

**A PROJECT REPORT  
ON  
TOWARDS ROBUST IMAGE STEGANOGRAPHY**

*Submitted in partial fulfillment of requirements to*

**IT461 - Project Work**

**ACHARYA NAGARJUNA UNIVERSITY**

**For the award of the degree**

**Bachelor of Technology**

**in**

**INFORMATION TECHNOLOGY**

**By**

**Sivangula Narendra Kumar (Y20IT112)**

**Shaik Arshad Ayub (Y20IT103)**

**Shaik Meera Hussain (Y20IT106)**



**APRIL - 2024**

**R.V.R & J.C.COLLEGE OF ENGINEERING (Autonomous)  
NAAC A+ Grade, NBA Accredited (Approved by A.I.C.T.E)  
(AFFILIATED TO ACHARYA NAGARJUNA UNIVERSITY)**

**Chandramoulipuram: :Chowdavaram**

**GUNTUR - 522019**

**DEPARTMENT OF INFORMATION TECHNOLOGY**  
**R.V.R & J.C COLLEGE OF ENGINEERING**  
**(AUTONOMOUS)**



**BONAFIDE CERTIFICATE**

This is to certify that this project titled “**TOWARDS ROBUST IMAGE STEGANOGRAPHY**” is the bonafide work of **Sivangula Narendra Kumar (Y20IT112), Shaik Arshad Ayub (Y20IT103), Shaik Meera Hussain(Y20IT106)** who have carried out the work under my supervision, and submitted in partial fulfillment for the award of the degree, **B.TECH.** during the year **2023-2024.**

**Dr. B. Hemanth Kumar**  
Professor, Dept of IT

**Dr. A. Srikrishna**  
Prof. & HOD, Dept of IT

## ACKNOWLEDGEMENT

The successful completion of any task would be incomplete without a proper suggestion, guidance and environment. Combination of these three factors acts like backbone to our Project **“TOWARDS ROBUST IMAGE STEGNOGRAPHY”**.

We would like to express our gratitude to the Management of R.V.R&J.C COLLEGE OF ENGINEERING for providing us with a pleasant environment and excellent lab facility.

We regard our sincere thanks to our Principal, **Dr. K. Srinivas** for providing support and stimulating environment.

We are greatly indebted to **Dr. A. Srikrishna**, Professor and Head of the Department Information Technology, for her valuable suggestion during the course period.

We would like to express our special thanks of gratitude to our guide **Dr. B. HEMANTH KUMAR**, who helped us in doing the Project successfully.

SIVANGULA NARENDRA KUMAR (Y20IT112)

SHAIK ARSHAD AYUB (Y20IT103)

SHAIK MEERA HUSSAIN (Y20IT106)

# TABLE OF CONTENTS

Title	i
Certificate	ii
Acknowledgement	iii
List of Figures	vi
List of Tables	vii
Abstract	1
1. INTRODUCTION	2
1.1 Applications	3
1.2 Literature Review	5
1.3 Technology	7
1.4 Significance of Work	7
2. EXISTING ALGORITHM	8
2.1 F5- A Steganography Algorithm	8
2.1.1 Introduction	8
2.1.2 Methodology	8
2.1.3 Results	11
2.2 Embedding by Modification Direction	12
2.2.1 Introduction	12
2.2.2 Methodology	12
2.2.3 Results	14
2.3 Summary	15
3. TOWARDS ROBUST IMAGE STEGANOGRAPHY	16
3.1 Introduction	16

3.2 Proposed Work	17
3.3 Methodology	17
3.4 Summary	23
4. RESULTS AND DISCUSSION	24
4.1 Results	24
5. CONCLUSION	36
REFERENCES	37

## LIST OF FIGURES

<b>S.NO</b>	<b>DESCRIPTION</b>	<b>PAGENO</b>
2.1	Images used in F5 – Algorithm	10
2.2	Redundancy Rate Sequence Image	14
2.3	Embedding rate and highest embedding efficiency	14
3.1	Proposed Framework Block Diagram	16
4.1	Interface for Embedding & Retrieving Functions	24
4.2	Selection of Original Image	25
4.3	Embedding Secret Message into Image	26
4.4	Comparison of Original and Stego Image	26
4.5	Retrieving of Message	27
4.6	Selection of Stego Image	28
4.7	Retrieved Message	28
4.8	Embedding Secret File	29
4.9	Selection of Original Image	30
4.10	Selecting Secret File	31
4.11	Embedding Secret File	32
4.12	Retrieving the Secret Message	33
4.13	Selecting the Stego Image	34
4.14	Retrieved Message	35

