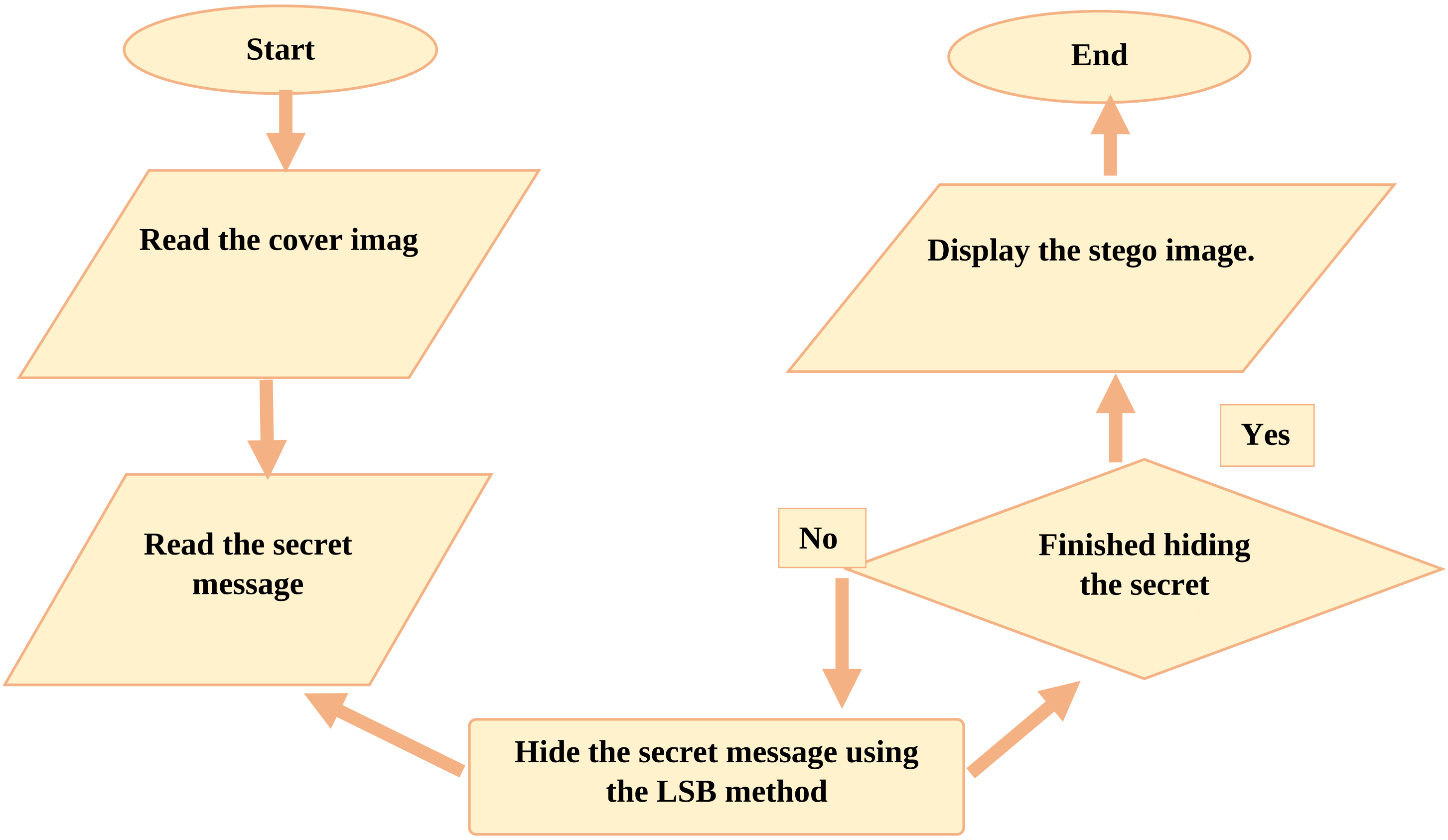


Figure 2.2: Diagrame of LSB hiding.

2.5 Implementation of Extraction:

To implement this method, we must go through a set of stages:

1. Read the image after hiding (stego image).
2. Converting the stego image into a matrix containing the pixels in binary.
3. Retrieve the confidential message(S) from the stego image using the retrieval method from the least significant bit for LSB.
4. Display the confidential message(S).

2.6 Algorithm of Extraction:

This algorithm shows how to work with:

Algorithm: 'Image-ExtractionLSB'

Input: Stego: Image(C'), Secret: Image(S);

Matrix: Extract pixels from the stego image and convert to binary and position in a matrix(M).

Vector: Holds confidential information pixels(V).

```
Begin
For (k=0 Until Hight of M, Do)
        For (j=0 in Width of M, Do)
            For (i=0 in    length of V, Do)
              V[i] ← M[k][j] ;
            Fin pour ;
        Fin pour
Fin pour ;
End.
```

2.7 Graph of LSB Function in Extraction:

This graph shows how the algorithm works from the beginning of the extract to the end of the secret message:

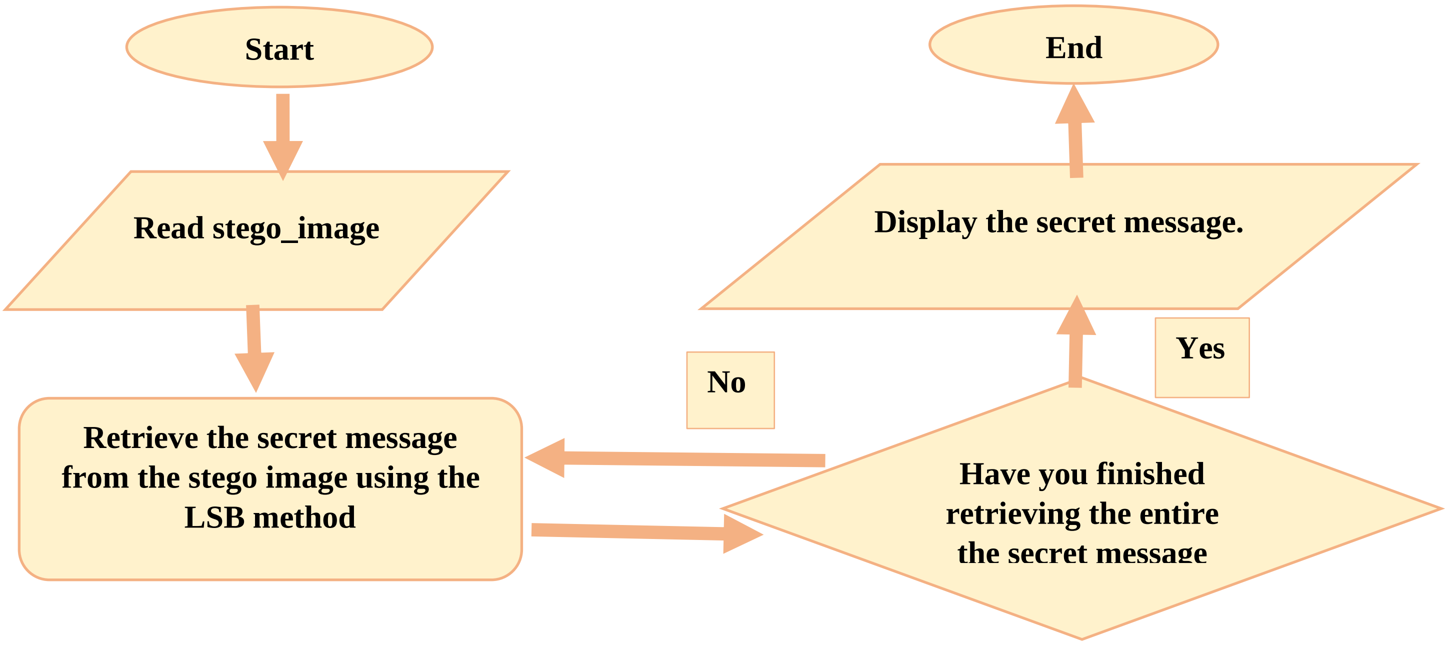

Start
Read stego_image
Retrieve the secret message from the stego image using the LSB method
Have you finished retrieving the entire the secret message
No
Yes
Display the secret message.
End

Figure2.3: Diagram of LSB Extraction.

2.8 Benefits of LSB Method:

Easy to implement.

Data can be extracted quickly.

Has high concealment capability [31].

The advantage of LSB hiding sits simplicity.

LSB embedding technique also allows high perceptual transparency [32]

2.9 Drawback of LSB Method:

Using encapsulation and extraction algorithms using the LSB method, anyone can easily extract the confidential message just like an original message.

Poor security [31].

3 Projection:

In our algorithm, we have proposed a new idea of pixel substitution of the image whose purpose is to hide them. It is based on the projection of each pixel of the original image onto another virtual image that appears as a rotated image. This rotation is explained by a virtual observer who is supposed to rotate around the center of the original image. Thus, the relationship between these two images is ensured by the fact that the plane of the virtual image is perpendicular to the observer direction. Thus, and because each pixel has a luminous ray, the position of intersection of its direction with that of the virtual image is used as the projection position.

This projection has, however, a disadvantage so that some parts of the original image disappear, while others are projected on the same positions. The reason is that the positions of the projected pixels must be rounded to the nearest integers.

3.1 Direct Projection Positions:

It is assumed that the original image is viewed from a position in a direction perpendicular to its centre, while its projection is viewed from an inclined position. First of all, the different positions of the observer are supposed to be located on a circle around the center of the image.

- ➢ The radius of this circle is what defines the distance from the center of the image. The reference position is then that where the direction of the observer is perpendicular to the plane of the original image.
- ➢ Consequently, any other position is defined by its angle of inclination relative to the reference position.

- The projection process is based on the intersection of the direction of the light beam towards the observer of each pixel of the original image and the virtual image whose plane is perpendicular to the current direction of the observer. The point of intersection is the position to be used to paste the original pixel.
- Moreover, it is clear that the new position is slightly different from the original one. Thus, when all pixels are projected, and depending on the angulation, the resulting image is either expanded or condensed from the original (Figure4).

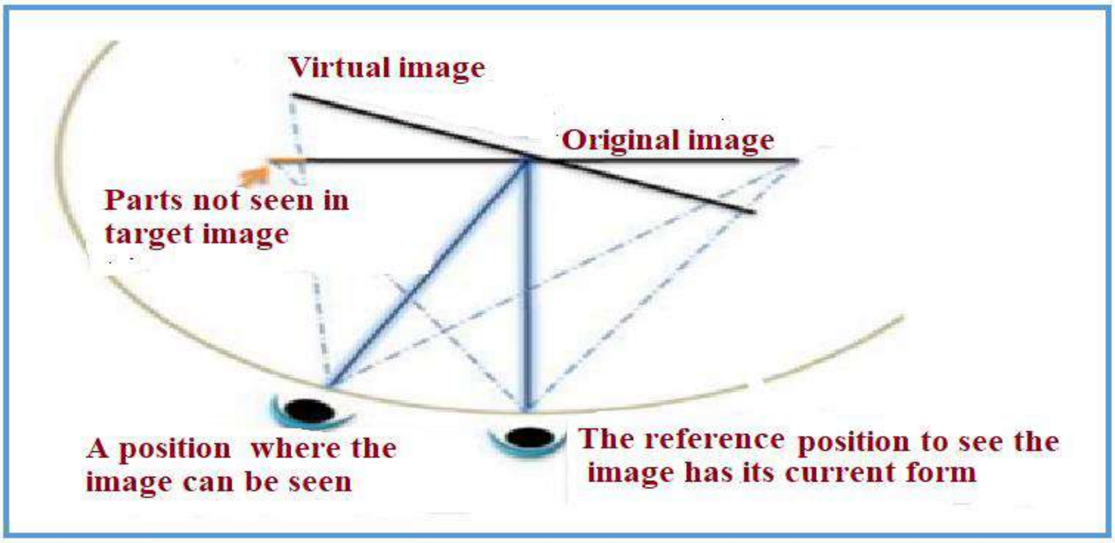

Virtual image
Original image
Parts not seen in target image
A position where the image can be seen
The reference position to see the image has its current form

Figure 2.4: System Projection Image.

- Consider the pixel located at Po of the center O of the image and seen from the reference position Vo, When the observer moves around the center of the image along the circle having its radius equal to → |OVo|, the position that the pixel Po will take in the image target becomes Pt. In other words, Pt is the position where the pixel Po is supposed to be seen in the target image when the observer position is tilted. So, the objective is to determine the value of Opt (Figure5).

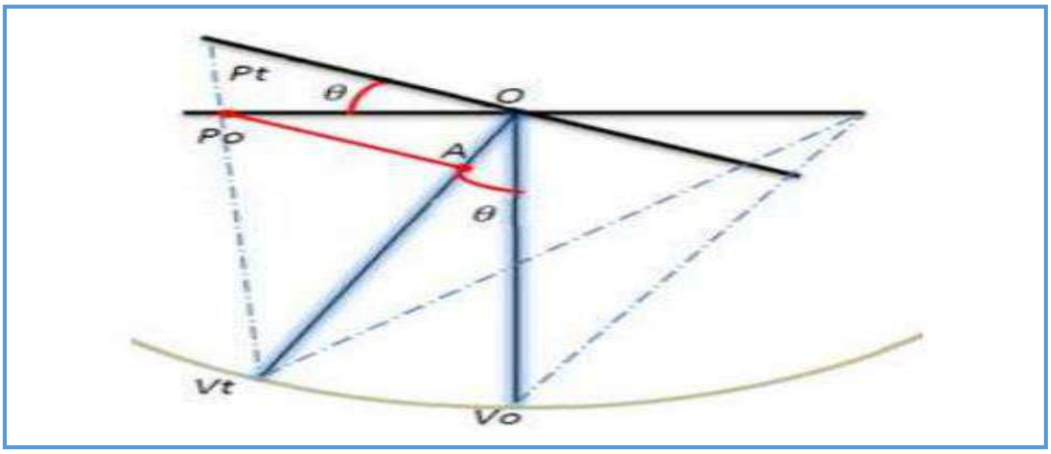


Figure 2.5: Estimate the Position of the Projection of a Pixel.

- Let θ be the angle between the directions of the reference position and the new position. This angle is the same between the plane of the original and the inclined image.
- Let A the perpendicular projection of Po on the new direction→ |OVt| ṅ as ΔPoAO is at aright triangle and OPoA =θ.
- It is easy to see that APo = OPo*cosθ. If you take in account ΔOPtV, using this expression and based on the truth of the expression following:

$$\frac{\mathrm{OPt}}{\mathrm{APo}} = \frac{\mathrm{OVt}}{\mathrm{AVt}} \tag{1}$$

- The value of OPT can be easily calculated from the resulting expression:

$$OPt = \frac{\mathrm{OVt}*\mathrm{OPo}*\cos\theta}{\mathrm{OVt}-\mathrm{OPo}*\sin\theta} \tag{2}$$

Because : AVt = OVt - OA.

- According to figure (5) the formula (2) is true only for a part that is half the image on the same side of the observer's position relative to the original. In fact, the other part of the image is projected onto the target image using formulas different.
- Since the expected results in the substitution algorithm do not depend on the complexity of the formulas or their number, only formula (2) was used. Thus, it is used simultaneously on both sides of the image. Also, instead of using each pixel alone, the construction of the new virtual image uses a whole line ora whole column as a processing unit. So, the formula (2) is used to evaluate the new position of a pixel, but uses a whole row or column of a color plane to copy it into that position.
- In formula (2), the reconstruction of the virtual image requires two main parameters which are the distance OVo = OVt of the observer and the side it/he takes in relation to the reference position (angle θ). Also, and because the algorithm transforms the image in horizontal and vertical directions, the distance OVt, and the angle θare both composed of two components representing horizontal displacement (OVtx and θx) and vertical displacement (OVty and θy). This means that the observer can be moved from its original position in the horizontal and/or vertical direction.
- Here is an example of a projected image where the white pixels are the intersection of the horizontal and vertical vector positions.

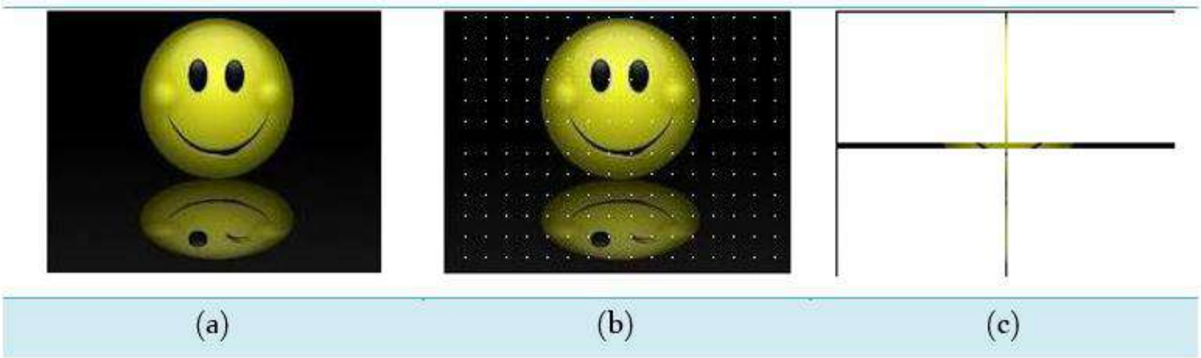


Figure2.6:(a) Original Image, (b) and (c) Projected Positions from (a).

- When constructing pixel positions two problems arise. The first is that of pixels with their projection positions outside the boundary of the virtual image. For example, Figure 3 in image (c) shows positions or holes that are not white. This means that they are not affected by the projection.
- The second problem appears and is due to the fact that each estimated position of OPt is rounded to the integer closest to its value, which allows to obtain several rows or columns of pixels having the same projection position. This occurs in the case where large angles are used that condense a whole part of the original image in a reduced portion of that projected. So, for a given position, we keep only one row point or column point and the others are ignored.
- The projection algorithm repeats the same operation on both sides of the image. This means that the projection is applied to the other side as if the virtual observer is moved to the left side and then to the right side.
- The idea of our algorithm does not require recovery of lost positions. Therefore, its two problems do not need to be fixed. But despite all this, it is very easy to detect the difference in the use of small or large angles.

3.2 Implementation of Hide:

To implement this method, we must go through a set of steps:

1. Read the cover image (C) .
2. Apply a projection method to your cover image (calculation OPT).
3. Converting the default cover image resulting from the projection method into matrix that contains pixels in binary.
4. Read the secret message(S).
5. Converting a secret message (S) into a binary message ('S).

6. Hiding the secret message in the default image resulting from the Projection process using the LSB masking method in the least significant bit.
7. Display the image after hidin(stego_image).

3.3 Algorithm of Hide:

The algorithm shows how the projection method works in hiding:

Algorithm: 'Image-Hide_Projecion'

Input: Cover: Image(C), Secret: Image(S);

Matrix: Extract pixels from the cover image resulting from the projection process and convert to binary and position in a matrix(M).

Vector: Extract pixels from the secret information and convert them into binary and put them in a beam(V).

```
Begin
For each Opo position (column/row) of the original image
Estimate Opt;
If Opt is not outside the limits of the target image then
If Opt is not yet used then
the Opt column/row of the target image;
Marked the use of Opt;
For (i=0   Until length of V, Do)
   For (k=0 Until Hight of M, Do)
        For (j=0   Until Width of M, Do)
              M[k][j] ←——— V[i];
        Fin pour ;
     Fin pour ;
Fin pour ;
End.
```

3.4 Diagram of Projection Function in Hide:

This diagram explains how the projection method follows in hiding:

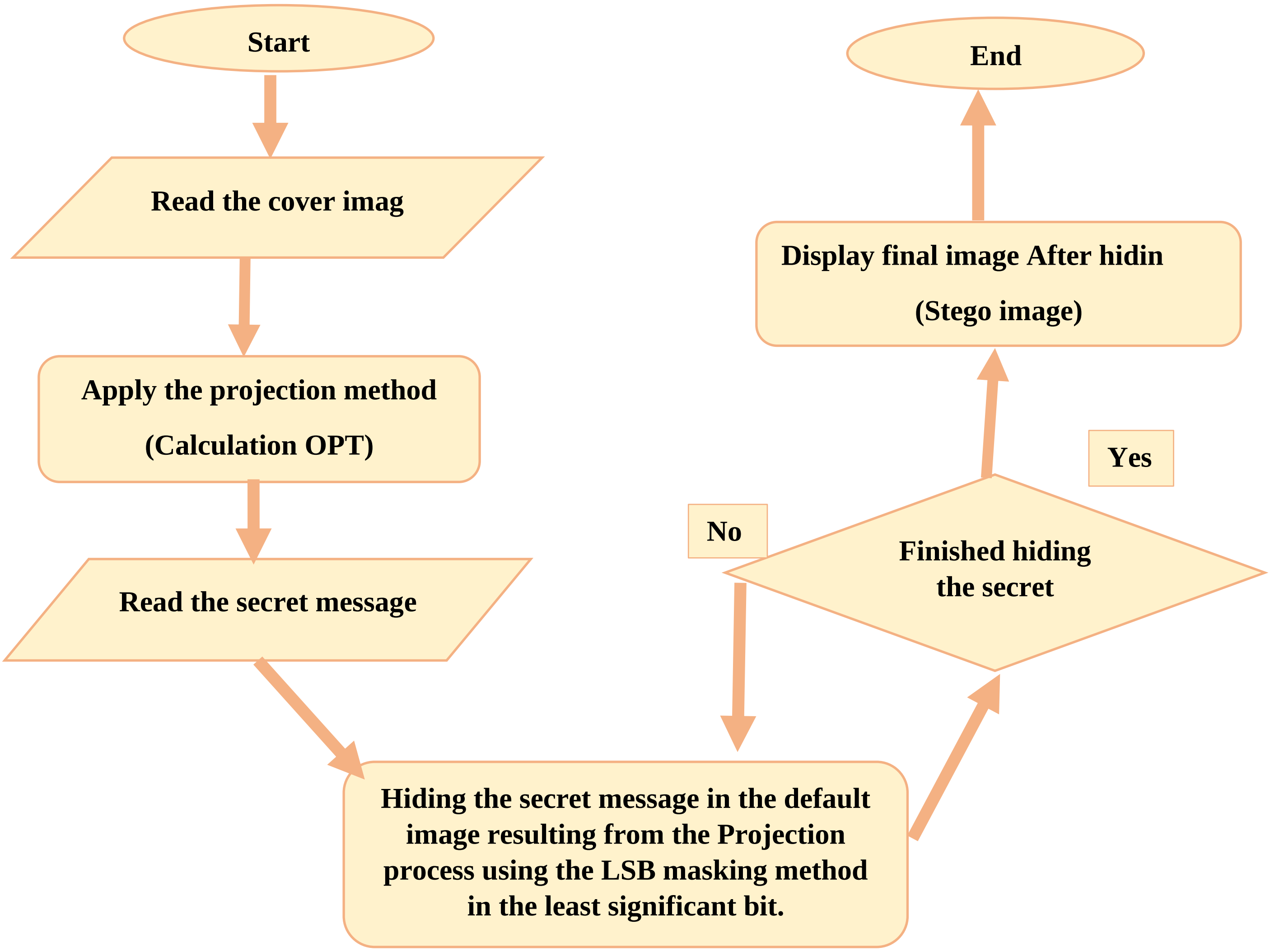

Start
Read the cover imag
Apply the projection method
(Calculation OPT)
Read the secret message
Hiding the secret message in the default image resulting from the Projection process using the LSB masking method in the least significant bit.
No
Finished hiding the secret
Yes
Display final image After hidin
(Stego image)
End

Figure 2.7: Diagram of Projection Hiding.

3.5 Implementation of Extraction:

To implement this method, we must go through a set of steps:

1- Read the image after hiding (stego image).

2- Extract the key for the projection process.

3- Retrieve the pixels caused by the projection process from the image after concealment and placing them in a matrix in the binary.

4- Recover the secret message (S) from the pixels caused by the projection process using the retrieval method from the least significant bit for LSB.

5- Display the confidential message(S).

3.6 Algorithm of Extraction:

The algorithm shows how the projection method works in hiding:

Algorithm: 'Image-Extraction_Projecion'

Input: Cover: Image(C), Secret: Image(S);

Matrix: Extract pixels from the stego image resulting from the projection process and convert to binary and position in a matrix(M).

Vector: Holds confidential information pixels(V).

```
Begin
Extract the key for the projection process.
   For (k=0   Until Hight of M, Do)
        For (j=0 Until Width of M, Do)
               For (i=0 Until length of V, Do)
                         V [i] ← M[k][j] ;
         Fin pour ;
      Fin pour ;
Fin pour ;
End.
```

3.7 Graph of Projection Function in Extraction:

This diagram explains how the projection method follows in extraction

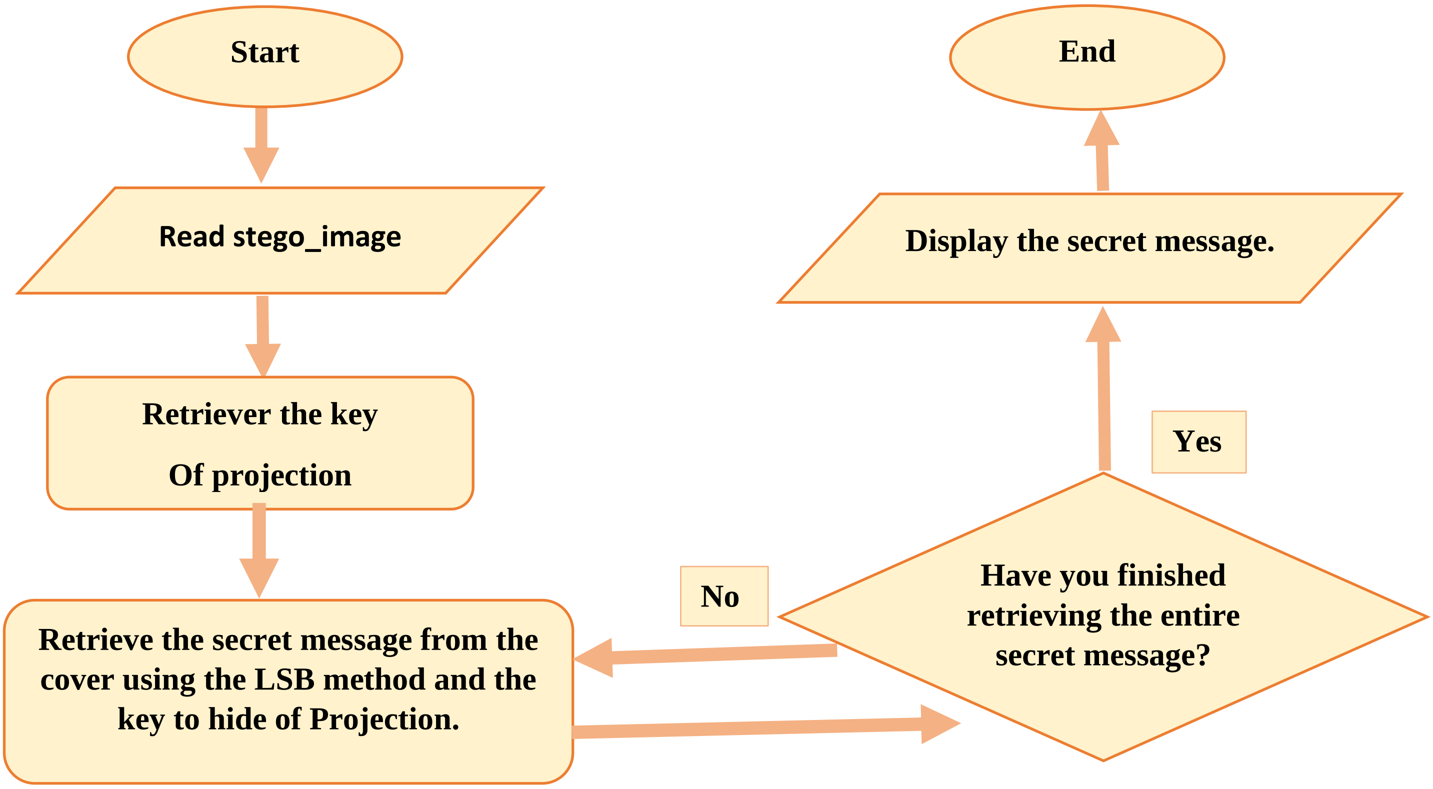

Start
Read stego_image
Retriever the key
Of projection
Retrieve the secret message from the cover using the LSB method and the key to hide of Projection.
No
Have you finished retrieving the entire secret message?
Yes
Display the secret message.
End

Figure2.8: Diagrame of Projection Extraction.

3.8 Benefits of Projection Method:

Increases the confidentiality of the information**.**

Confidential data is difficult to extract.

She has high hiding ability.

3.9 Drawback of Projection Method:

The cover image is distorted which raises doubts about the existence of information hiding.

has two main problems that are appeared from using high-low values of distance and angle. Those values result on non-using of certain pixels location.

4 Genetic Algorithm:

Genetic algorithms are often used in the hiding process and the purpose of This is to keep the information secret and improve the concealment.

4.1 Working of Genetic Algorithm:

GA starts its working from a set of solutions rather than a single solution. The set of solutions is called population. The initial population is generated randomly. Each solution of the problem is adequately represented by encoding a string (Chromosome) of bits or characters (Genes). Every chromosome has a fitness value associated with it.

The collection of chromosomes with their corresponding fitness values is called population. The population at a particular instance is called generation.

Fitness function is one of the major decisive parameters of „Genetic Algorithm". It defines the objective of the problem to be optimized. A pair of chromosomes based on their fitness values is used to reproduce off springs.

The genetic properties of both the chromosomes are intermixed to generate better offspring, such a mechanism is called crossover. After crossover operation, the genetic characteristics of the generated offspring are further modified. Mutation is a procedure to modify the characteristics of the generated offspring to make it more effective. The algorithm terminates when the required condition is fulfilled.[33]

5 Projection with GA:

In our thesis, we used the genetic algorithms in the projection process To increase the confidentiality of the information.

5.1 Working Principle:

The operation of the genetic algorithm passes by fundamental stages:

- First, we divided the default image that results from the projection process to a group of chromosomes.
- We considered each chromosome containing 40 pixels.
- After that, we create a general population group, a group of chromosomes that we obtain.
- In the general population, we applied the most important processes of the genetic algorithm (crossover, mutation, selection).
- In the crossover process, every time we select 2 random parent chromosomes and apply the process to obtain 2 new child chromosomes and return the children's chromosomes to the population.
- In the mutation process we choose a random location and apply the process to all the chromosomes in the group and in the same position.
- In the selection process we choose the best generation and apply the concealment

- We return these stages until we reach the best generation of the greatest Fitness and the largest PNSR value.

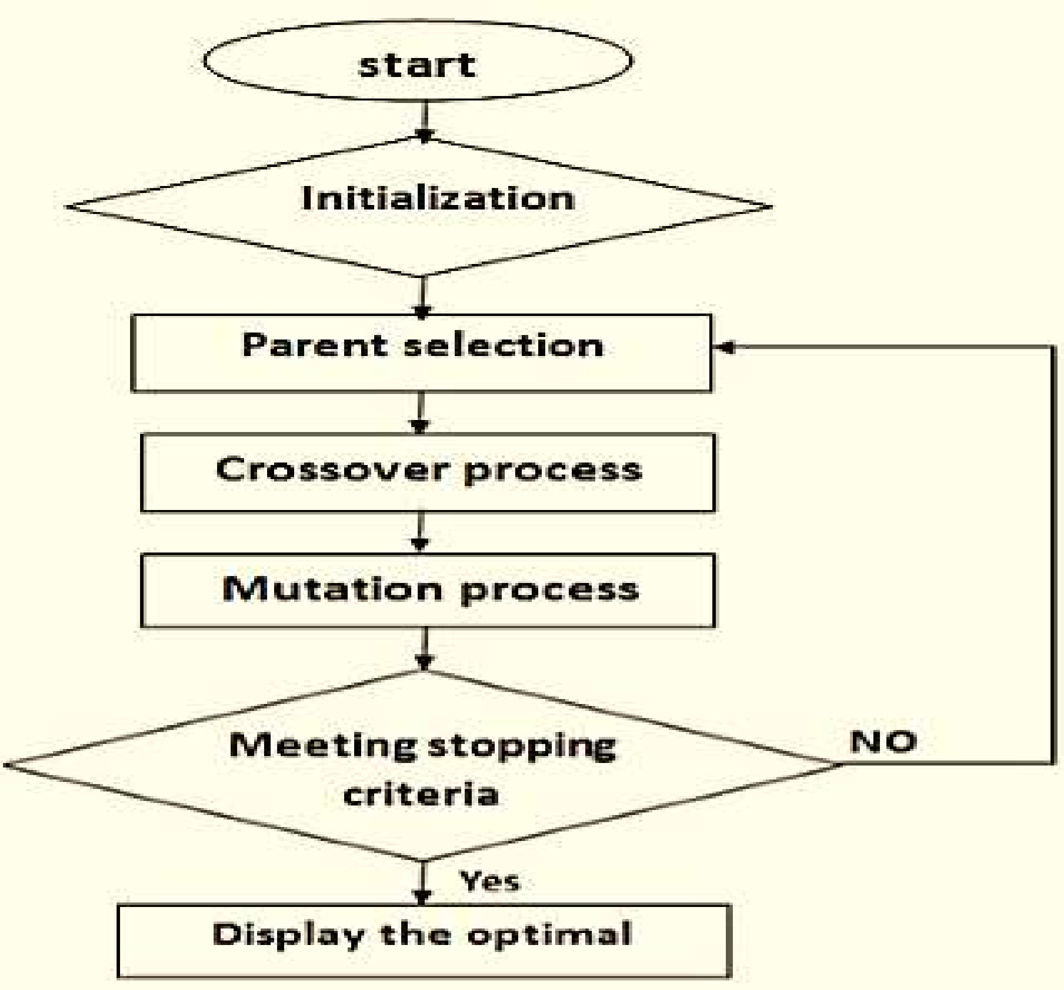


Figure 2.9: Flowchart of Genetic [28].

5.2 Implementation of Hide:

To achieve this, you must follow the steps below from start to finish:

1. Read the cover image (C).
2. Apply the projection method.
3. Read the secret message and convert it into a binary(S).
4. Call the genetic algorithm to determine the best hidden sites.
5. Hide the secret message in The default image resulting from the projection process at sites designated by the genetic algorithm using the Least Significant Bit (LSB) method (The length of the chromosome=24 *40 bit, Primary Generation Length=The length of a multiplication width The default image that results from the projection process).
6. Determine fitness by calculating Peak Signal to Noise Ratio (PNSR) and Mean square error (MSE),:

$$F = \frac{(\alpha * MSE) + (\beta * \text{PNSR})}{F_{max-old}}$$

- $\alpha + \beta = 1$
- F_{max} : The value of the new fitness.
- F_{old} : The value of the old fitness.

7. Display the image after hidin(stego_image) With the final sites of concealment identified by the genetic algorithm, the key is sent to the projection and GA.

5.3Algorithm of Hide:

The algorithm shows how the projection and GA method works in hiding:

Algorithm: 'Image-Hide_ProjectionGA'

Input: Cover: Image(C), Secret: Image(S);

Matrix: Extract pixels from the cover image resulting from the projection process and convert to binary and position in a matrix(M).

Vector: Extract pixels from the secret information and convert them into binary and put them in a beam(V).

```
Begin
For each Opo position (column/row) of the original image
Estimate Opt;
If Opt is not outside the limits of the target image then
If Opt is not yet used then
the Opt column/row of the target image;
Marked the use of Opt;
For (i=0 Until length of V, Do)
   For (k=0 Until Hight of M, Do)
        For (j=0 Until Width of M, Do)
  We call the genetic algorithm and apply it to the matrix(m);
              M[k][j] ←———— V[i] ;
         Fin pour ;
      Fin pour
Fin pour ;
End.
```

5.4 Diagram of Projection with GA Function in Hiding:

This diagram explains how the projection with GA method follows in hiding:

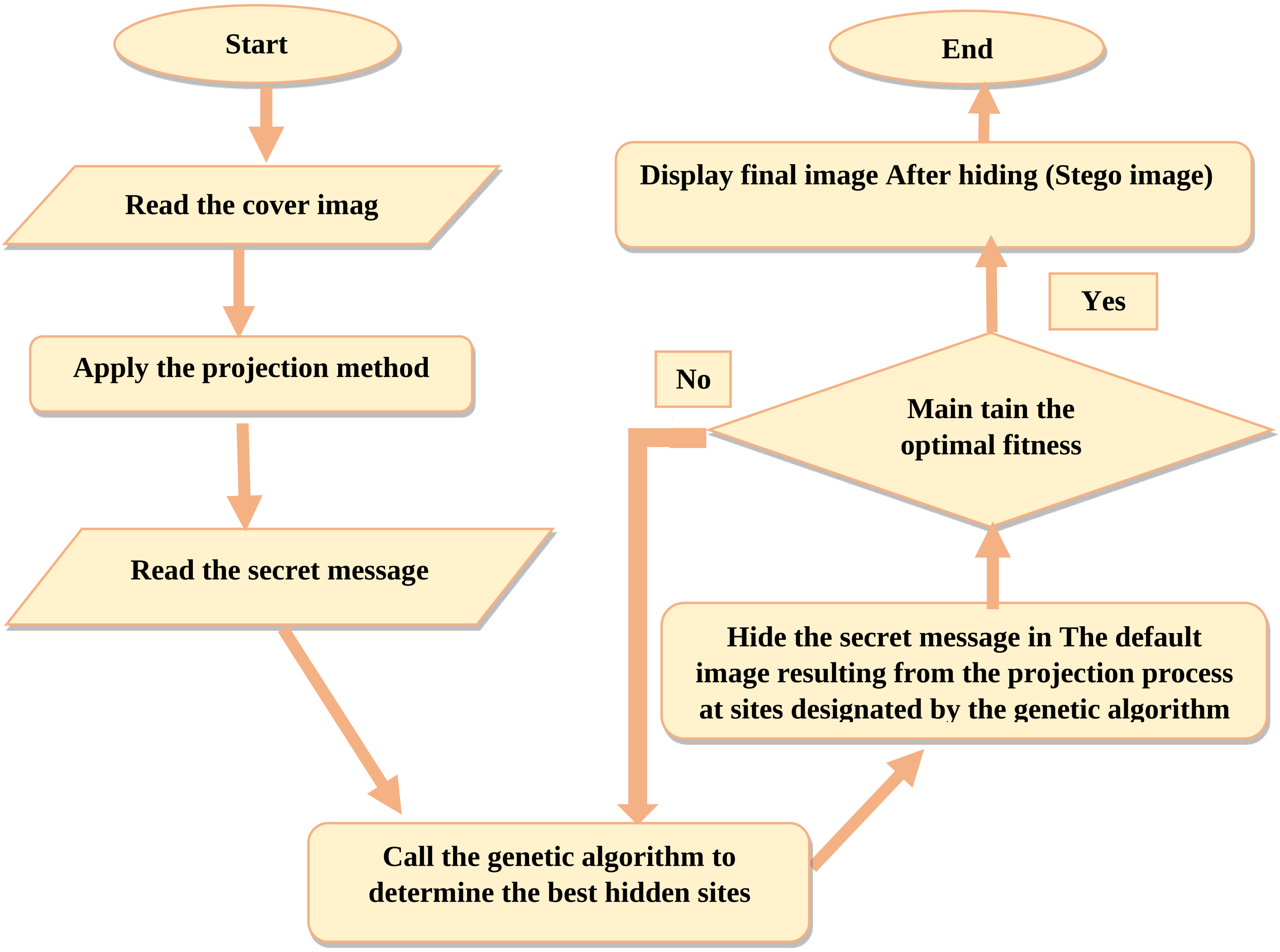

Start
Read the cover imag
Apply the projection method
Read the secret message
Call the genetic algorithm to determine the best hidden sites
Hide the secret message in The default image resulting from the projection process at sites designated by the genetic algorithm
Main tain the optimal fitness
No
Yes
Display final image After hiding (Stego image)
End

Figure2.9: Diagrame of Projection with GA Hiding.

5.5 Implementation of Extraction:

To achieve this, you must follow the steps below from start to finish:

1-Read the image after hiding(stego_image).

2-Extract the key for the projection process.

3-Extract the key to genetic algorithms.

4- Retrieve the confidential message from the default image resulting from the projection process at sites set by the genetic algorithm.

5- Display the confidential message(S).

5.6 Algorithm of Extraction:

The algorithm shows how the projection and GA method works in Extraction:

Algorithm: 'Image-Extraction_Projecion'

Input: Cover: Image(C), Secret: Image(S);

Matrix: Extract pixels from the stego image resulting from the projection process and convert to binary and position in a matrix(m).