## **LIST OF TABLES**

<b>S.NO</b>	<b>DESCRIPTION</b>	<b>PAGENO</b>
I	Connection and Comparison in F5-Algorithm	11
II	Comparison of Several JPEG files create with F5	12

## ABSTRACT

The behaviour of posting images on social network platforms is happened everywhere and every single second. Thus, the communication channels offered by various social networks have a great potential for covert communication. However, images transmitted through such channels will usually be JPEG compressed, which fails most of the existing steganographic schemes. In this paper, we propose a novel image steganography framework that is robust to such channels. In particular, we first obtain the channel compressed version (i.e., the channel output) of the original image. Secret data is embedded into the channel compressed original image by using any of the existing JPEG steganographic schemes, which produces the stego-image after the channel transmission. To generate the corresponding image before the channel transmission (termed as the intermediate image), we propose a coefficient adjustment scheme to slightly modify the original image based on the stego-image. The adjustment is done such that the channel compressed version of the intermediate image is exactly the same as the stego-image. Therefore, after the channel transmission, the secret data can be extracted from the stego-image with 100% accuracy. Various experiments are conducted to show the effectiveness of the proposed framework for image steganography robust to JPEG compression.



# CHAPTER 1

## INTRODUCTION

DATA hiding is a technique of embedding secrets into the digital media imperceptibly, which can be categorized into watermarking and steganography according to different applications. Watermarking is the process of marking the digital media for copyright protection [1]–[3], while steganography is mainly developed for covert communication [4]–[10]. Different types of media data are considered in the literature for steganography including text [5], image [6],[7], audio [8] and video [9], where the image steganography is the most popular.

The task of image steganography is to make tiny changes on the pixels either in the spatial domain or the transformed domain to carry sufficient secret information. Meanwhile, the statistical and visual features of the original image are preserved. Early image steganographic methods adjust the value of the pixel (or coefficient) either by following a specific statistical model or reducing the modification caused by data embedding [11]–[14]. In [14], the DCT coefficients are categorized into four bands with different embedding rates. The stego-images generated by these schemes can be easily detected using modern steganalysis tools [15]. Recent works on image steganography focus on syndrome trellis coding (STC) based data embedding [16]–[20]. In these schemes, different distortion functions are designed to measure the distortion caused due to the data embedding.

The STC seeks a solution to minimize such distortion, which achieves relatively good performance in terms of resisting the steganalysis tools. With the social networks more and more popular, the corresponding communication channels become a useful resource for covert communication. To take advantage

of such channels, it is necessary and urgent to develop image steganographic schemes that are robust to JPEG compression.

This project mainly focuses on image steganography framework that is able to resist the JPEG compression. Unlike the existing schemes, this framework does not require any error correction on the secret data, which guarantees the full recovery of the secret data from the stego-image.

## **1.1 APPLICATIONS**

Steganography technique can be implemented using several cover file formats. The secret information that are being embedded in the cover image also need not be necessarily a text secret message, it can also be a media like contents. Primarily, the steganography types are always defined based on the cover file being used in the embedding phase. The cover file used can be any media format files like image file, audio or video files. Rarely, steganography also takes advantage on the computer protocols and embeds the secret text in it. Steganography techniques can be classified as follows:

### **➤ TEXT STEGANOGRAPHY**

In text steganography technique, the cover file used would be in text format and the secret message being embedded in the cover file also would be predominantly of text type. Text steganography embedding strategies are based on number of lines, white spaces, capital letters, like used in radio communication's Morse code. Text steganography is a versatile and intriguing method for hiding secret messages within text files. By employing various embedding strategies and ensuring the security of the embedding key, it's possible to create covert communication channels that are challenging to detect.