Vector: Holds confidential information pixels.

```
Begin
Extract the key for the projection process.
Extract the key to genetic algorithms.
   For (k=0 Until Hight of M, Do)
        For (j=0 Until Width of M, Do)
               For (i=0 Until length of V, Do)
                        V [i] ← M[k][j];
         Fin pour ;
      Fin pour
Fin pour ;
End.
```

5.7 Graph of Projection with GA Function in Extraction:

This diagram explains how the projection with GA method follows in Extraction:

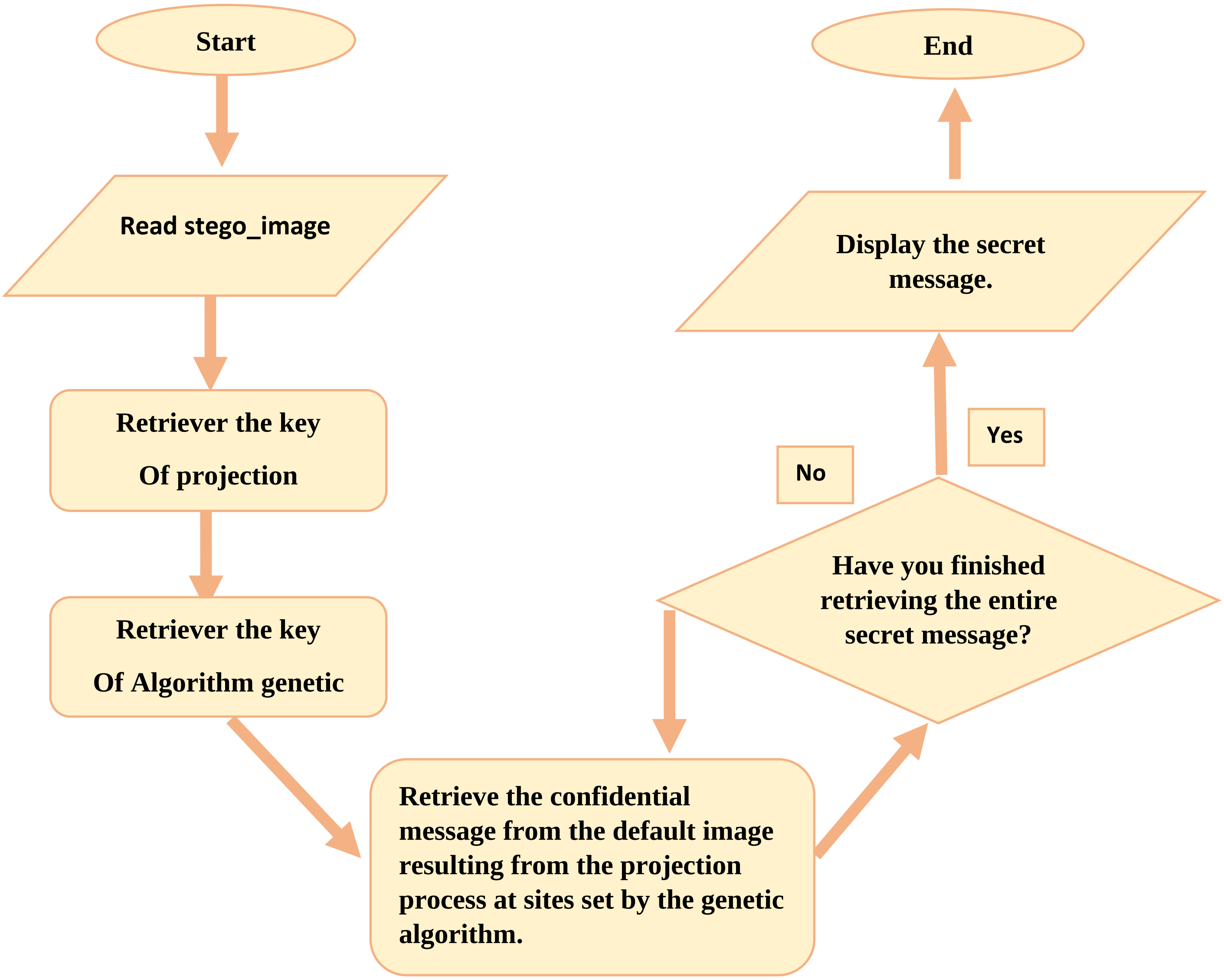

Start
Read stego_image
Retriever the key
Of projection
Retriever the key
Of Algorithm genetic
Retrieve the confidential message from the default image resulting from the projection process at sites set by the genetic algorithm.
Have you finished retrieving the entire secret message?
No
Yes
Display the secret message.
End

Figure2.10: Diagrame of Projection with GA Extraction.

6 Conclusion:

We concluded that the LSB method is easy to use by anyone who can retrieve Confidential information is easy.

The projection method is a sophisticated method that keeps the information confidential The analyst extracts it unless the hide key is obtained.

The projection method with the genetic algorithm is an ideal method because it maintains The secrecy of the information from its random work, which analysts can only extract If you know how to hide the genetic algorithm.

CHAPTER III: IMPLEMENTATION AND RESULTS

1 Introduction:

The statistical tests are intended to verify whether the properties of the inserted image have been modified. In general, the decision is made by comparing the quality standards of a particular image to the standards [34]. A watermark algorithm is said to be effective if the resulting metric measures are equal to the criteria [35]. In this chapter, we examined the results obtained in our work and compare them with methods LSB Based on many statistical measures. In addition, objective cognitive quality measures are used as an alternative method of determining the differences that result from a picture watermark [36].

2 Materials Used in Development:

The application is developed on a PC with the characteristics next:

- Operating System: Windows10 64-bit.
- Ram: 4 GB.
- Processor: Intel® Core ™ i5-7200U CPU @ 2.50GHz 2.71GHz

The chosen programming language is very efficient to develop the application. This is the java Specifically; the NetBeans IDE 8.2 RC environment was used.

3 Image Quality Metrics:

Statistical tests aim at investigating whether the characteristics of an input image have been modified or not. Generally, a decision is taken by comparing the quality metrics of a given image with norms [34].A watermarking algorithm is said to be efficient if the yielded metric measures are equal to the norms [35].In this work, we investigated the quality performance of the proposed watermarking method using numerous statistical metrics. Additionally, perceptual objective quality measures are exploited as an alternative approach to quantify the dissimilarities caused by image watermarking [36].The investigated metrics are as follow:

3.1 Mean Square Error (MSE):

MSE stands for the mean squared difference between the original image and the projected image. It is calculated by formula (1).

$$MSE = \left(\frac{1}{M\times N}\right)\sum_{i=1}^{M}\sum_{j=1}^{N}\left(a_{ij} - b_{ij}\right)^{2} \qquad (1)$$

-a_{ij} is the value of the pixel at the coordinates (i, j) in the original image.

- b_{ij} is the pixel's value at the same coordinates in the corresponding generated image [37].

3.2 Peak Square Noise Ratio (PSNR)

PSNR, given by Eq. 4, is a classical quality index defined as the ratio between the maximum possible pixel value (e.g., equals to 255 in case of RGB color images) and the mean square error [38].It is given by:

$$PNSR = \frac{10log_{10}255^2}{MSE} \tag{2}$$

- PSNR of RGB color images can be calculated by evaluating then summing up MSEs of all channels. That's to say, the peak value $\frac{255^2}{MSE}$ is replaced with $\frac{255^2 \times 3}{\sum_{i\in\{R,G,B\}} MSE_i}$.

-PSNR is more consistent with the presence of error compared to the SNR.

3.3 Average Difference (AD):

AD is the difference average between the original and the cover image. AD is given by the Eq. 3.[37]:

$$AD = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(a_{ij}-b_{ij})}{MN} \tag{3}$$

-a_{ij} is the value of the pixel at the coordinates (i, j) in the original image.

- b_{ij} is the pixel's value at the same coordinates in the corresponding generated image.

3.4 Maximum Difference (MD):

MD, given by Eq. 4, is the maximum difference among pixels of the original image and their corresponding ones in the cover image [37].

$$MD = MAX\left|a_{ij} - b_{ij}\right| \tag{4}$$

-a_{ij} is the value of the pixel at the coordinates (i, j) in the original image.

- b_{ij} is the pixel's value at the same coordinates in the corresponding generated image.

3.5 Peak Mean Square Error (PMSE):

It stands for the mean square error (MSE) based on the square of the maximum value among original image pixels. [37] by Eq. 4:

$$PMSE = \frac{1}{MN} \times \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(a_{ij}-b_{ij})^2}{[MAX(a_{ij})]^2} \tag{5}$$

-a_{ij} is the value of the pixel at the coordinates (i, j) in the original image.

- b_{ij} is the pixel's value at the same coordinates in the corresponding generated image.

3.6 Normalized Cross-Correlation (NK):

NK is one of the methods used for template matching. The process used for finding incidences of cover images and original images. by Eq. 6:

$$NK = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(a_{ij}\times b_{ij})}{\sum_{i=1}^{M}\sum_{j=1}^{N}(a_{ij}{}^{2})} \tag{6}$$

-a_{ij} is the value of the pixel at the coordinates (i, j) in the original image.

- b_{ij} is the pixel's value at the same coordinates in the corresponding generated image.[37]

3.7 Structural Content (SC):

SC can be used to determine the nearest of the original image from the cover image.

by Eq. 7:

$$SC = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(a_{ij}{}^{2})}{\sum_{i=1}^{M}\sum_{j=1}^{N}(b_{ij}{}^{2})} \tag{7}$$

-a_{ij} is the value of the pixel at the coordinates (i, j) in the original image.

- b_{ij} is the pixel's value at the same coordinates in the corresponding generated image.[37]

3.8 Laplacian Mean Square Error (LMSE):

LMSE quantifies the quality of image reconstruction. by Eq. 8:

$$LMSE = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\left(O(a_{ij})-O(b_{ij})\right)}{\sum_{i=1}^{M}\sum_{j=1}^{N}\left(O(a_{ij})\right)^{2}} \tag{8}$$

-a_{ij} is the value of the pixel at the coordinates (i, j) in the original image.

- b_{ij} is the pixel's value at the same coordinates in the corresponding generated image.

- where $O\left(a_{ij}\right) = a_{i+1j} + a_{i-1j} + a_{ij+1} + a_{ij-1} - 4a_{ij}$. [35]

3.9. Normalized Absolute Error (NAE):

NAE is a measure of how far is the stego image from the original cover image with the value of zero being the perfect fit. Big value of NAE indicates poor quality of the resulting image after embedding. by Eq. 9:

$$NAE = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}|a_{ij}-b_{ij}|}{\sum_{i=1}^{M}\sum_{j=1}^{N}|a_{ij}|} \tag{9}$$

-a_{ij} is the value of the pixel at the coordinates (i, j) in the original image.

- b_{ij} is the pixel's value at the same coordinates in the corresponding generated image.[37]

3.10 Robustness Test (BER):

Robustness test aims at examining the capability of a system to resist signal modifications in real-life applications. The detection rate is measured by Bit Error Rate (BER) using Eq. 10.

$$BER = \frac{Ext_{bit}}{T_{bit}} \tag{10}$$

-where, Ext_{bit} is the number of successfully extracted watermark bits, and T_{bit} is the total number of original watermark bits.[37]

4 The Forms Selected for Testing:

We have selected these images for our application tests

- Image (1): the secret data (66×66).
- Image (2): the secret data (200×200).
- Image (3): the cover image (200×200).
- Stego_image: Cover photo after hiding
- Image Extracted: Confidential information extracted from the cover after hiding

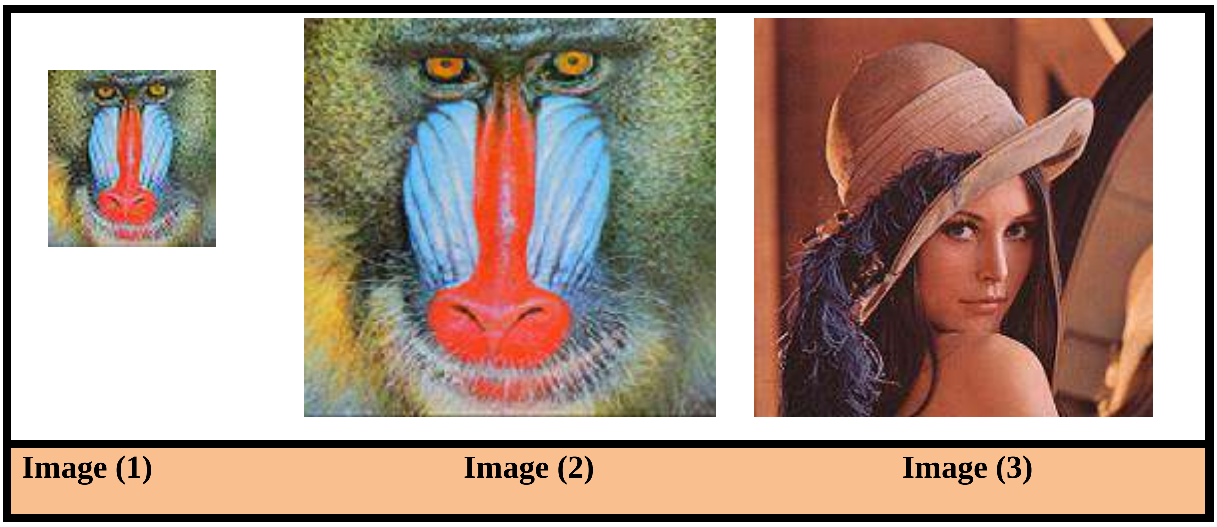


Figure 3.1: Model for Testing.

4.1 LSB Method Test:

We would like to use the LSB information in two different stages:

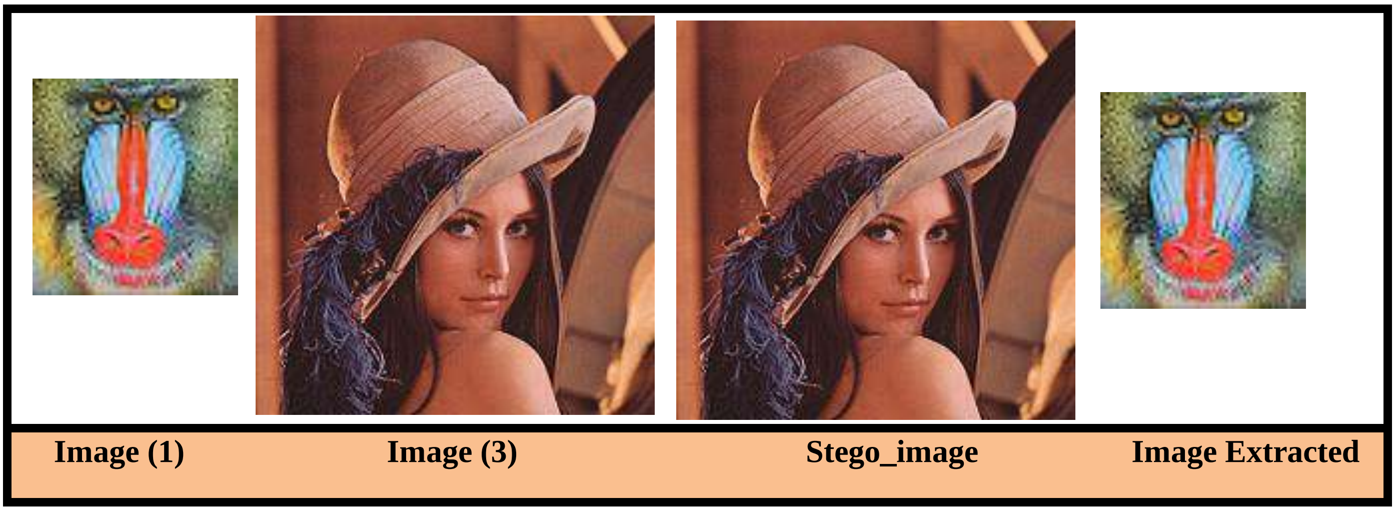


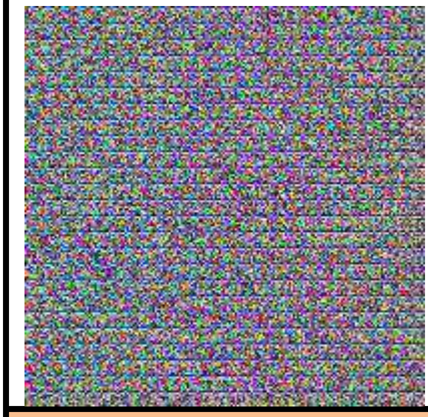
Figure 3.2: Hiding Confidential Information (1) in the Cover Image (3) using the LSB method

- Results:

Figure 2 shows that when the image (1) is small in size on image (3), there is no change in the stego image. The cover before and after the hiding is similar and, we have extracted the confidential information that matches the image (1) the opposite is true, which is what the following test shows:

Image (2)

Image (3)



Stego_image

Image Extracted

Figure 3.3: Hiding Confidential Information (2) in the Cover Image (3) using the LSB method

- Results:

Figure 3 shows that when images (2) and (3) are the same dimensions, there is a significant change in the image of Stego. Cover before and after hiding is deformed, and we have extracted the confidential information that same the image (1).

4.2 Projection Method Test:

In this stage we have added the following factors for the projection process:

- Angle = 0.26 radians and distance Ovo = 0.6 to Columns and lines.
- The dimensions of the virtual image resulting from the projection process in light of these data are equal (172×172).

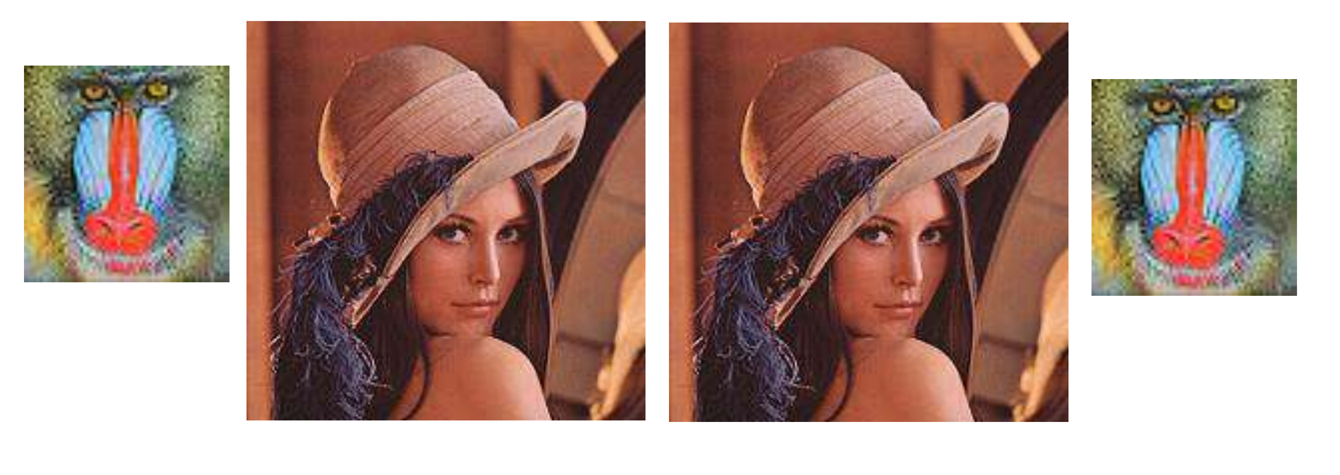


Image (1) Image (3) Stego_image Image Extracted

Figure 3.4: Hiding Confidential Information (1) in the Cover Image (3) using the Projection method

- Results:

Figure 4 shows that when the image (1) is small in size on image (3), there is no change in the stego image. The cover before and after the hiding is similar and, we have extracted the confidential information that the same of the image (1) the opposite is true, which is what the following test shows:

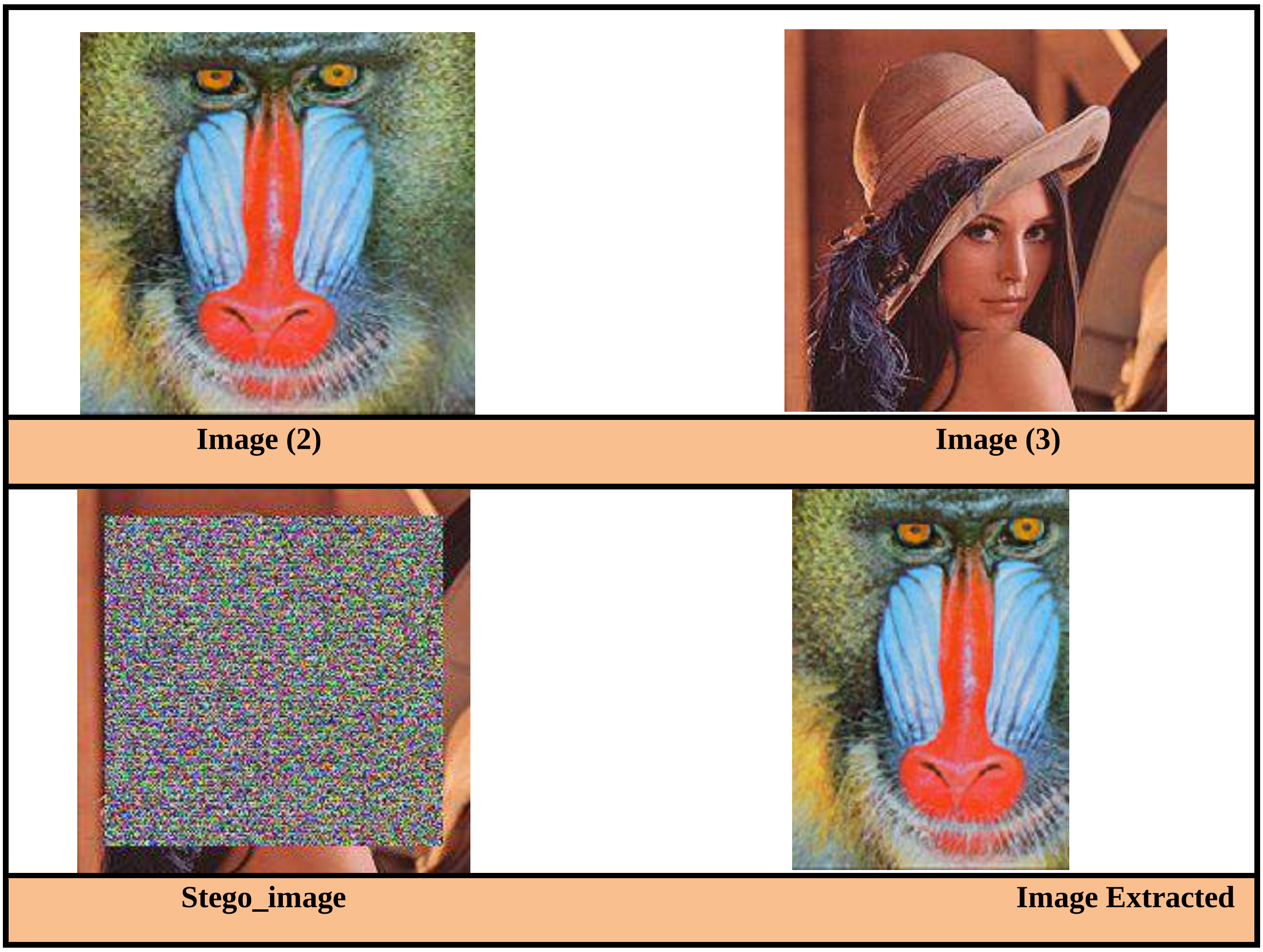


Image (2)

Image (3)

Stego_image

Image Extracted

Figure 3.5: Hiding Confidential Information (2) in the Cover Image (3) using the Projection method

- Results:

Figure 5 shows that when images (2) and (3) are the same dimensions, there is a significant change in the Stego image. The cover before and after hiding is deformed and we have extracted the confidential information, incomplete, not identical to the original.