## ➤ AUDIO STEGANOGRAPHY

Embedding secret messages into digital sound file is known as audio steganography. Steganography methods can embed messages in variety of sound files including .au, .mp3, .wav etc., Human Auditory System (HAS) are exploited in the process of audio steganography. Like the image steganography, this technique also embeds more data into a cover file. Audio steganography offers a unique and intriguing way to hide secret messages within digital sound files. By leveraging the characteristics of the Human Auditory System and employing various embedding techniques, it's possible to create covert communication channels that are challenging to detect.

## ➤ VIDEO STEGANOGRAPHY

Video steganography hides the given secret message into a cover object, whose file type is video. In this steganography, video frames are used as carrier source and this cover frame allows the sender to embed a secret message. Video steganography offers a powerful and versatile method for hiding secret messages within video files. By leveraging the characteristics of video frames and employing various embedding techniques, it's possible to create covert communication channels that are challenging to detect.

## ➤ PROTOCOL STEGANOGRAPHY

Protocol steganography is a new approach for data hiding, which is becoming popular in recent days. In this steganography, network layer protocol of TCP/IP suite are used for data hiding and not restricted only to these protocols. In network layer of OSI model, covert channels are used for data hiding. Covert channels breaches security policies of the

network system. These channels are either used to steal the information or communicate secret message over a network using the network protocol. Example protocols used in the protocol steganography are TCP, IPv4, NFS, CIFS etc.,

## ➤ IMAGE STEGANOGRAPHY

Image steganography is a technique in which hiding of a secret message implemented by taking the cover object as the image file. In this steganography, graphic images are commonly used as cover source and this cover image allows the user to embed a huge number of bits from the secret message. The major benefit of image steganography is that an attacker's focus isn't drawn by the cover image. Image steganography offers a versatile and effective method for hiding secret messages within digital images. By leveraging various embedding techniques and taking advantage of the rich visual data in images, it's possible to create covert communication channels that are challenging to detect.

## **1.2 LITERATURE REVIEW**

### **1.2.1 F5 – A Steganographic Algorithm**

The F5 steganographic algorithm is a data hiding technique that embeds secret messages into JPEG images. The F5 algorithm works by first converting the JPEG image into its discrete cosine transform (DCT) coefficients. These coefficients are then shuffled using a pseudorandom permutation, and the secret message is embedded into the shuffled coefficients. The shuffled coefficients are then converted back into the JPEG domain, and the stego-image is created. The

F5 algorithm uses a number of techniques to make it difficult to detect. First, the secret message is embedded into the least significant bits (LSBs) of the DCT coefficients. This makes it difficult to detect the changes to the coefficients, as the LSBs are often noisy and difficult to measure. Second, the F5 algorithm uses a pseudorandom permutation to shuffle the DCT coefficients before embedding the secret message. This further obfuscates the location of the changes to the coefficients. The F5 algorithm has been shown to be very secure against a variety of steganalysis attacks. The F5 algorithm is a powerful tool for hiding secret messages in JPEG images. It is secure, effective, and easy to implement. If you are looking for a way to hide secret messages in images, the F5 algorithm is a good choice.

### **1.2.2 Efficient Steganographic Embedding by Exploiting Modification Direction**

Efficient Steganographic Embedding by Exploiting Modification Direction is a steganographic algorithm that embeds secret messages into digital images by exploiting the direction of modification. The algorithm works by first converting the image into its discrete cosine transform (DCT) coefficients. These coefficients are then divided into blocks, and the secret message is embedded into the blocks by changing the least significant bits (LSBs) of the coefficients in a specific direction. The algorithm has a number of advantages over other steganographic algorithms. First, it is more efficient than other algorithms that do not exploit the direction of modification. This is because the algorithm only changes the LSBs of the coefficients in a specific direction, which means that it does not need to change as many coefficients as other algorithms. Second, the

algorithm is more secure than other algorithms that do not exploit the direction of modification. This is because the direction of modification can be used to hide the presence of the secret message, making it more difficult for steganalysis algorithms to detect. The algorithm has been shown to be effective in a number of experiments. In one experiment, the algorithm was able to embed a secret message of 1000 bits into a 256x256 image without significantly affecting the quality of the image.