4.2.1 Direct Image Projection Analysis:

you will make experimentation and a discuss of:

1 - The relationship between the size of the image and the corresponding max size of the watermark. The distance and angle have respectively been set to 2.0 and 30o for both vertical and horizontal projection.

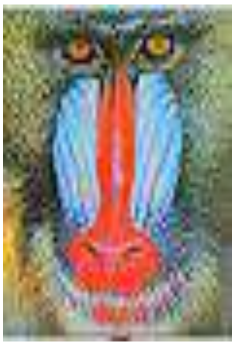
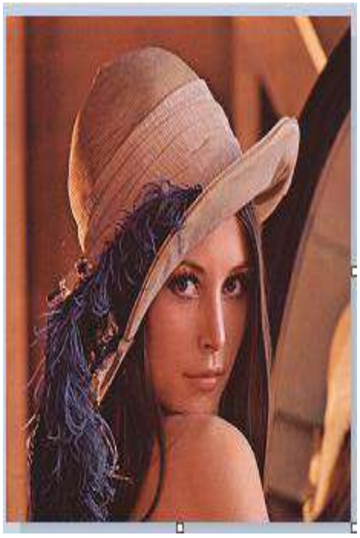
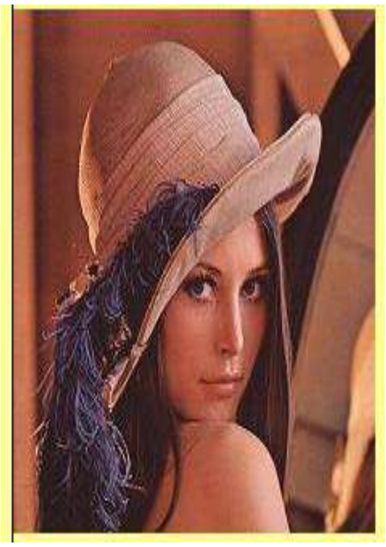
See table1:

Image size	64*64	128*128	256*256	512*512	1024*1024
max size of the watermark	60*60	122*122	248*248	496*496	994*994
percentage	87.89%	90.84%	93.85%	93.85%	94.22%

Table 3.1: The Relationship Between the Size of the Image and the Corresponding Max Size of the Watermark.

- We notice from Table 1 that when setting the rotation angle at 30 degrees and the distance at 2.0, the image size decreases compared to the original.
- As the image size increases, the maximum watermark size increases and the percentage increases, and there is a direct relationship between image size, maximum watermark size and percentage.
- The percentage refers to the pixels used when hiding confidential information in the cover photo is directly related to the cover photo.

2 - Some outcomes yielded by our proposed techniques in different scenarios where the images from left to right are the watermark, the original image, the generated cover image and the extracted watermark. (a) 66x66 pixels watermark and 225x225 pixels image(b) 200x200 pixels watermark and 225x22 pixels image (c) 220x220 pixels watermark and 225x225 pixels image.See table2:

Scenario **(A)**	 **A.1 watermark** **(66*66)**	 **A.2 original image** **(225*225)**	 **A. 3 generated cover image**

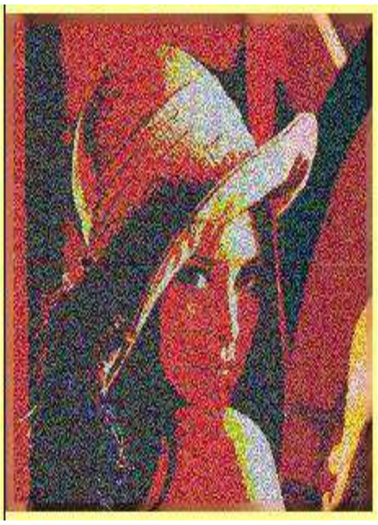
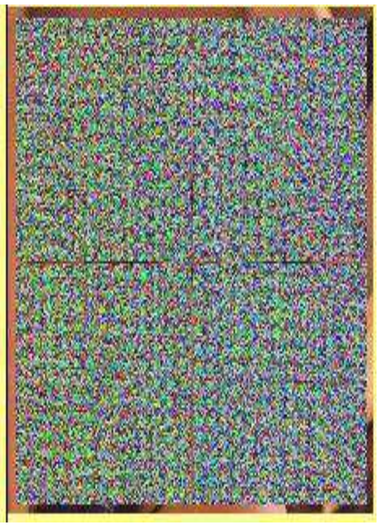
Scenario **(B)**	 **B.1 watermark** **(200*200)**	 **B.2 original image** **(225*225)**	 **B. 3 generated cover image**
Scenario **(C)**	 **C.1 watermark** **(220*220)**	 **C.2 original image** **(225*225)**	 **C. 3 generated cover image**

Table 3.2: Different Scenarios of the Images

- From Table 2, we note that the shape of the image after the hide process changes in 3 different scenarios.
- In the first scenario, when the size of the secret message was (66 * 66) and the cover image (225 * 225), we did not notice the change of the cover image after masking.

- In the second scenario, when the size of the secret message was (200 * 200) and the cover image (225 * 225), we recorded little change in the cover image after masking.
- In the third scenario, when the size of the secret message was (220 * 220) and the cover picture was (225 * 225), we recorded a change in the size of the cover image after concealing the confidential information to the extent that it completely changes its appearance.
- In the end, we conclude that as the size of the secret message increases to the size of the cover, the shape of the cover changes. Since we use the pixels in the cover image for masking, so the more pixels we use, the look of the cover changes.

3- The effect of choosing the rotation angle on the hiding available proportion. The image size and distance are 250x250 and 1.5 respectively in all cases.

See table3:

Angle(rad)	**0**	**0.26**	**0.52**	**0.78**	**1.04**	**1.30**	**1.57**
Available Proportion	0%	89.11%	86.11%	80.28%	50.69%	15.68%	0.001%

Table 3.3: The effect of choosing the rotation angle on the hiding.

- From Table 3, we note that as the rotation angle increases, the available hiding ratio decreases in image size by 250 * 250 and in constant distance 1.5.
- There is an inverse relationship between angle and percentage of disappearance, the larger the rotation angle, the less pixels used in the masking process.

4- The effect of choosing the distance on the hiding available proportion. The image size and angle are 250x250 and 0.5 respectively in all cases.

See table3:

Distance	**0.4**	**0.8**	**1.2**	**1.6**	**2.0**	**2.4**	**2.8**
Available Proportion	44.08%	67.89%	81.72%	89.11%	92.16%	92.16%	90.63%

Table 3.4: The effect of choosing the distance on the hiding.

- Note from the fourth table that the greater the distance, the greater the hiding percentage available in the image size of 250 * 250 and its angle of 0.5, meaning that there is a direct relationship between the distance and the percentage of disappearance.

4.3 the Use of Genetic Algorithms with Projection:

In this stage we have added the following factors for the projection process:

- Angle = 0.26 radians and distance Ovo = 0.6 to Columns and lines.
- The dimensions of the virtual image resulting from the projection process in light of these data are equal (172×172).

4.3.1 Problem Description:

The simplest way to hide confidential binary data on an image is to use the projection method that changes the positions of the pixels in the cover image by the inputs (angle and dimension) that Entered by the user and used the least important per pixel bit method to hide binary data in the image. This is not safe where an attacker can simply find input to the projection process (angle and dimension) to quickly retrieve hidden information. We have called genetic algorithms to maintain the integrity and confidentiality of information because it is difficult to extract information. Our proposed work consists mainly of a concealment algorithm based on projection and genetic algorithm.

Given:(Cover image), (secret data).

Find: Best locations to hide confidential information.

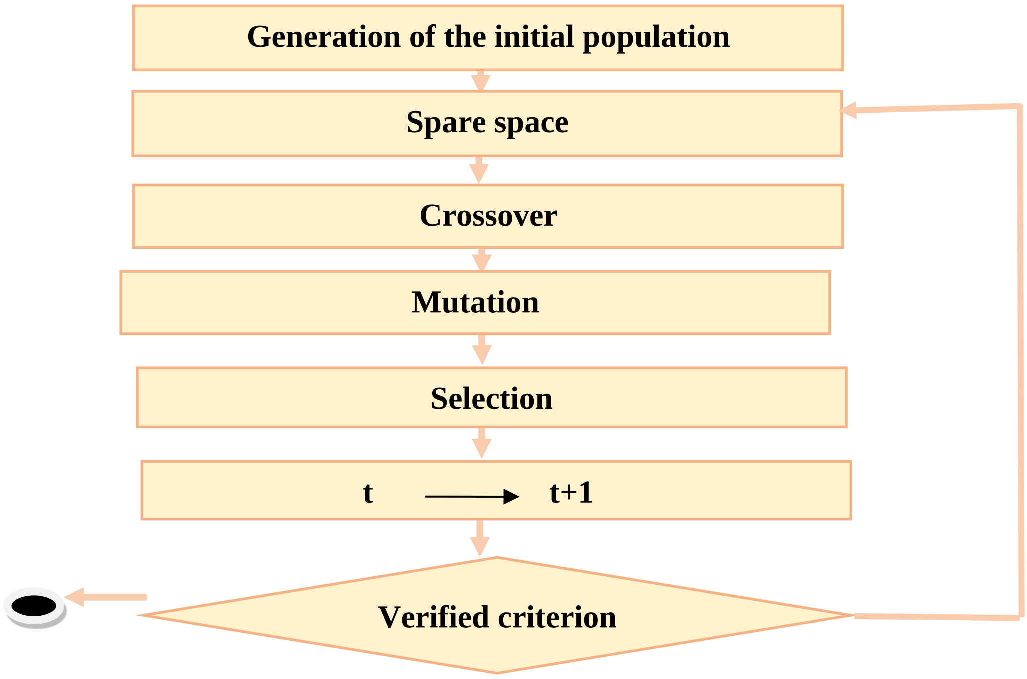


Figure 3.6: Classical Genetic Algorithm Structure.

1)Problem representation: Representation problem is search on best hiding locations ensuring confidentiality of information.

2) Population: We create a group of chromosomes that is the length of each chromosome (Number of pixels *24) one chromosome contains a number 1 and 0 group.

- The chromosome is created by dividing the cover image into groups of pixels.

Example chromosome 380:

```
chromosome [380]
01011101001011100010101101011011001011100010010101010110001011010010010001011110001011000
01011101001011000010100101011010001011000010100001011111001011010010100001011100001010011
01011001001001100010010101011110001011110010110001011101001100000010110001010110001010001
01011000001010110010010101010011001001110010010101010110001010000010100001011011001010111
01011000001001100010010101010001001001100010001101010011001010100010100001010010001010000
01010001001001100010011001001101001001010010010101001100001010100010011101001100001001010
01001111001001100010011001001100001001010010010101001010001000110010010001001110001001001
01001111001001000010011001001011001000110010001101001101001001100010010101001111001001010
01010001001001010010010001010000001010000010011101001111001001110010011001001111001001000
01001111001001010010010001001110001001110010011101001110001001000010001101010000001001001
```

Figure 3.7: example of create chromosome.

```
chromosome [1]
01011101001011100010101101011011001011100010010101010110001011010010010001011110000101000
01011101001011000010100101011010001011000010100001011111001011010010100001011100000100111
01011001001001100010010101011110001011110010110001011101001100000010110001010110000100101
01011000001010110010010101010011001001110010010101010110001010000010100001011011000101011
01011000001001100010010101010001001001100010001101010011001010100010100001010010000101000
01010001001001100010011001001101001001010010010101001100001010100010011101001100000100010
01001111001001100010011001001100001001010010010101001010001000110010010001001110000100101
01001111001001000010011001001011001000110010001101001101001001100010010101001111000100110
01010001001001010010010001010000001010000010011101001111001001110010011001001111000100100
01001111001001010010010001001110001001110010011101001110001001000010001101010000000100101
 chromosome [2]
01011101001011100010101101011011001011100010010101010110001011010010010001011110000101000
01011101001011000010100101011010001011000010100001011111001011010010100001011100000100111
01011001001001100010010101011110001011110010110001011101001100000010110001010110000100101
01011000001010110010010101010011001001110010010101010110001010000010100001011011000101011
01011000001001100010010101010001001001100010001101010011001010100010100001010010000101000
01010001001001100010011001001101001001010010010101001100001010100010011101001100000100010
01001111001001100010011001001100001001010010010101001010001000110010010001001110000100101
01001111001001000010011001001011001000110010001101001101001001100010010101001111000100110
01010001001001010010010001010000001010000010011101001111001001110010011001001111000100100
```

```
01001111001001010010010001001110001001110010011101001110001001000010001101010000000100101
 chromosome [3]
01011101001011100010101101011011001011100010010101010110001011010010010001011110000101000
0101110100101100001010010101101000101100001010000101111100101101001010000101110000000100111
01011001001001100010010101011110001011110010110001011101001100000010110001010110000100101
01011000001010110010010101010011001001110010010101010110001010000010100001011011000101011
01011000001001100010010101010001001001100010001101010011001010100010100001010010000101000
0101000100100110001001100100110100100101001001010100110000101010001001110100110000000100010
01001111001001100010011001001100001001010010010101001010001000110010010001001110000100101
01001111001001000010011001001011001000110010001101001101001001100010010101001111000100110
0101000100100101001001000101000000010100000010011101001111001001110010011001001111000100100
0100111100100101001001000100111000100111001001110100111000100100001000110101000000100101
 Chromosome [4]
01011101001011100010101101011011001011100010010101010110001011010010010001011110000101000
0101110100101100001010010101101000101100001010000101111100101101001010000101110000000100111
01011001001001100010010101011110001011110010110001011101001100000010110001010110000100101
01011000001010110010010101010011001001110010010101010110001010000010100001011011000101011
01011000001001100010010101010001001001100010001101010011001010100010100001010010000101000
0101000100100110001001100100110100100101001001010100110000101010001001110100110000000100010
01001111001001100010011001001100001001010010010101001010001000110010010001001110000100101
01001111001001000010011001001011001000110010001101001101001001100010010101001111000100110
0101000100100101001001000101000000010100000010011101001111001001110010011001001111000100100
01001111001001010010010001001110001001110010011101001110001001000010001101010000000100101
```

Figure 3.8: Example of Create Population

3) Spare space: We split the cover photo to create a chromosomes collection

4) Crossover: This process is done by random selection of chromosomes, and then we determine the center of each. Then we change the second half of the chromosome, so that the second half of the chromosome is placed with the first half of the chromosome and the second half of the chromosome. Place it with the first half of the second chromosome and produce 2 new chromosomes. Example in figure 3.11.

- Before Crossover :

Chromosome28 :

```
chromosome_pere1 before crossover= 28
11001100011101110101001011001010011101000100111011001010011100110101000011000110011010000
11000000011100100100101111001001011100110100111011001011011101100101001011001011011101001
11000111011100010100111111001100011101100101000111001010011101000100111111001000011001111
11000111011101000101000011001001011101000101001011001010011100110100111111000111011010011
11001000011101010101000111001011011100110100110111001011011101010101000011001011011010011
11001000011101110101001011001001011101100100111111010000011110010101100011001100011010100
11001001011110010100111111001010011110000101100011010010011110100101100011001101011010111
11001101011110010101011011001011011101110101001111001000011101000101000011001010011010011
11001011011101110101001011001010011101110101001011000100011101000101000111000100011010010
11000111011101110101010111000111011101010101000111000101011100100101000011000101011010001
```

Figure 3.9 : Chromosome 28 Before Crossover.

Chromosome53

```
Chromosome_pere2 before crosser = 53
10101100010100000011010110110011010101000011100010101100010100010011000110101010010110111
10101010010100100011011010101010010101010011010110101111010100110011010110101100010110111
10101010010101110011101010101110010101110011011110101101010101010011100010101001010111000
10101100010100110011100010110000010110000011010110101101010100110011100010101001010111100
10101100010110010011110110110000010111100011110101011100101101100111101101011110101111110
10101110010110100011110010110001010111110100010010110101010111000011101110110010010111000
10101001010110000011101010110110010111110101001010101101011101100111100110110011011001101
10100111010101010011110010101110010101010011100110110000010111010011100110110100010111100
10101101010110010011100110101111010110110011110101011010101110100111111101100100110000000
10110001010111000011110010110010010110110011111110101110010110100011110110101110010110111
```

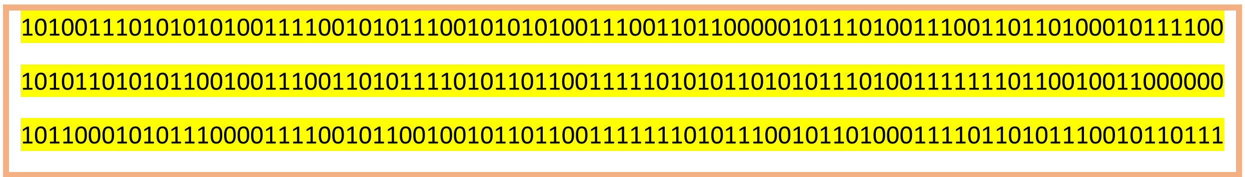
Figure 3.10 : Chromosome 53 Before Crossover.

Figure 3.5: chromosome 54 before crossover.

- ➢ After Crossover :

Chromosome28 :

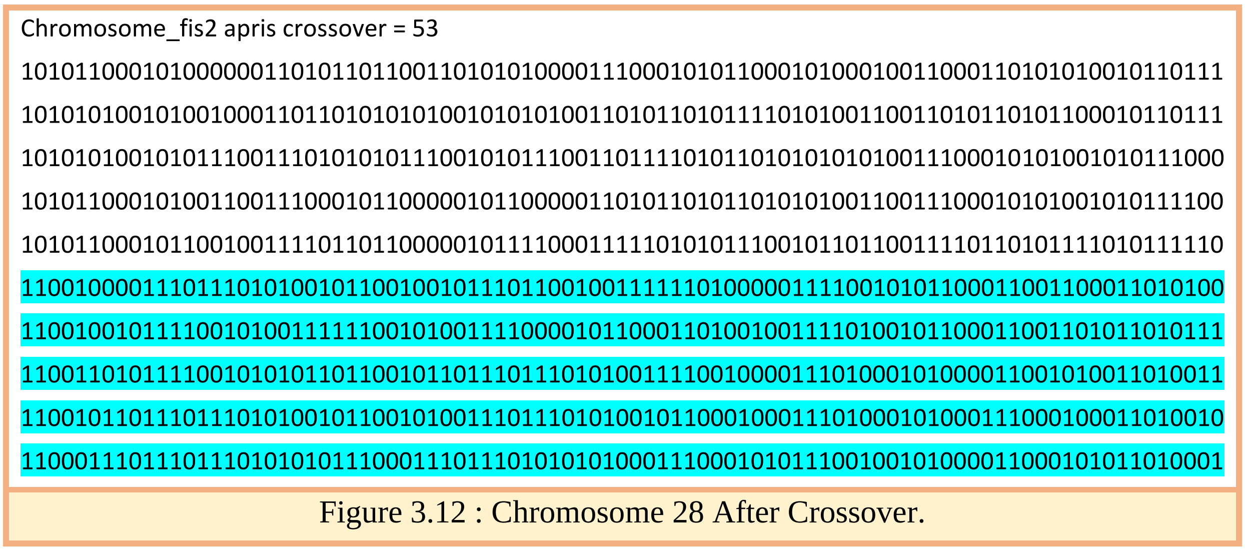
```
Chromosome_fis1 after crosser = 28
11001100011101110101001011001010011101000100111011001010011100110101000011000110011010000
11000000011100100100101111001001011100110100111011001011011101100101001011001011011010001
11000111011100010100111111001100011101100101000111001010011101000100111111001000011001111
11000111011101000101000011001001011101000101001011001010011100110100111111000111011010011
11001000011101010101000111001011011100110100110111001011011101010101000011001011011010011
10101110010110100011110010110001010111110100010010110101010111000011101110110010010111000
10101001010110000011101010110110010111110101001010101101011101100111100110110011011001101
```



```
1010011101010101010011110010101110010101010011100110110000010111010011100110110100010111100
1010110101011001001110011010111101011011001111010101101010111010011111110110010011000000
1011000101011100001111001011001001011011001111111010111001011010001111011010111001011011
```

Figure 3.11 : Chromosome 28 After Crossover.