### **1.3 TECHNOLOGY**

Steganography is the art of covered or hidden writing. The purpose of steganography is covert communication to hide the existence of a message from a third party. Image based steganography utilize the images as cover medium to hide data. A steganography technique that works for all cover images irrespective of their formats provides flexibility to the user. This algorithm adapts the best suitable cover image to hide the data by capacity estimation process. The technique provides stego-image with a great robustness. The output stego-image and the original cover image cannot be differed by naked eyes. This algorithm gives best results for image formats like JPEG.

### **1.4 SIGNIFICANCE OF WORK**

Steganography is the study and practice of concealing information within objects in such a way that it deceives the viewer as if there is no information hidden within the object. Simply put, it is hiding information in plain sight, such that only the intended recipient would get to see it.

## **CHAPTER 2**

### **EXISTING ALGORITHM**

#### **2.1 F5 – A STEGANOGRAPHIC ALGORITHM**

##### **2.1.1 Introduction**

F5 steganographic algorithm is a data hiding technique that embeds secret messages into JPEG images. The F5 algorithm works by first converting the JPEG image into its discrete cosine transform (DCT) coefficients. These coefficients are then shuffled using a pseudorandom permutation, and the secret message is embedded into the shuffled coefficients. First, the secret message is embedded into the least significant bits (LSBs) of the DCT coefficients. This makes it difficult to detect the changes to the coefficients, as the LSBs are often noisy and difficult to measure. Second, the F5 algorithm uses a pseudorandom permutation to shuffle the DCT coefficients before embedding the secret message. The F5 algorithm has been shown to be very secure against a variety of steganalysis attacks. The F5 algorithm is a powerful tool for hiding secret messages in JPEG images.

##### **2.1.2 Methodology**

This paper presents efficient embedding of message into the coefficients that are obtained by performing Discrete Cosine Transform (DCT) on JPEG image. The message is embedded in the LSB bits of the DCT coefficients, this is because the LSB bits have the least impact on the state of the image.

This method involves two phases:

- 1) Embedding Phase
- 2) Extraction Phase

### **2.1.2.1 Embedding Phase**

The first step is to convert the image into its discrete cosine transform (DCT) coefficients. This is done by dividing the image into blocks of 8x8 pixels, and then calculating the DCT of each block. The DCT coefficients are then shuffled using a pseudorandom permutation. The secret message is embedded into the DCT coefficients using matrix encoding. This is done by creating a matrix that maps each bit of the secret message to a change in the DCT coefficients. The changes to the DCT coefficients are then spread out using permutative straddling. This ensures that the changes are not easily detected by steganalysis algorithms. The shuffled DCT coefficients are then converted back into the JPEG domain. This is done by inverting the DCT transformation. The stego-image is then created by saving the shuffled DCT coefficients in a JPEG file. If we take a secret message of 100 bits long. We can create a code word for the secret message by dividing it into blocks of 8 bits. Each block of 8 bits will be represented by a single number from 0 to 255. We can then use a matrix to map each number to a change in the DCT coefficients. For example, the number 0 might be mapped to a change of +1 in the first DCT coefficient, the number 1 might be mapped to a change of -1 in the second DCT coefficient, and so on.