Chromosome53 :



```
Chromosome_fis2 apris crossover = 53
101011000101000000110101101100110101010000111000101011000101000100110001101010100101101111
1010101001010010001101101010101001010101001101011010111101010011001101011010110001011011
1010101001010111001110101010111001010111001101111010110101010101001110001010100101011100 0
1010110001010011001110001011000001011000001101011010110101010011001110001010100101011110 0
1010110001011001001111011011000001011110001111101010111001011011001111011010111101011111 0
1100100001110111010100101100100101110110010011111101000001111001010110001100110001101010 0
1100100101111001010011111100101001111000010110001101001001111010010110001100110101101011 1
1100110101111001010101101100101101110111010100111100100001110100010100001100101001101001 1
1100101101110111010100101100101001110111010100101100010001110100010100011100010001101001 0
1100011101110111010101011100011101110101010100011100010101110010010100001100010101101000 1
```

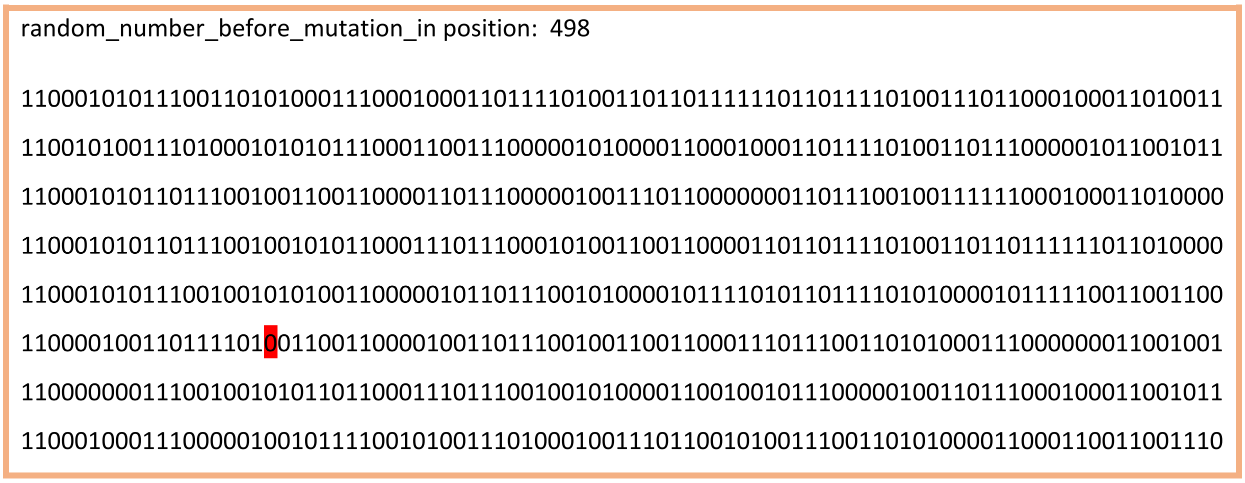
Figure 3.12 : Chromosome 28 After Crossover.

5) **Mutation:** We choose the same position randomly in all chromosomes then we invert it so if value is 0 it will be replaced by 1 and if it is 1 it will be replaced by 0**.**for example in Figure 3.14.

I took the example of the second chromosome from the group in both cases, Where we replace the value of 0 with 1:

➢ Before Mutation :

Chromosome2 :



```
random_number_before_mutation_in position:  498

11000101011100110101000111000100011011110100110110111111011011110100111011000100011010011
11001010011101000101010111000110011100000101000011000100011011110100110111000000101100101 1
11000101011011100100110011000011011100000100111011000000011011100100111111000100011010000
11000101011011100100101011000111011100010100110011000011011011110100110110111111011010000
11000101011100100101010011000001011011100101000010111101011011110101000010111110011001100
110000100110111110100110011000010011011100100110011000111011100110101000111000000011001001
1100000001110010010101101100011101110010010100001100100101110000010011011100010001100101 1
1100010001110000010010111100101001110100010011101100101001110011010100001100011001100111 0
```

```
110001110111000101001111110001010110111001001010110010010111001001010000110001000110011 10
110000010110110101001100110000110111000000100111010111111011011000100101111000101011001110
```

Figure 3.13: Chromosome 2 Before Mutation.

- After Mutation :

Chromosome2 :

```
random_number_after_mutation_in position:  498
11000101011100110101000111000100011011110100110110111111011011110100111011000100011010011
110010100111010001010101110001100111000001010000110001000110111101001101110000001011001011
110001010110111001001100110000110111000000100111011000000011011100100111111000100011010000
110001010110111001001010110001110111000101001100110000110110111101001101101111111011010000
110001010111001001010100110000010110111001010000101111010110111101010000101111110011001100
110000100110111110110110011000010011011100100110011000111011100110101000111000000001100100 1
110000000111001001010110110001110111001001010000110010010111000000100110111000100011001011
110001000111000000100101111001010011101000100111011001010011100110101000011000110011001110
110001110111000101001111110001010110111001001010110010010111001001010000110001000110011 10
110000010110110101001100110000110111000000100111010111111011011000100101111000101011001110
```

Figure 3.13: Chromosome 2 After Mutation.

6) **Selection:** in all the population we replace the chromosomes, with their sons in the intersections at each stage, after that we calculate the fitness value and finally we choose the best generation for the best fitness.

7) **Verified criterion:** A verified standard: Here we put the corresponding chromosomes in place for the best fitness to see what the chromosomes have improved or not.

4.3.2 Projection Method Test with GA:

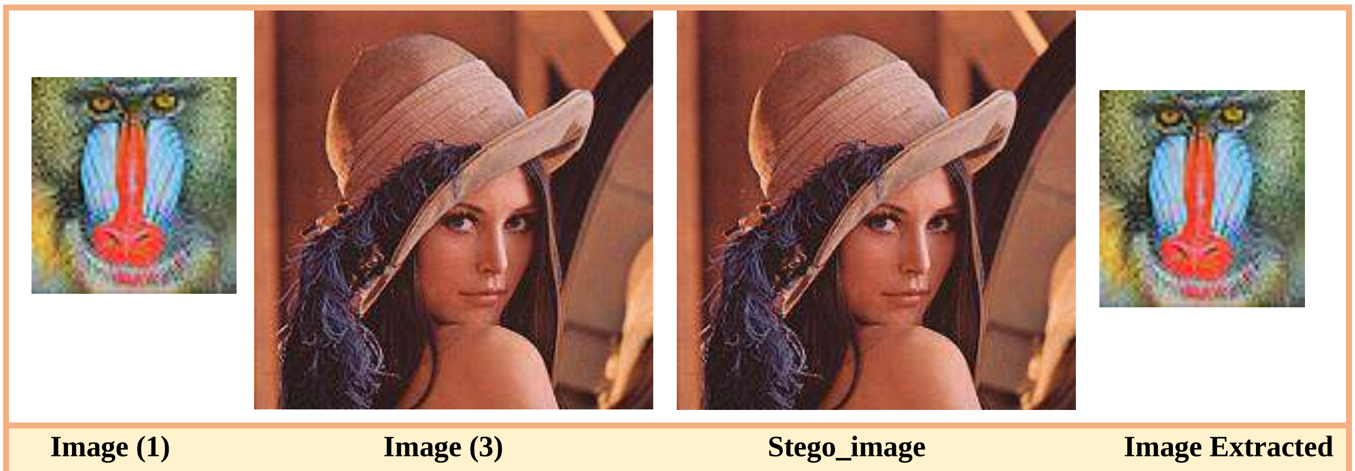


Figure 3.14: Hiding Confidential Information (1) in the Cover Image (3) using the Projection method with GA

- Results:

Figure 14 shows that when the image (1) is small in size on image (3), there is no change in the stego image. The cover before and after the hiding is similar and, we have extracted the confidential information that the same of the image (1) the opposite is true, which is what the following test shows:

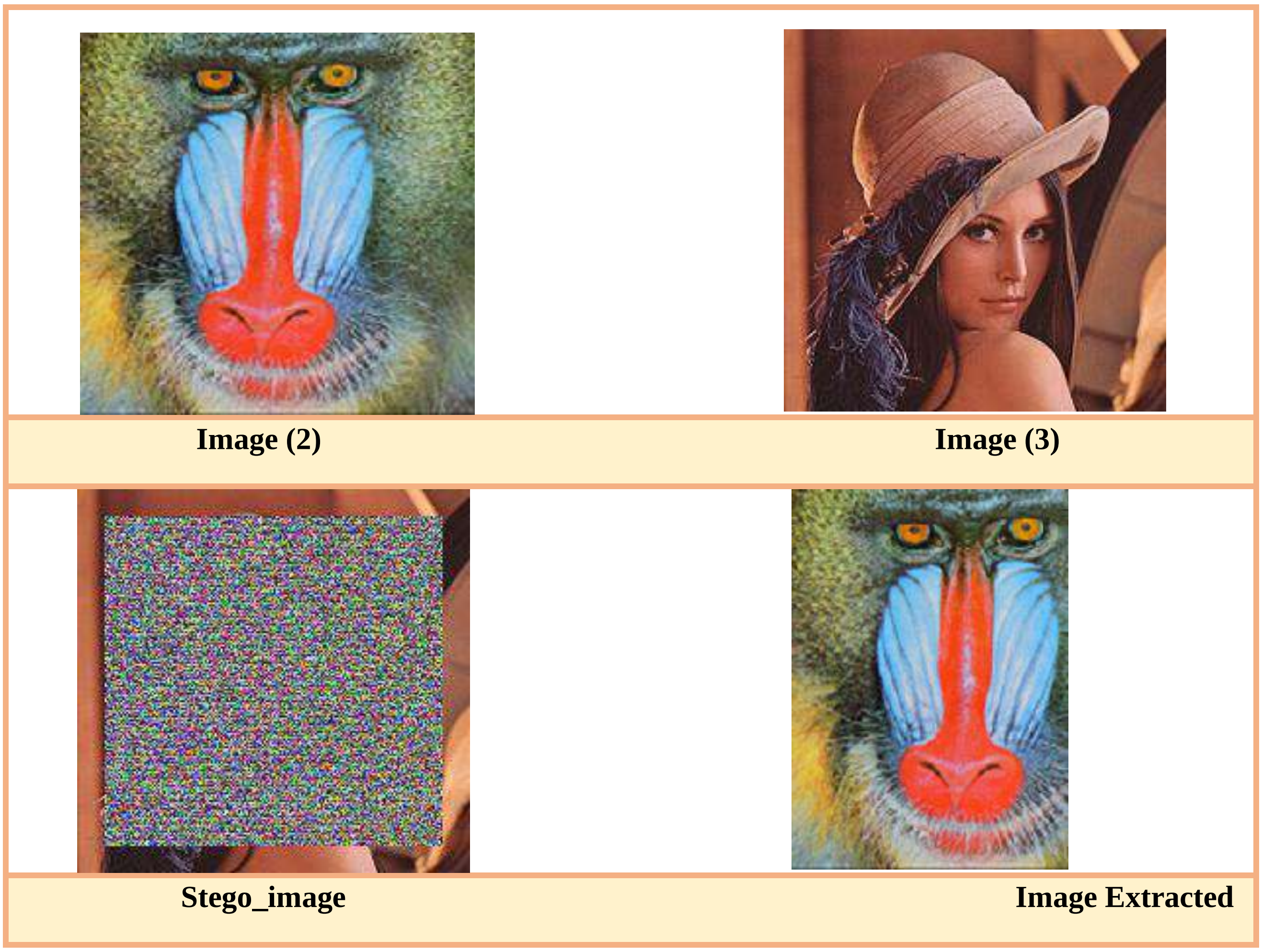


Image (2)

Image (3)

Stego_image

Image Extracted

Figure 3.15: Hiding Confidential Information (2) in the Cover Image (3) using the Projection method with GA

- Results:

Figure 15 shows that when images (2) and (3) are the same dimensions, there is a significant change in the Stego image. The cover before and after hiding is deformed and we have extracted the confidential information, incomplete, not identical to the original.

5 Quality Metrics Results:

Quality metrics results yielded by the proposed method compared to the conventional LSB substitution and Projection based LSB.

Method	Secret message (Kbytes)	MSE	PNSR	PMSE	AD	MD	SC	NK	LMSE	NAE	BER
Projection based GA	8	0.070	64.42	1.25	-0.068	1	0.999	1.0	3,01 E-22	0.007	100
	16	0.139	61.45	2.481	-0.112	1	0.998	1.0	-8.97 E-20	0.001	100
	32	0.270	58.57	4.81	-0.002	3	0.999	1.0	-1.01 E-20	0.002	100
	64	0.917	53.277	1.630	-0.009	5	0.999	1.0	-2.70 E-19	0.006	100
	128	2.47	48.96	4.401	-0.014	9	0.999	1.0	-4.80 E-20	0.09	100
	256	7.806	43.977	1.395	-0.001	17	0.999	1.0	-2.66 E-19	0.09	100
Projection based LSB	8	0.079	63.93	1.404	-0.0769	2	0.999	1.0	-6.91 E-20	0.007	100
	16	0.155	60.99	2.75	-0.121	2	0.998	1.0	-1.82 E-19	0.001	100
	32	0.343	57.542	6.106	-0.003	3	0.999	1.0	-2.09 E-19	0.003	100
	64	1.040	52.728	1.850	-0.012	5	0.999	1.0	-2.85 E-19	0.006	100
	128	3.470	47.498	6.169	-0.031.	13	0.998	1.0	-2.14 E-19	0.006	100
	256	10.785	42.573	1.927	-0.14	29	0.927	1.0	-1.38 E-19	0.08	100
Conventi_onal LSB	8	0.079	63.92	1.404	-0.0763	2	0.999	1.0	1.20 E-7	0.001	100
	16	0.164	60.745	2.920	-0.135	3	0.998	1.0	7.31 E-8	0.001	100
	32	0.344	57.53	6.116	-0.100	3	0.998	1.0	2.52 E-7	0.002	100
	64	1.128	52.375	2.006	-0.275	5	0.996	1.0	1.16 E-6	0.005	100
	128	4.498	46.371	7.997	-0.008	13	0.996	1.0	1.07 E-5	0.07	100

	256	10.903	42.526	1.948	--0.206	29	0.995	1.0	5.06 E-6	0.07	100

Table3. 5: Quality Metrics Results.

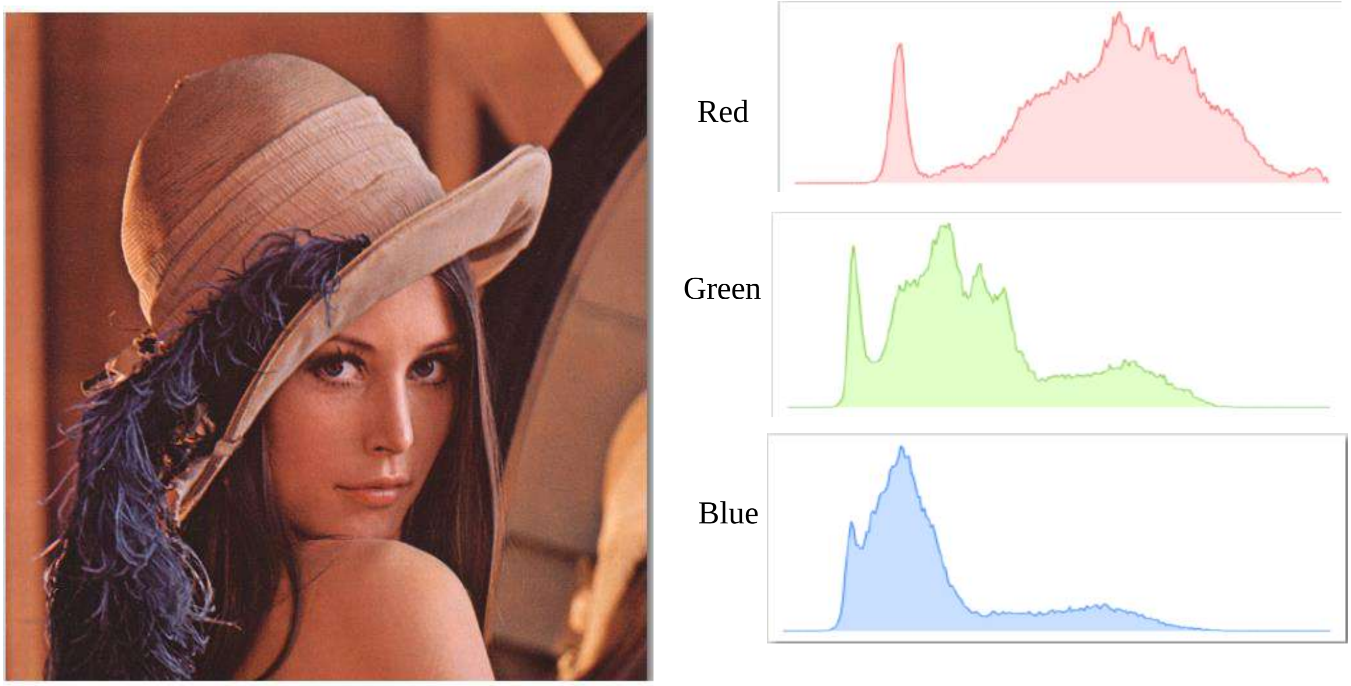
- We notice from the fifth table that the quality measures differ between conventional LSB method, the projection method, and the projection method based the genetic algorithm.
- Results obtained when using the projection method that based on the algorithm genetic shows the highest quality of results obtained at Use only the method of projection, and the results obtained when using a method Projection shows the best quality compared to the LSB method.
- We notice from the 5 table in the three methods that the value of MSE is decreasing, the value of PNSR increases, and AD gives negative values, MD is variable, PMSE is variable, BER is constant at 100,NK is constant at 1,SC approaches 1,LMSE is variable, and NAE approaches zero. All these results indicate the best masking

6 Security Analysis:

1- Gray-level histograms extracted from:

a) An original image :

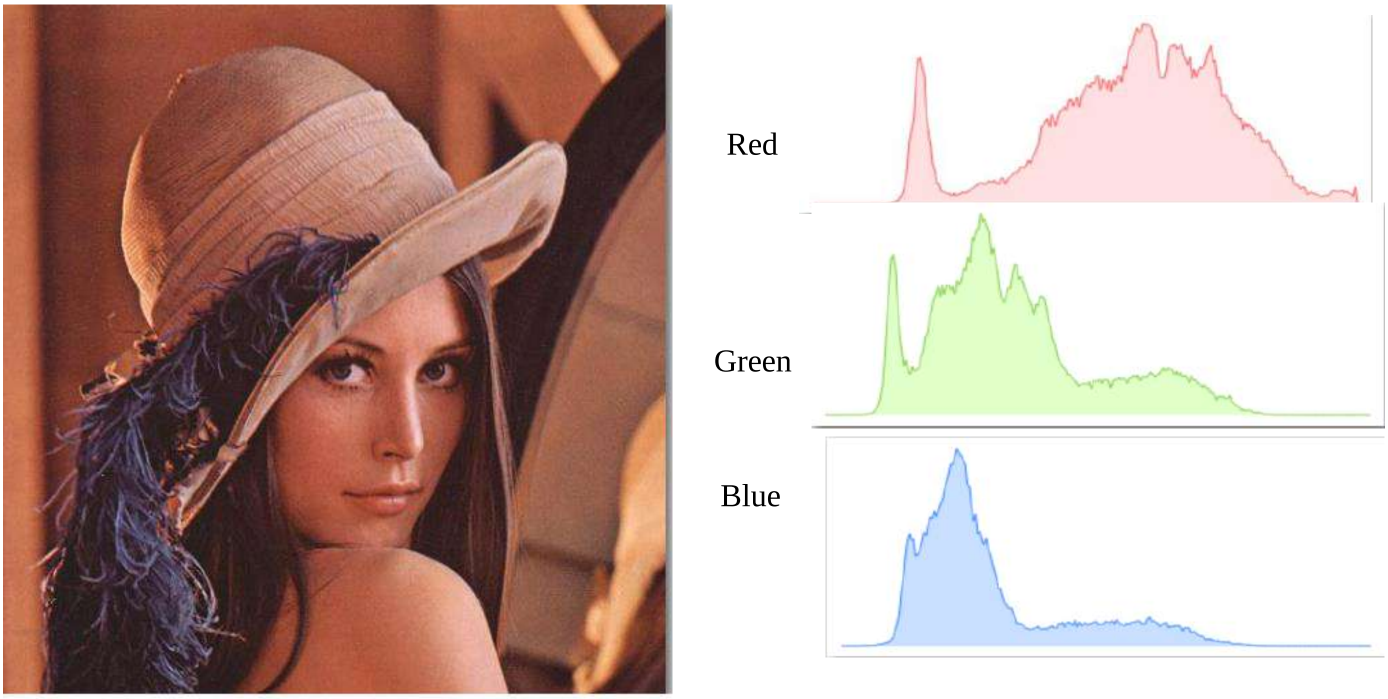
The gray level of the primary colors (red, green, blue) for an original image that was not subject to any Process. see figure (



Figur3.16: Histograms of RGB an Original Image

b) the cover image from Projection method with genetic algorithms:

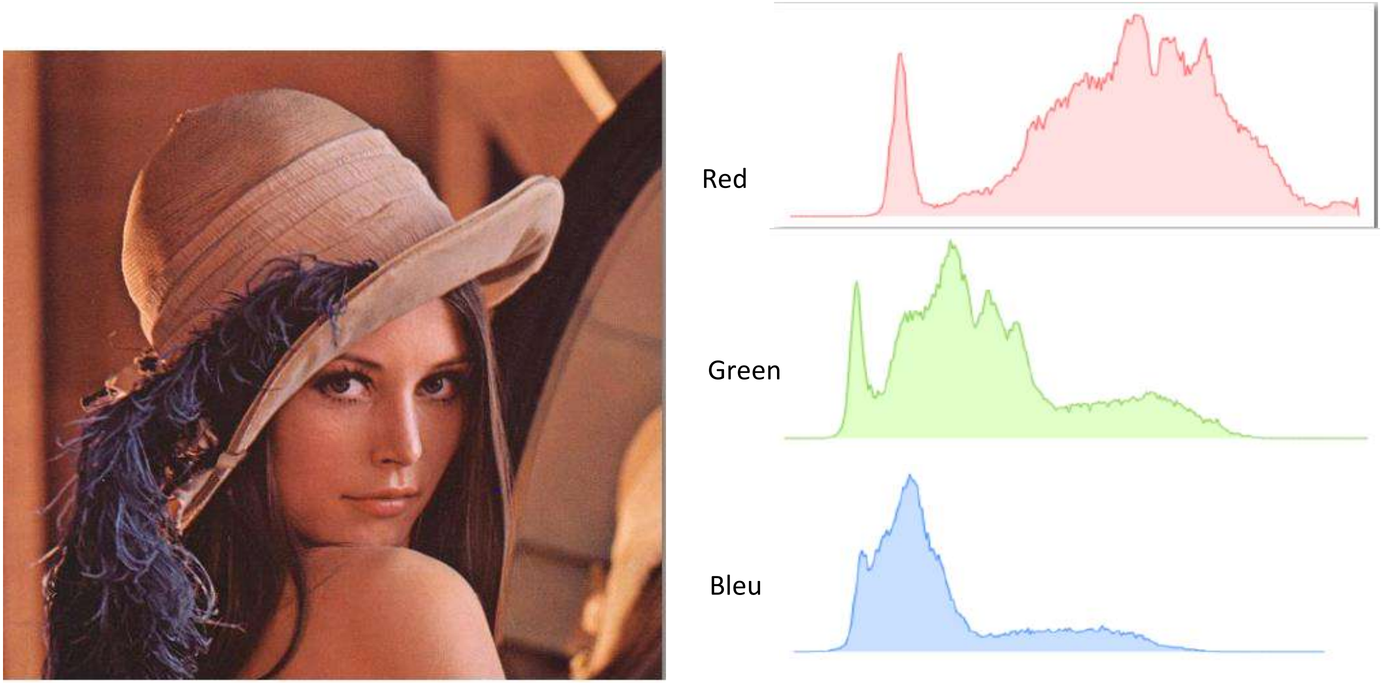
The gray level of the primary colors (red, green, blue) of an original image has undergone Projection method with genetic algorithms. see figure (17).



Figur3.17: Histograms of RGB on the Cover Using Projection Method with GA.

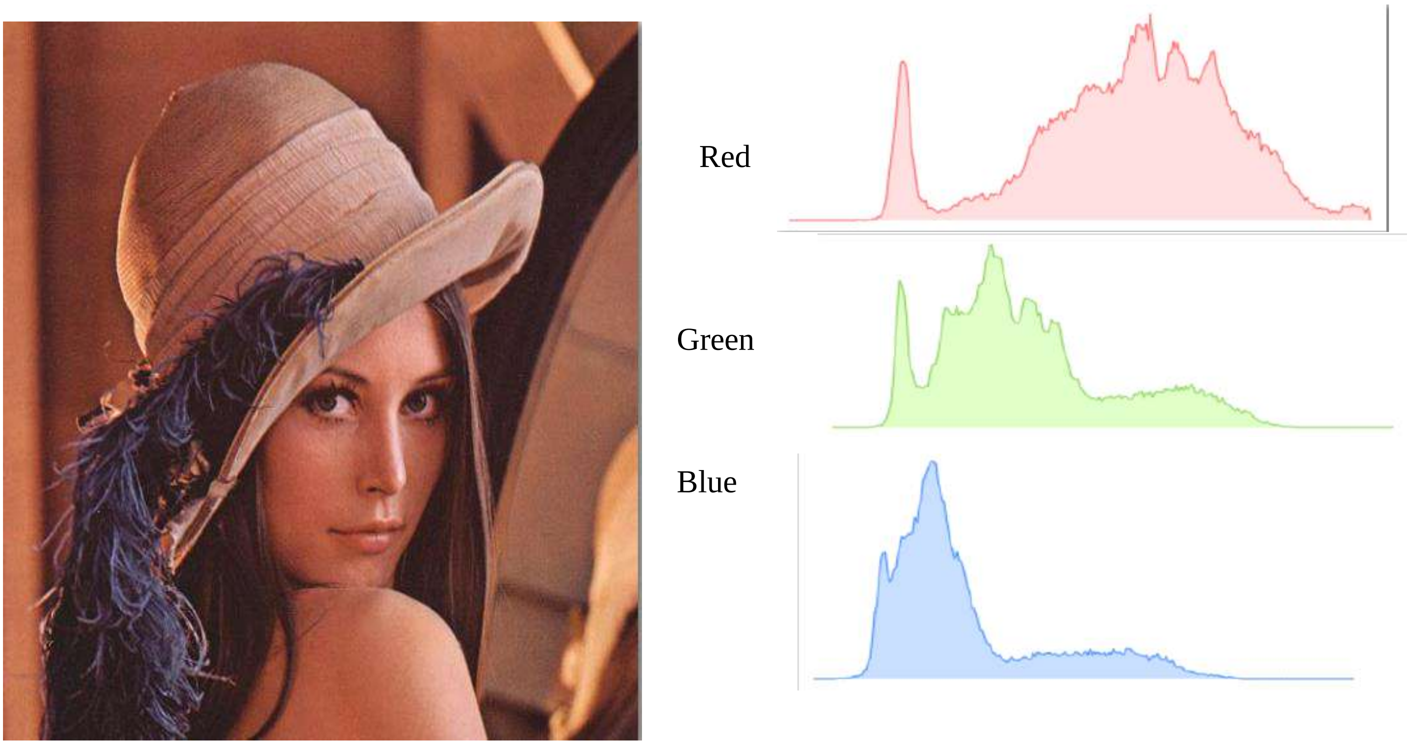
c) the cover image from Projection:

The gray level of the primary colors (red, green, blue) of an original image has undergone Projection method. see figure (18).



Figur3.18: Histograms of RGB the Cover Image from Projection.

d) the cover image from LSB method:
The gray level of the primary colors (red, green, blue) of an original image has undergone LSB method. see figure (19).



Figur3.19: Histograms of RGB the Cover Image from LSB.

- The proportions of the three colors changed in the curves compared to the proportions obtained in the original image.
- The three-color curves when using the projection method with the genetic algorithms are very close to the original image curves compared to the LSB method.

2- The outcomes of attacking cover images generated by the proposed method and Projection based LSB.

Steganalysis Method	Tool	Detection Outcome
Statistical Chi-Square Attack	StegDetect 0.4	False
Statistical RS Analysis	Virtual Steganography Laboratory	False
Linear Discriminant Analysis	StegDetect 0.4	False

Table3. 5: The Outcomes of Attacking.

- From Table 5 we conclude that the three methods (Statistical Chi-Square Attack, Statistical RS Analysis, Linear Discriminant Analysis) it not detected the confidential information in stego image So the methods (Projection, Projection with GA) we used them in our work are effective.

7 Conclusion:

In this chapter we get the following results:

- The results have proven the quality and efficiency of the developed method.
- Concealment of confidential information using the developed method of projection has been shown to increase the confidentiality of the information.
- The use of genetic algorithms has resulted in an increase in the confidentiality of information and has improved concealment.
- A following value was obtained for PNSR by using genetic algorithms.
- When using the genetic algorithm, we get the greatest similarity between cover image and image after image.

GENERAL CONCLUSION

Our thesis addressed the solution of hiding images problems by developing a near to optimal method that allows to achieve this goal, we have proposed a projection method that allows hiding of confidential information in an advanced way that enabled us to get good results, which means that no one can retrieve the confidential information unless he is able to obtain the key (angle, distance) Other than the well-known LSB method. However, the projection method has some defects when retrieving the information (that is, if the dimensions of the confidential information are less or equal to the dimensions of the default image resulting from the projection process, we can retrieve it completely. Retrieval).

The use of genetic algorithms increased the confidentiality of the information because it operates randomly, and it also improved concealment, which led to the best results for obtaining the optimal value of PNSR = 64.42. The fitness value is 33.25.

We also found the ratio of similarity of the cover before and after concealment, it is due to the size of the secret information. Whenever its size is small, the proportion of similarity is large, meaning that a few pixels have changed their values, while the larger the size of the secret information, the cover is deformed to change a large number of pixels.

ANNEX:

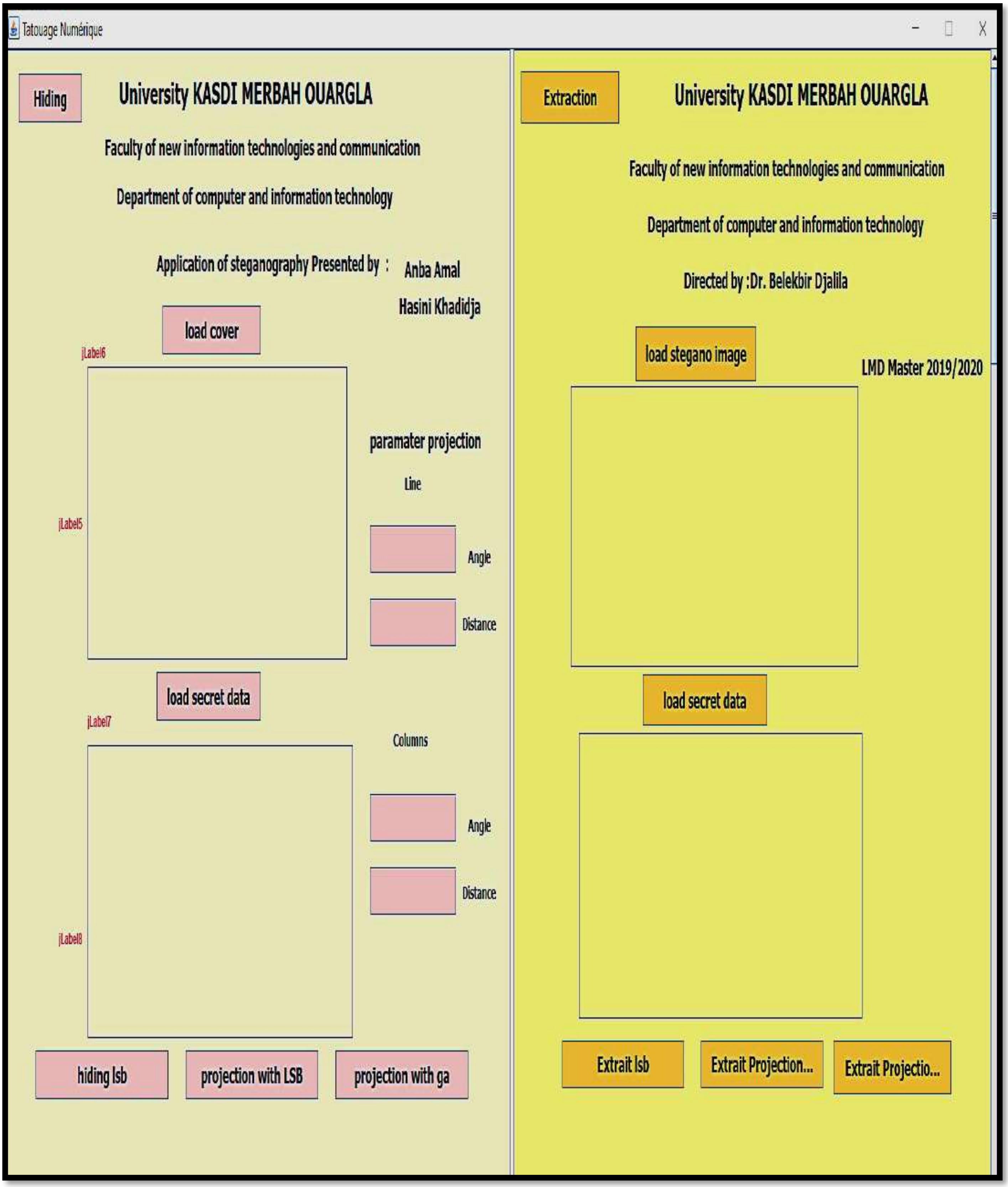

Tatouage Numérique
Hiding
University KASDI MERBAH OUARGLA
Faculty of new information technologies and communication
Department of computer and information technology
Application of steganography Presented by :
Anba Amal
Hasini Khadidja
load cover
jLabel6
jLabel5
paramater projection
Line
Angle
Distance
load secret data
jLabel7
Columns
Angle
Distance
jLabel8
hiding lsb
projection with LSB
projection with ga
Extraction
University KASDI MERBAH OUARGLA
Faculty of new information technologies and communication
Department of computer and information technology
Directed by :Dr. Belekbir Djalila
load stegano image
LMD Master 2019/2020
load secret data
Extrait lsb
Extrait Projection...
Extrait Projectio...

Figure A.1: Interface of application

- The general interface of the application is divided into two parts:
  - ➢ A special section for hiding.
  - ➢ A special section for extraction.

1- The first step is to upload a cover photo

2- The second step is to hold the confidential information

3- The third step we apply hiding.

- Hiding section, we have 3 methods:
  - ➢ Hiding with LSB.
  - ➢ Hiding with Projection.
  - ➢ Hiding with Projection and GA.

Whereas in hiding with the use of Projection method and Projection with GA method requires input (angle, distance).

4- Fourth step we apply the extraction.

1- Extraction section We have 3 extraction methods according to the hiding:

The extraction by LSB.

The extraction by Projection.

The extraction by Projection with GA.

- LSB

```
private void hidingLSBActionPerformed (java.awt. event.ActionEvent evt) {
     if(w*h<=la*lg){
        int p = 0,p1=0, RR,GG,BB;
        String vr1,vb1,vg1;
        steganoimage a=new steganoimage();
        int q=0;int c=0;
        //************* crée  matrice de RGB(mbgr) cover*************************//
        char[][] mbgr = new char [la*lg*3][8];
        int m=0,ps=0;
        String ss="";
         for(q=0; q<la;q++){
           for(c=0;c<lg;c++){
              p1 = imagc.getRGB(q,c);
              RR = (p1>>16) & 0xff;vr1=a.dectobin(RR);
              GG = (p1>>8) & 0xff;vg1=a.dectobin(GG);
              BB = p1 & 0xff;vb1=a.dectobin(BB);
              //************* charger les vecteure de rgb de cover dans les matrice*************//
              for(int k=0;k<8;k++){
                 mbgr[m][k]=vb1.charAt(k);
                 mbgr[m+la*lg][k]=vg1.charAt(k);
                 mbgr[m+la*lg*2][k]=vr1.charAt(k);
              }m++;} }
           //**************************creé vecteur de pixel  de secret data secret ***********//
           int x=7,t=0,o=0; char[] scr = new char [w*h*24];
           String sr,sg,sb;
           int rs,gs,bs;
         for(int i=0;i<h;i++)
         for(int j=0;j<w;j++){
            ps = images.getRGB(i,j);
```

```
int rs,gs,bs;
                for(int i=0;i<h;i++)
                for(int j=0;j<w;j++){
                    ps = images.getRGB(i,j);
                    rs = (ps>>16) & 0xff; sr=a.dectobin(rs);
                    gs = (ps>>8) & 0xff; sg=a.dectobin(gs);
                    bs = ps & 0xff;  sb=a.dectobin(bs);
                   for(int k=0;k<8;k++){
                        scr[o]=sr.charAt(k);
                        scr[o+(w*h*8)]=sg.charAt(k);
                        scr[o+(16*w*h)]=sb.charAt(k);
                          o++;}
                }
                //***********************hide rgb de secret dans cover*********************//
                o=0;
                for(x=7;o<h*w*24&& x>=0;x--)
                for(t=0;o<h*w*24&& t<la*lg*3;t++){
                    mbgr[t][x]=scr[o];
                    px=x;pt=t;
                    o++;}
                System.out.println("x=  "+x+"  t=  "+t);
               //****************ارجاع  les pixel aux cover ********
                int po=0,rr=0,rg=0,rb=0;
                for( int i=0;i<la;i++){
                for(int j=0;j<lg;j++){
                    rb =a. ts(po,mbgr);
                    rg = a.ts(po+(la*lg),mbgr);
                    rr = a.ts(po+(2*la*lg),mbgr);
                    p = (rr<<16) | (rg<<8) | rb;
                    imagc.setRGB(i,j , p);
                    po++;}
p = (rr<<16) | (rg<<8) | rb;
                imagc.setRGB(i,j , p);
                po++;}
                  }try
             { File ff = new File("C:\\Users\\lenevo\\Documents\\stegano_imagLSB.jpg");
                ImageIO.write(imagc, "jpg", ff);
               // imaged=ImageIO.read(ff);
            }
             catch(IOException e)
             {
                System.out.println(e);
             }
             File file = jFileChooser1.getSelectedFile();
             BufferedImage image = null;
             try {
                image = ImageIO.read(file);
             } catch (IOException ex) {
                Logger.getLogger(steganoimage.class.getName()).log(Level.SEVERE, null, ex);
             }
          }else
          System.out.print("erure : changer un image d'un taille petit au egale la taille de cover ");


   }
```