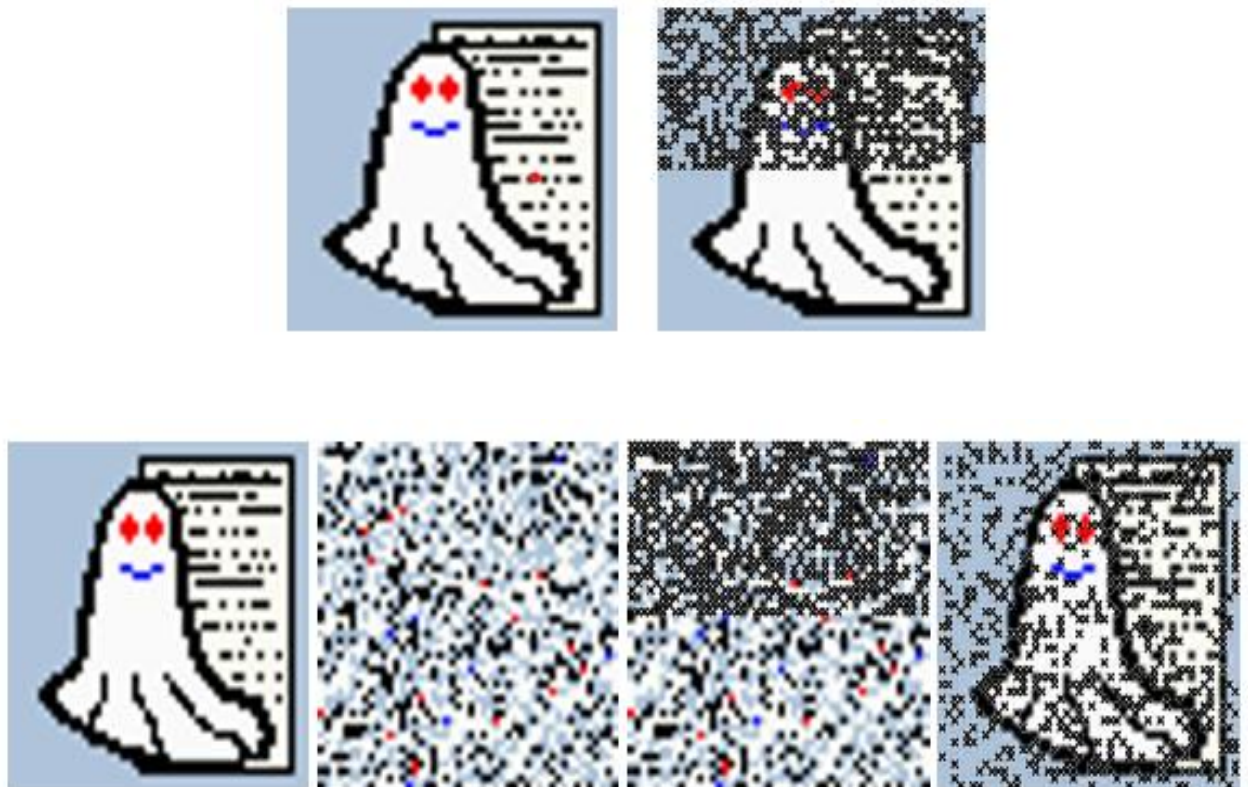
### **2.1.2.2 Extraction Phase**

In the Extraction phase, the secret message can be extracted from the stego-image by reversing the embedding process. The stego-image is converted into its DCT coefficients, and the secret message is extracted using matrix decoding. The secret message is then reassembled from the blocks. By considering the above calculated DCT coefficients and obtaining the Coefficients matrix we can embed data. Once we have the matrix, we can embed the secret message into the DCT coefficients by changing the DCT coefficients according to the matrix. For example, if the first bit of the secret message is 0, we would change the first DCT



coefficient by +1. If the second bit of the secret message is 1, we would change the second DCT coefficient by -1, and so on. After we have embedded the secret message, we can spread out the changes using permutative straddling. This is done by randomly shuffling the DCT coefficients before saving the stego-image. This ensures that the changes are not easily detected by steganalysis algorithms. The secret message can be extracted from the stego-image by reversing the embedding process. The stego-image is converted into its DCT coefficients, and the secret message is extracted using matrix decoding. The secret message is then reassembled from the blocks.

Also, in F5- A Steganographic Algorithm if there is Continuous embedding of data then the changes will get concentrated at a particular area. So, by using a mechanism known as Permutative Straddling the changes get scattered. Fig 2.1 shows the message images used in this paper.



**Fig 2.1: Images Used in the Paper**

### 2.1.3 Results and Conclusion

Table I shows the connection between change density and embedding rate. This is if the embedding rate increased the density of image gets increased. In Table II we are comparing several JPEG images created using F5 Algorithm.

Many steganographic algorithms offer a high capacity for hidden messages, but are weak against visual and statistical attacks. Tools withstanding these attacks provide only a very small capacity. Matrix encoding and permutative straddling enable the user to decrease the necessary number of steganographic changes and to equalise the embedding rate in the steganogram. F5 accomplishes a steganographic proportion that exceeds 13 % of the JPEG file size. Please understand this result as a friendly provocation for security analysts. On the other hand, F5 is able to decrease the embedding rate arbitrarily. The software with its source code is public.