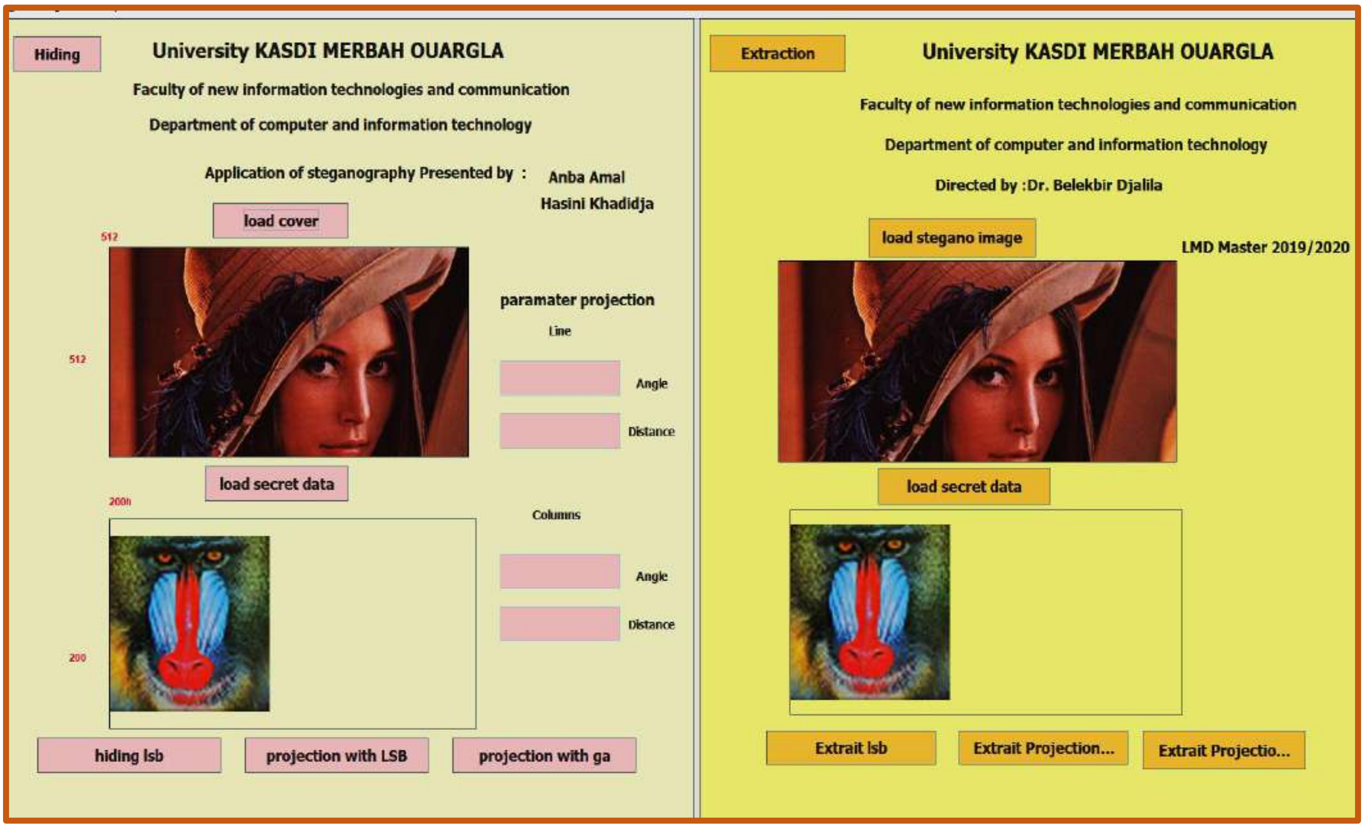
Figure A.2: code source of hiding_LSB.

```
private void Extrait_lsbActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
         //*******************************crée matrice mbgr1 *************************//
        char[][] mbgr1 = new char [la*lg*3][8];
        int p2=0;
        int RR,GG,BB; String vr1,vb1,vg1;
        steganoimage a=new steganoimage();
        int q=0;int c=0;int m=0;
          for(q=0; q<la;q++){
            for(c=0;c<lg ;c++){
                p2 = imagc.getRGB(q,c);
                RR = (p2>>16) & 0xff; vr1=a.dectobin(RR);
                GG = (p2>>8) & 0xff;vg1=a.dectobin(GG);
                BB = p2 & 0xff; vb1=a.dectobin(BB);
 //************* charger les vecteure de rgb de image stegano dans les matrice**************//
                for(int k=0;k<8;k++){
                    mbgr1[m+la*lg*2][k]=vr1.charAt(k);
                    mbgr1[m+la*lg][k]=vg1.charAt(k);
                 mbgr1[m][k]=vb1.charAt(k);
              }m++;}}
//*************************crée vecteur que prené les rgb desecret data**************
       char[] scr1 = new char [w*h*24];
     int t=pt,pr=(w*h*24)-1;
       for(int x=px;x<8;x++){
          for(;t>=0; t--){
             scr1[pr]=mbgr1[t][x];
             pr--; ;}
          if (t==-1)
          t=la*lg*3-1;;
       }
```

```
            if (t==-1)
          t=la*lg*3-1;
 }
        //**********************ارجاع les rgb de secret data aux foto**************
        int u=0,i=0,j=0;  String sr="",sg="",sb="";   int rs,gs,bs;
        for(int z=0;z<(w*h*8);z++){
            if(u<8){
                sr=sr+scr1[z];     sg=sg+scr1[z+(8*w*h)];     sb=sb+scr1[z+(16*w*h)];
                u++;
               } else{
                rs=bintod(sr);    gs=bintod(sg);    bs=bintod(sb);
               p = (rs<<16) | (gs<<8) | bs;
                if(i<h)
                if(j<w){
        images.setRGB(i,j,p);
                    j++;
         }else {
                 j=0;i++; images.setRGB(i,j,p);
                 j++;}
              sr="";sg="";sb="";u=0;
              sr=sr+scr1[z];    sg=sg+scr1[z+8*w*h];  sb=sb+scr1[z+16*w*h];
              u++;
      }
        }
              try{
          File f = new File("C:\\Users\\lenevo\\Documents\\secretLSB.jpg");
          ImageIO.write(images, "jpg", f);
       } catch(IOException e)
       {  System.out.println(e);
```

Figure A.3 : code source of Extrait_LSB

- EX: Implementation Hide and extracted with LSB:



University KASDI MERBAH OUARGLA

Faculty of new information technologies and communication

Department of computer and information technology

Application of steganography Presented by : Anba Amal
Hasini Khadidja

University KASDI MERBAH OUARGLA

Faculty of new information technologies and communication

Department of computer and information technology

Directed by :Dr. Belekbir Djalila

LMD Master 2019/2020

Figure A.4: Implementation with LSB

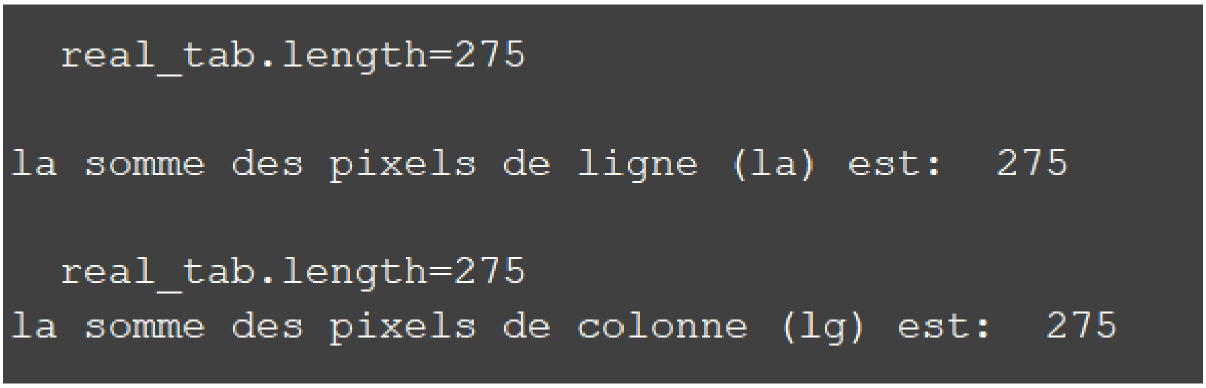
- Projection:

```
public static int [] calcul_opt (double OVo,double  theta,int collin){
    int [] lin2 = new int[collin];
int pos; int bl = 0; int taille_tab=0; double opt,OPt; double length; int r =collin/2; int count=0;
OVo =  OVo * collin;
double rem;
for (int indic = 0;indic<collin; indic++)
{     lin2[indic] = -1;  }
if (theta > 0)    // the iewer is at the right side of the original view
     {   rem   = (collin%2);
          if ( rem == 0)     // number of columns even
            { r --;
             for (int k = 0; k < collin ; k++)
             { if (k <= r )// points on the left of the original center
                   {   length = r-k+1;}
```

```
                else  {length = k-r;}
                  opt = (OVo*length*Math.cos(theta)/(OVo-(length*Math.sin(theta)))) ;
                    OPt = Math.round(opt);
                       if (k <= r)
                    {   pos = r- (int) OPt;
                         if(OPt == 0)
                            { pos=r; }}
             Else {       pos =  r +  (int) OPt;      }
            if ( OPt <= r) { if  (lin2[pos] == -1){
                          lin2[pos] = k;
                         if (k >-1)taille_tab++;
                       count++;} else{
                                }
}else{     }
            }}else    // Number of columns odd
{   for (int k =1 ; k<collin; k++)
             {    if   (k == r+0.5 )
                  { // do nothing
                  }else{ if ( k < r )
                      {     length = r-k+0.5;} else{     length = k-r-0.5; }
                  opt = (OVo*length*Math. cos(theta)/(OVo-length*Math.sin(theta))) ;
                 OPt = Math.round(opt);
                 if (k < r) {
                        pos = (int) OPt;           //%r+0.5-OPt;
                        if (OPt==0 ) {      pos=0;   }
                      }
                   else
                      {
                       pos = collin-  (int) OPt-1;//r+0.5+OPt;
                      }
}if ((OPt < r )&& (lin2[pos] == -1))
                                    {lin2[pos] = k;
                                    if (k >-1)taille_tab++;
                                    count++;}
}}}}
                int [] real_tab = new int[taille_tab];taille_tab=0;
for ( int g = 0; g< collin ; g++)
  { if (lin2[g] >-1){
       real_tab[taille_tab]=lin2[g];
       taille_tab++;}
}    Arrays.sort(real_tab);
     System.out.println();     System.out.println("  real_tab.length="+real_tab.length);
 // Arrays.sort(real_tab);      real_tab;}
```

Figure A.5: code source of Projection.

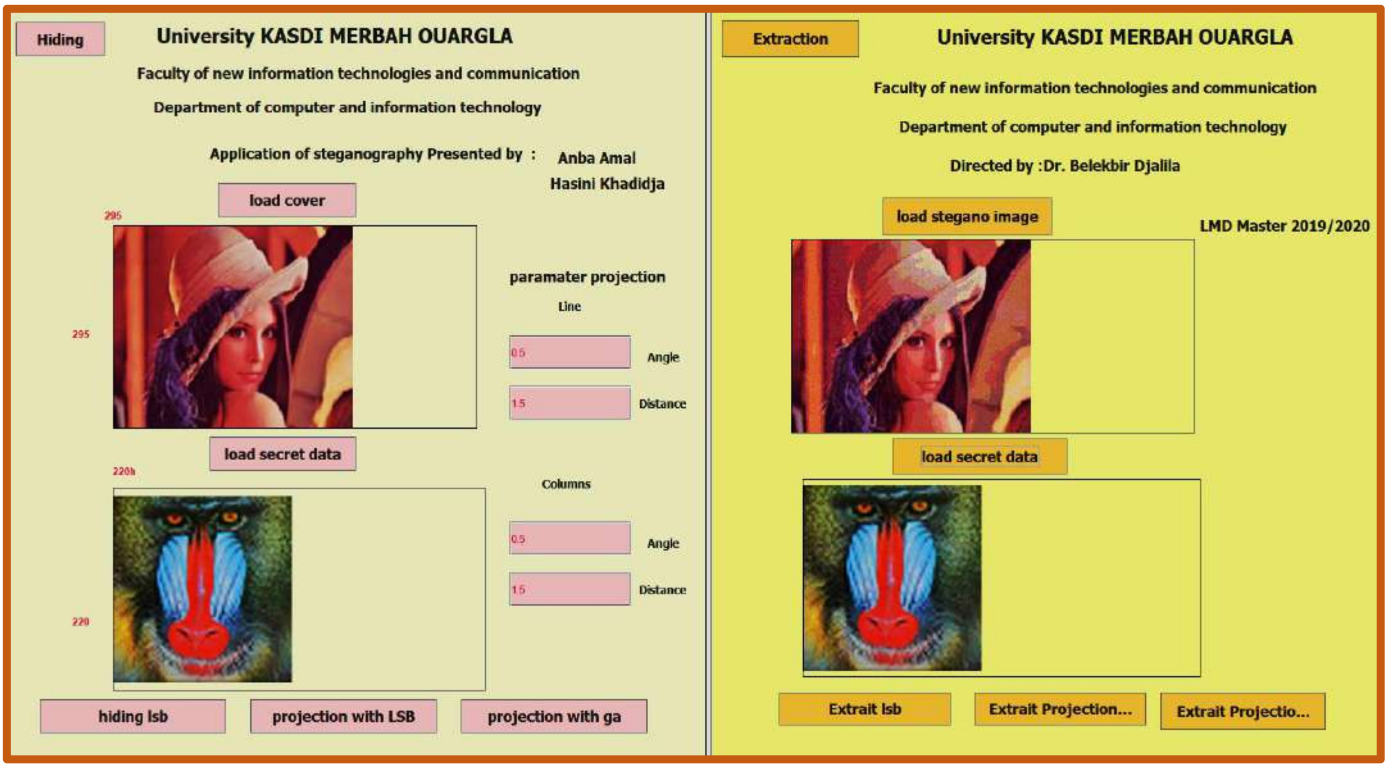
- ➢ EX: Implementation Hide and extracted with Projection based LSB:
- ➢ Size of image with Projection =275*275



```
  real_tab.length=275

la somme des pixels de ligne (la) est:  275

  real_tab.length=275
la somme des pixels de colonne (lg) est:  275
```



University KASDI MERBAH OUARGLA

Faculty of new information technologies and communication

Department of computer and information technology

Application of steganography Presented by : Anba Amal
Hasini Khadidja

paramater projection

Line

Angle: 0.5

Distance: 1.5

Columns

Angle: 0.5

Distance: 1.5

University KASDI MERBAH OUARGLA

Faculty of new information technologies and communication

Department of computer and information technology

Directed by :Dr. Belekbir Djalila

LMD Master 2019/2020

Figure A.6: Implementation with Projection

GA :To create the GA we applyed 4 basic proceses(crossover, mutation, selection, fitness)

```
public static void crosver(Population population) {
   for (int pop = 0; pop < (population.chromosome.length /40); pop++) {
            v.add(pop); }
//******************************************************
        //for (int i = 0; i < (population.chromosome.length / 40); i++) {
            int[] tab = new int[2];
            tab = nbr_random_non_répétif();
            int nbr_random = tab[0];
           int nbr_random2 = nbr_random;
            int nbr_random1 = tab[1];
            final int nbr_random11 = nbr_random1;
//***************affich cromosome peres*********
         Chromosome  va=population.get_chromosome(nbr_random);
         System. out.println ("chromosole_pere1 avant crosver= "+nbr_random);
         for(int i=0;i<40;i++){
             if(i%4==0)
             System. out.println (" ");System. out.print (va.get_genes(i)); }
       System. out.println (" ");
       Chromosome  va1=population.get_chromosome(nbr_random1);
       System. out.println ("chromosole_pere2 avant crosver = "+nbr_random1);
       for(int i=0;i<40;i++){
          if(i%4==0)
       System. out.println (" ");  System. out.print (va1.get_genes(i));} System. out.println (" ");
//   //******************************************************
        Chromosome fise1 = new Chromosome(40);Chromosome fise2 = new Chromosome(40);
        int k;
        for (k = 0; k < 20; k++) {
           fise1.add_genes(population.chromosome[nbr_random].get_genes(k), k);
           fise2.add_genes(population.chromosome[nbr_random11].get_genes(k), k); }
        for (k = 20; k < 40; k++) {
           fise2.add_genes(population.chromosome[nbr_random].get_genes(k), k);
           fise1.add_genes(population.chromosome[nbr_random11].get_genes(k), k);}
//*************************affich fise*********
       System. out.println ("chromosole_fis1 apris crosver = "+nbr_random);
      for(int i=0;i<40;i++){
         if(i%4==0)
            System. out.println (" "); System. out.print (fise1.get_genes(i));
      }      System. out.println (" ");
      System. out.println ("chromosole_fis2 apris crossver = "+nbr_random1);
      for(int i=0;i<40;i++){
         if(i%4==0)
            System. out.println (" ");System. out.print (fise2.get_genes(i));
      } System. out.println (" ");}
```

Figure A.7:code source of Crossover.

```
public static double fitness(double mse, double pnsr, double nf) {
        double f = nf;
      double a = 0.5, b = 0.5;
        f = ((a * mse) + (b * pnsr)) / f;
return f; }
```

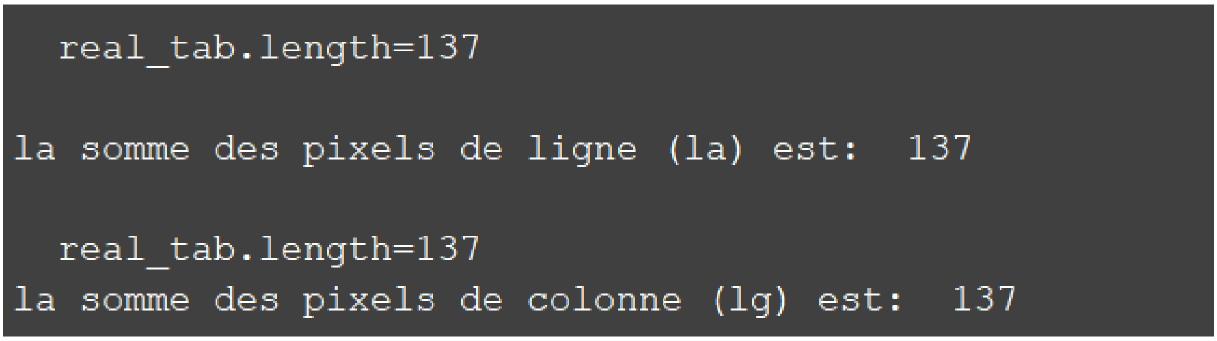
Figure A.8:code source of Fitness.

```
public static void mutation(Population Population_mov, int random_number) {
          Population population = new Population(la * lg);
        population = Population_mov;
         System. out.println ("random_number_avant_mutation_dans la position:  "+
random_number);
        for(int ii=0;ii<40;ii++){
             if(ii%4==0)
                 System. out.println (" ");
                 System. out.print (population.get_chromosome(2).get_genes(ii));}
          for (int j = 0; j <(population.chromosome.length / 40); j++) {
            for (int i = 0; i < population.get_chromosome(j).toString().length(); i++) {
                if (i == random_number) {


                    population.get_chromosome(j).mutation_bit(i);
                }}}
        System.out.println("");
                    System. out.println ("random_number_apris_mutation_dans la position:  "+
random_number);
               for(int ii=0;ii<40;ii++){
             if(ii%4==0)
                 System. out.println ("");
                 System. out.print (population.get_chromosome(2).get_genes(ii));}
```

Figure A.9 :code source of Mutation.

- EX: Implementation Hide and extracted with Projection and GA:
- Size of image with Projection =275*275



```
  real_tab.length=137

la somme des pixels de ligne (la) est:  137

  real_tab.length=137
la somme des pixels de colonne (lg) est:  137
```

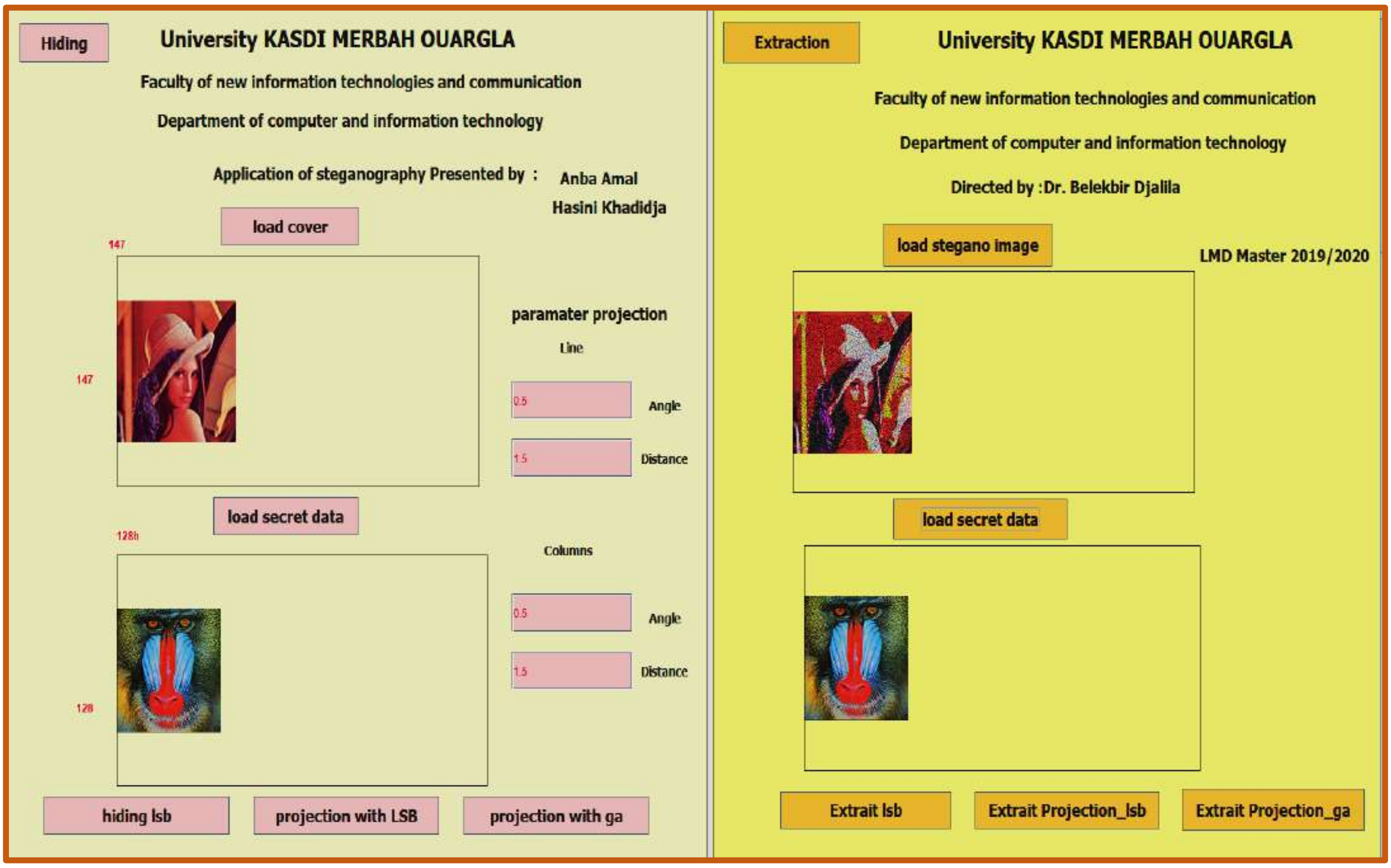

Hiding
University KASDI MERBAH OUARGLA
Faculty of new information technologies and communication
Department of computer and information technology
Application of steganography Presented by :
Anba Amal
Hasini Khadidja
load cover
147
147
paramater projection
Line
0.5
Angle
1.5
Distance
load secret data
128h
Columns
128
0.5
Angle
1.5
Distance
hiding lsb
projection with LSB
projection with ga
Extraction
University KASDI MERBAH OUARGLA
Faculty of new information technologies and communication
Department of computer and information technology
Directed by :Dr. Belekbir Djalila
load stegano image
LMD Master 2019/2020
load secret data
Extrait lsb
Extrait Projection_lsb
Extrait Projection_ga

Figure A.10: Implementation with Projection and GA.