**Table I: Connection between change density and embedding rate**

<b>k</b>	<b>n</b>	<b>Change Density</b>	<b>Embedding Rate</b>	<b>Embedding Efficiency</b>
1	1	50.00%	100.00%	2
2	3	25.00%	66.67%	2.67
3	7	12.50%	42.86%	3.43
4	15	6.25%	26.67%	4.27
5	31	3.12%	16.13%	5.16
6	63	1.56%	9.52%	6.09
7	127	0.78%	5.51%	7.06
8	255	0.39%	3.14%	8.03
9	511	0.20%	1.76%	9.02

**Table II: Comparison of several JPEG files created with F5**

File name	File Size (bytes)	Embedded Size(bytes)	Ratio embedded to steganogram size	Embedding efficiency	Quantiser quality
expo.bmp	15,62,030	0	(carrier medium)	-	-
expo80.jpg	1,29,879	0	-	-	80%
ministeg.jpg	1,29,760	213	0.20%	3.8	80%
maxisteg.jpg	1,15,685	1548000.00%	13.40%	1.5	80%
expo75.jpg	1,14,712	0	-	-	75%

## **2.2 EFFICIENT STEGANOGRAPHIC EMBEDDING BY EXPLOITING MODIFICATION DIRECTION**

### **2.2.1 Introduction**

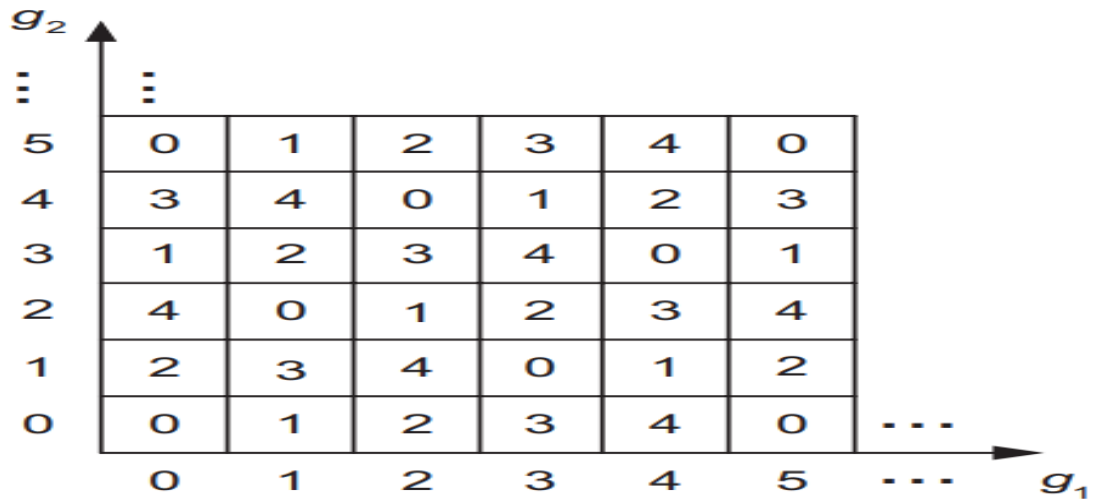
The main aim of steganography is to send a message through some innocuous carrier. The message to be sent could be a text, an image or it can be an audio file. Steganography techniques prevent the fact that a secret message is being sent at all. Digital stenography allows changes to be made to what are known as digital carriers such as images or sounds. The changes represent the hidden message, but results if successful in no discernible change to the carrier.

### **2.2.2 Methodology**

Cryptography and steganography are different. Cryptographic techniques can be used to scramble a message so that if it is discovered it cannot be read. If a cryptographic message is discovered it is generally known to be a piece of hidden information, but it is scrambled so that it is difficult or impossible to understand

and de-code. Steganography hides the very existence of a message. It is a good practice to combine the encryption and steganographic techniques by encrypting a message using cryptography and then hiding the encrypted message using steganography. The resulting stego-image can be transmitted without revealing that the secret information has been exchanged. Furthermore, even if an attacker were to defeat the steganographic technique and detect the message from the stego-image, he would still require the cryptographic decoding key to decipher the encrypted message.