- Quality metrics:

```
static double calcul_PSNR(BufferedImage image_C,BufferedImage image_S){

double psnr=0.0, RR=0.0;

double mse=calcul_mse(image_C,image_S);

RR=(255*255*3)/ mse;     psnr= (double) 10* Math.log10(RR); return  psnr; }
```

Figure A.11: code source of PNSR

```
static double calcul_mse (BufferedImage image_C,BufferedImage image_S){
double mse=0.0,mseR=0.0,mseG=0.0,mseB=0.0,sum_sq = 0.0;
int p1=0,R=0,G=0,B=0,p2=0,R1=0,G1=0,B1=0;
for (int i = 0; i <image_C.getHeight(); i++)    {
   for (int j = 0; j <image_C.getWidth() ; j++){
               p1 = image_C.getRGB(j, i) ;
               R = (p1>>16) & 0xff;
               G = (p1>>8) & 0xff;
               B =  p1 & 0xff;
               p2 = image_S.getRGB(j,i) ;
              R1 = (p2>>16) & 0xff;
              G1 = (p2>>8) & 0xff;
              B1 =  p2 & 0xff;
        mseR=mseR+((R-R1)*(R-R1));
        mseG=mseG+((G-G1)*(G-G1));
        mseB=mseB+((B-B1)*(B-B1));
}
}
sum_sq= mseR+mseG+mseB;
mse = (double) sum_sq / (double)( (double) 3 *(double)image_C.getWidth()*(double)image_C.getHeight());
return  mse;
}
```

Figure A.12: code source of MSE

```
public double  calcul_ad (BufferedImage image_C,BufferedImage image_S){
  double ad=0.0,sum=0.0,adR=0.0,adG=0.0,adB=0.0;int  p1,R,G,B,p2,R1,G1,B1;
    for (int i = 0; i <image_S.getHeight(); i++)    {
    for (int j = 0; j <image_S.getWidth() ; j++) {
                 p1 = image_C.getRGB(j, i) ;p2 = image_S.getRGB(j,i);
                R = (p1>>16) & 0xff;  R1 = (p2>>16) & 0xff;
                G = (p1>>8) & 0xff;G1 = (p2>>8) & 0xff;
                B =  p1 & 0xff;B1 =  p2 & 0xff;
        adR=adR+(R- R1); adG=adG+(G-G1);  adB=adB+(B-B1); }}sum=adR+adG+adB;
   ad =(double)( sum) /  ((double)3*(double)image_S.getWidth()*(double)image_S.getHeight());
    return ad;}
```

Figure A.13: code source of AD.

```
public double calcul_md (BufferedImage image_C,BufferedImage image_S){
  double sus=0.0,md = 0.0,sousR=0.0,sousG=0,sousB=0,a1=0.0,a2=0.0,a3=0.0;
  int  p1=0,R=0,G=0,B=0,p2=0,R1=0,G1=0,B1=0;
  p1=image_C.getRGB(0, 0);p2=image_S.getRGB(0,0);
  R = (p1>>16) & 0xff;R1 = (p2>>16) & 0xff; a1=Math.abs(R-R1);
  G = (p1>>8) & 0xff; G1 = (p2>>8) & 0xff;a1=Math.abs(G-G1);
  B =  p1 & 0xff;      B1 =  p2 & 0xff;a1=Math.abs(B-B1);
    for (int i = 0; i <image_C.getHeight(); i++)    {
    for (int j = 0; j <image_C.getWidth() ; j++){
                p1 = image_C.getRGB(j, i) ;p2 = image_S.getRGB(j,i);
                R = (p1>>16) & 0xff;  R1 = (p2>>16) & 0xff;
                G = (p1>>8) & 0xff;  G1 = (p2>>8) & 0xff;
                B =  p1 & 0xff;      B1 =  p2 & 0xff;
                sousR=Math.abs(R-R1);a1=Math.max(a1,sousR);
                sousG=Math.abs(G-G1);a2=Math.max(a2,sousG);
                sousB=Math.abs(B-B1);a3=Math.max(a3,sousB);       }}
     md=(double) (a1+a2+a3);  return md;}
```

Figure A.14: code source of MD.

```
public double calcul_nk(BufferedImage image_C,BufferedImage image_S){
   double tt=0.0,nk=0.0,foix=0.0,foixR=0.0,foixG=0.0,foixB=0.0,tr=0.0,tg=0.0,tb=0.0;
   int  p1=0,R=0,G=0,B=0,p2=0,R1=0,G1=0,B1=0;
   for (int i = 0; i <image_C.getHeight(); i++)    {
   for (int j = 0; j <image_C.getWidth() ; j++) {
           p1 = image_C.getRGB(j, i) ;p2 = image_C.getRGB(j,i);
            R = (p1>>16) & 0xff;  R1 = (p2>>16) & 0xff;
            G = (p1>>8) & 0xff;    G1 = (p2>>8) & 0xff;
            B =  p1 & 0xff;B1 =     p2 & 0xff;
            foixR=foixR+(R1*R);  foixG=foixG+(G1*G); foixB=foixB+(B1*B);
           foix =foixR+foixG+foixB;
       tr+=Math.pow(R, 2);tg+=Math.pow(G, 2);tb+=Math.pow(B, 2);
      }}
   tt =(double) tr+tg+tb;
   nk = (double)foix/ (double)tt;
   return nk;
}
```

Figure A.15: code source of NK.

```
int ber =(w*h*100)/(w1*h1);
    System.out.print("la valeure de BER est ==:"+ber);
//******************************
w,h=size of secret original.
w1,h1=size of secret befor extracted.
```

Figure A.16: code source of BER.

```
public double calcul_sc(BufferedImage image_C,BufferedImage image_S){
double op1 =0.0,op2 = 0.0,sc=0.0,scR=0.0,scG=0.0,scB=0.0,scR1=0.0,scG1=0.0,scB1=0.0;
   int  p1=0,R=0,G=0,B=0,p2=0,R1=0,G1=0,B1=0;
   for (int i = 0; i <image_C.getHeight(); i++)    {
   for (int j = 0; j <image_C.getWidth() ; j++) {
           p1 = image_C.getRGB(j, i) ;p2 = image_S.getRGB(j,i);
            R = (p1>>16) & 0xff;     R1 = (p2>>16) & 0xff;
            G = (p1>>8) & 0xff;       G1 = (p2>>8) & 0xff;
            B =  p1 & 0xff;          B1 =  p2 & 0xff;
           scR+=Math.pow(R, 2);scG+=Math.pow(G, 2);scB+=Math.pow(B, 2);
           op1 =scR+scG+scB;
           scR1+=Math.pow(R1, 2);scG1+=Math.pow(G1, 2);scB1+=Math.pow(B1, 2);
           op2 =scR1+scG1+scB1;
   }}sc=(double)op1/(double)op2;
 return sc;}
```

Figure A.17: code source of SC.

```
public double calcul_PMSE(BufferedImage image_C,BufferedImage image_S){
      int  p1=0,R=0,G=0,B=0,p2=0,R1=0,G1=0,B1=0;
       p1 = image_C.getRGB(0, 0) ;  R = (p1>>16) & 0xff;G = (p1>>8) & 0xff;B =  p1 & 0xff;
   for (int i = 0; i <image_C.getHeight(); i++)    {
   for (int j = 0; j <image_C.getWidth() ; j++) {
          p1 = image_C.getRGB(j,i);
            R1 = (p1>>16) & 0xff; R=Math.max(R,R1);
            G1 = (p1>>8) & 0xff; G=Math.max(G,G1);
            B1 =  p1 & 0xff;    B=Math.max(B,B1);
   }} double aa=Math.pow(Math.abs(R+G+B),2);
   double d=calcul_mse( image_C, image_S);
   return (double) d/ (double)aa;
}
```

Figure A.18: code source of PMSE.

```
static double calcul_nae (BufferedImage image_C,BufferedImage image_S){
int  sousR=0,sousG=0,sousB=0,sum_sq=0,nae=0,sr=0,sg=0,sb=0,sum_sq1=0;
int p1=0,R=0,G=0,B=0,p2=0,R1=0,G1=0,B1=0;
for (int i = 0; i <image_C.getHeight(); i++)    {
   for (int j = 0; j <image_C.getWidth() ; j++){
            p1 = image_C.getRGB(j, i) ;
            R = (p1>>16) & 0xff;
            G = (p1>>8) & 0xff;
            B =  p1 & 0xff;
              p2 = image_S.getRGB(j,i);
               R1 = (p2>>16) & 0xff;
               G1 = (p2>>8) & 0xff;
              B1 =  p2 & 0xff;
          sousR+= Math.abs((R-R1));
            sousG+=Math.abs((G-G1));
            sousB+=Math.abs((B-B1));
 sr=sr+Math.abs(R);
sg=sg+Math.abs(G);
sb=sb+Math.abs(B); }}
sum_sq= sousR+sousG+sousB;
System.out.print("la valeure de sum_sq==="+sum_sq);
sum_sq1= sr+sg+sb;
System.out.print("la valeure de sum_sq1==="+sum_sq1);
nae =  ( sum_sq )  / (sum_sq1 );
return nae;
}
```

Figure A.19: code source of NAE.

Annex

```
static double calcul_lmse(BufferedImage image_C,BufferedImage image_S){

intp1=0,R1=0,G1=0,B1=0,p2=0,R2=0,G2=0,B2=0,p3=0,R3=0,G3=0,B3=0,p4=0,R4=0,G4=0,B4=0,p5=0,R5=0,G5=0,B5
=0,

p6=0,R6=0,G6=0,B6=0,p7=0,R7=0,G7=0,B7=0,p8=0,R8=0,G8=0,B8=0,p9=0,R9=0,G9=0,B9=0,p10=0,R10=0,G10=0,B
10=0;

double LMSE=0.0,LMSE1=0.0,LMSE2=0.0,ob=0.0,oa=0.0;

for (int i=0;i<image_C.getHeight();i++){

   for (int j=0;j< image_C.getWidth();j++){

if ( (i != 0)&&(i != image_C.getHeight()-1)){

if ( (j != 0)&&(j !=  image_C.getWidth()-1)) {

//**********************************************************************

   p1 = image_C.getRGB(i+1,j) ;    p3 = image_C.getRGB(i,j+1) ;  p5 = image_C.getRGB(i,j) ;

   R1 = (p1>>16) & 0xff;         R3=(p3>>16) & 0xff;          R5=(p5>>16) & 0xff;

   G1 = (p1>>8) & 0xff;          G3 = (p3>>8) & 0xff;        G5 = (p5>>8) & 0xff;

    B1 =  p1 & 0xff;             B3 = p3 & 0xff;              B5 = p5& 0xff;

     p2 = image_C.getRGB(i-1,j) ;     p4 = image_C.getRGB(i,j-1) ;

     R2 = (p2>>16) & 0xff;         R4 = (p4>>16) & 0xff;

      G2 = (p2>>8) & 0xff;          G4 = (p4>>8) & 0xff;

      B2 = p2 & 0xff;              B4 = p4 & 0xff; ;

      //************************************************************************

   p6 = image_S.getRGB( i+1,j) ;    p7 = image_S.getRGB(i-1,j) ;  p8 = image_S.getRGB(i,j+1) ;

   R6 = (p6>>16) & 0xff;         R7=(p7>>16) & 0xff;          R8=(p8>>16) & 0xff;

   G6 = (p6>>8) & 0xff;          G7 = (p7>>8) & 0xff;        G8 = (p8>>8) & 0xff;

    B6 =  p6 & 0xff;             B7 = p7 & 0xff;              B8 = p8& 0xff;

     p9 = image_S.getRGB(i,j-1) ;     p10 = image_S.getRGB(i,j) ;

     R9 = (p9>>16) & 0xff;         R10 = (p10>>16) & 0xff;

      G9 = (p9>>8) & 0xff;          G10 = (p10>>8) & 0xff;

      B9 = p9 & 0xff;              B10= p10 & 0xff;


  ob =(((double )(R1+G1+B1)/(double)(3))+((double)(R2+G2+B2)/ (double)
(3))+((double)(R3+G3+B3)/(double)(3))+((double)(R4+G4+B4)/(double)(3)))-(4*((double)(R5+G5+B5)/(double)(3)));

  oa=(((double )(R6+G6+B6)/(double)(3))+((double)(R7+G7+B7)/ (double)
(3))+((double)(R8+G8+B8)/(double)(3))+((double)(R9+G9+B9)/(double)(3)))-
```

```
      ob =(((double )(R1+G1+B1)/(double)(3))+((double)(R2+G2+B2)/ (double)
(3))+((double)(R3+G3+B3)/(double)(3))+((double)(R4+G4+B4)/(double)(3)))-
(4*((double)(R5+G5+B5)/(double)(3)));

  oa=(((double )(R6+G6+B6)/(double)(3))+((double)(R7+G7+B7)/ (double)
(3))+((double)(R8+G8+B8)/(double)(3))+((double)(R9+G9+B9)/(double)(3)))-
(4*((double)(R10+G10+B10)/(double)(3)));

       LMSE1= LMSE1+(ob-oa);

       LMSE2= LMSE2+(ob*ob);

    }} }}

 LMSE = (double)(LMSE1)/(double)(LMSE2);

return LMSE;}
```

Figure A.20: code source of LMSE.

REFERENCES:

[1]. Moin Hamsa Mohamed, Moin Nadia Mohamed, Moin Chakib Mohamed, "An Optimized Genetic Algorithm-based Approach for Steganography " ,Iraqi Journal of Statistical Sciences 2011.

[2]: Mr. Jayesh Surana, Aniruddh Sansole, Bhavesh Joshi , Deepesh Sarma , Nilesh Choudhary "Steganography Techniues ", 2017 IJRDER /Volume 5, Issue 2/ISSN : 2321-9939.

[3]: M. Hussain and M. Hussain, "A Survey of video Steganography Techniques," Int. J. Adv. Sci.Technol., vol. 54, pp. 113–124, (2013).

[4]: Mehdi Hussain, Ainuddin Wahid Abdul Wahaba, Yamani Idna Bin Idrisa, Anthony T. S. Ho, Ki-Hyun Jung "Image steganography in spatial domain", march 2018.

[5]: M.Bendjeriou Ramzi , M.Arar Abdelhkim, ' une étude comparative entre la stéganographie JPEG et la stéganographie TCP/IP pour une meilleure techniques de dissimulation des données' , memiore fin d'etude 2016.2017.

[6]: Https://techdifferences.com/difference-between-steganography-and-cryptography.html , " Diffrence between Steganography and Cryptography ", 2018.

[7]: S.Divya,B.deepthy,D.sravanthi,"Cipher Text Design with Asymmetric RSA Algorithm"International Journal of Advanced Engineering Research and Science (IJAERS),2349-6495(P) | 2456-1908(O).

[8] : B. Li, J. He, J. Huang, Y.Q. Shi, ' A Survey on Image Steganography and Steganalysis', Journal of Information Hiding and Multimedia Signal Processing, 2011.

[9] : G.Kaur and A.Kochhar, "A Steganography Implementation based on LSB et DCT ", 2012.

[10] : P.Chen, H. Lin, "A DWT Based Approach For Image Steganography" ,Int .J. of Applied Sci , 2006.

[11] : H.Kaur and J. Rani, "A Survay On Different Techniques Of Steganography" , MATEC Web of Conferences ,2016.

[12]: Ishwarjot singh, J.p Raine, "Advance Data Scheme For Secret Data Hiding System Using Hop field et LSB", International journal of computer trends and technology , July 2013.

[13] :X. Zhang, S. Wang, 'Efficient Steganographic Embedding by Exploiting Modification Direction ', IEEE Communications Letters, 2006.

[14]: V.M. Potdar, E. Chang,' Grey level modification steganography for secret communication', in: Industrial Informatics, 2004.

[15]: Nadia moin mohamed, "The use of a genetic algorithm to develop a method for cloaking", college of science and mathematics, University of almosul, 28/06/2012.

[16]: Goldberg, D. E, "Genetic algorithms in search, optimization, and machine learning", (1989) .

[17]: Manik Sharm, "Role and Working of Genetic Algorithm in Computer Science ", Articale in International Journal of Computer Applications and Information Technolog, December 2012.

[18]: Dr Rajib Kumar, "Introduction To Genetic Algorithms ", Bhattach arjya professor Departement of Civil Engineering IIT Guwahati , 24/04/2015.

[19]: Suvarna Patil , Manisha Bhende "comparaison and analysis of Differnent Mutation Strategies to impove the performance of Genetic Algorithm , JICSIT , 2014

[20]:R.Hols mark and M.Hogder , "Modeling and prototyping of Network on Chip" , Sweden 2002.

[21]: Aura Conci , Luiz Brazil , Simone Becellar, Trueman Machenry ,"AES Cryptography in Color Image Steganography by Genetic Algorithms " , 'Computer Science Departement , Computer Institute , UFF', 2015.

[22]: Nitasha Soni , Dr Tapas kumar , " Study of Various Mutation Operators in Genetic Algorithms " , Lingaya's University , Faridabad , Nitasha Soni et Al , (IJCSIT) International Journal of Computer Science and Information Technologies , 2014.

[23]: Shen Wang, Bian yang, Xiamu Niu, "A Steganography Method based on Genetic Algorithm", Journal of Information Hiding and Multimedia Signal Processing, 01/01/2010.

[24]: Jasim Aissa Raya, Jasim Aissa Riham, Mohamed Solieman Inaam, "The Use of a Genetic Algorithms to Encode Gray and Image Data and Hide it in an Image ", Al-Rafidian Journal of Computational Sciences and Mathematics, Issue (10), 2013.

[25]: Mr.Himani Mehra , Turan kumar Sahu , Garima Tiwari , "Steganography using Genetic Algorithm along with Visual Cryptography for Wireless Network Application ", M. tech in

Communication System , Departement of Electronic and Communication Engineering , INDIA , Journal of Advanced Research in Copputer and Communication Engineering , 02 /2016.

[26]: Aayasha kausar and Deepty Dubey , " Secure Data Transmission 'Interogation of Genetic Algorithm and Visual Cryptography Technique '", Journal of computer and Information Technology Sciences , www.isca.in , www.isca .me ,12/04/2016.

[27]: R.Jaassim Essa ,N.A.Z.Abdullah , R.D.Al-dabbagh , " Steganography Technique using Genetic Algorithm "Iraqi Journal of Science , 01/2018.

[28]: S.Talbi, H.Azadi-Motlagh , "A Novel Technique for Steganography Method Based on Improved Genetic Algorithm Optimization in Spatial Domain ", Iranian Journal of Electrical and Electronic Engineering 02/06/2013.

[29]: SIDHARTHA SANKAR PRADHAN ،ATMASWAROOPA TRIPATHY , "Efficient Steganography Using LSB and Encryption Technique " , APEX INSTITUTE OF TECHNOLOGY & MANAGEMENT Pahala, Bhubaneswar, India , July 2012.

[30]: Mohammed Abdul Majid , Rossilawati Sulman ," An Improved LSB Image Steganography Technique Using Bit-Invers In 24 Bit Color Image" , Journal of Theoretical and Applied Information Technology , October 2015.

[31]: Wu Zahijun , " least Significant Bit " in information hidingin speech signals for secure communication , 2015

[32]: S.Elseoud , I.Eddin , "On The Information Hiding Technique Using Least Significant bits steganography " , international journal of computer science and information security ,2013.

[33]: Benazia Meriem , Kemassi Imane , "Solving Task Migration Problems by Optimizing the Crossover and Mutation probabilities using tabu search ",Thèse de Master Universit ´e de Technologie de Ouargla ,Soutenue le 02/07/2019.

[34]: Avcibas, I., Memon, N., Sankur, B.: 'Steganalysis using image quality metrics'. IEEE. Trans. Imag. Process., 2003, (12), pp. 221–229.

[35]: Cox, I.J., Miller, M.L., Bloom, J., Kalker, T., Fridrich, J.: 'Digital Watermarking and Steganography Second Edition', 2008.

References

[36]: Lin, Y., Abdulla, W.: 'Objective quality measures for perceptual evaluation in digital audio watermarking', Signal Process. IET., 2011, (5), pp. 623–631.

[37]: Ahmet, M., Paul, E., Fisher, S.: 'Image Quality Measures and Their Performance' IEEE. Trans. On. Commu., Dec.1995, 43, (12), pp. 2959-2965.

[38]: Soliman, M.M., Hassanien, A.E., Onsi, H.M.: 'An adaptive watermarking approach based on weighted quantum particle swarm optimization', Neural Comput. Appl. 2015, (27), pp. 469–481.