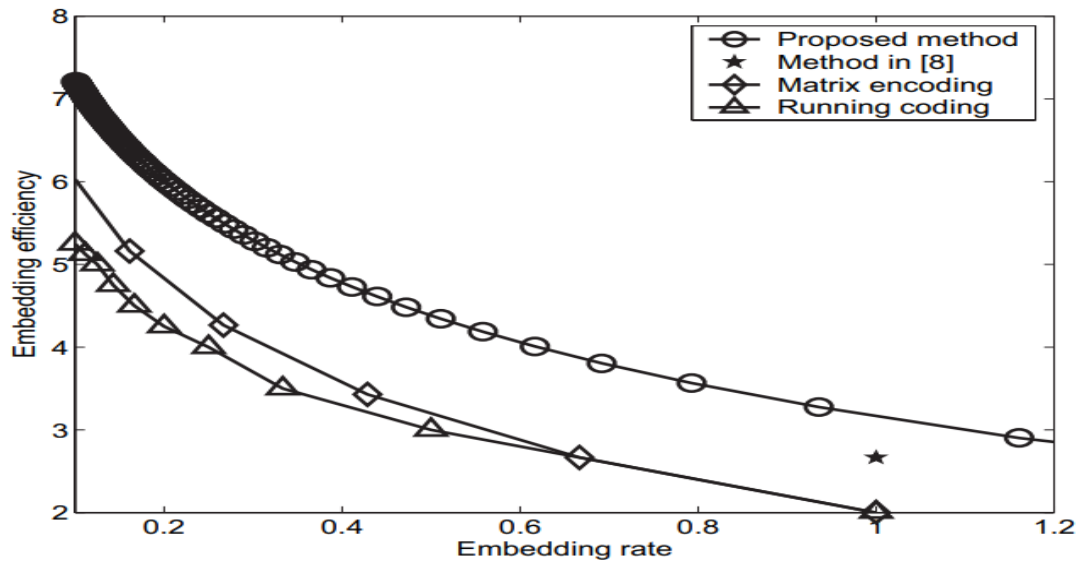
The main idea of the proposed steganographic method is that each secret digit in a  $(2n + 1)$ -ary notational system is carried by  $n$  cover pixels, where  $n$  is a system parameter, and, at most, only one pixel is increased or decreased by 1. Actually, for each group of  $n$  pixels, there are  $2n$  possible ways of modification. The  $2n$  different ways of alteration plus the case in which no pixel is changed form  $(2n + 1)$  different values of a secret digit. Before data-embedding, a data-hider can conveniently convert a secret message into a sequence of digits in the notational system with an odd base  $(2n + 1)$ . If the secret message is a binary stream, it can be segmented into many pieces with  $L$  bits, and the decimal value of each secret piece is represented by  $K$  digits in a  $(2n + 1)$ -ary notational system, where  $L = \lceil K * \log_2 (2n + 1) \rceil$ . For example, the binary sequence (1101 0110 1001) can be expressed as (23 11 14) in a 5-ary notational system where  $L = 4$  and  $K = 2$ . Thus, the redundancy rate in the  $(2n+ 1)$  - ary sequence is shown in fig. 2.2.



**Fig 2.2: Redundancy rate in the  $(2n+1)$ - ary sequence**

### 2.2.3 Results and Conclusion

It has been shown that the matrix encoding method is derived from the binary Hamming coding.



**Fig 2.3: Embedding rate and highest embedding efficiency**

Two parameters are used as the performance metrics: embedding efficiency  $E$  that is a ratio between the number of embedded bits and the distortion energy

caused by data embedding, and embedding rate  $R$  that is the number of secret bits embedded in each cover pixel. Performance comparison has been made among the proposed EMD method, the running coding, the matrix encoding. The results in Fig 2.3 show that, at any given embedding rate, the proposed method has the highest embedding efficiency, resulting in the least distortion and best security. The distortion function makes it possible for embedding the secret data into the image pixels by calculating the amount of the distortion to be happened. Using that function the secret data will be embedded into all pixel values making it uniformly embedding. Uniform embedding revisited distortion makes it possible for generating the stego-images with lower distortion rate.

## **2.3 SUMMARY**

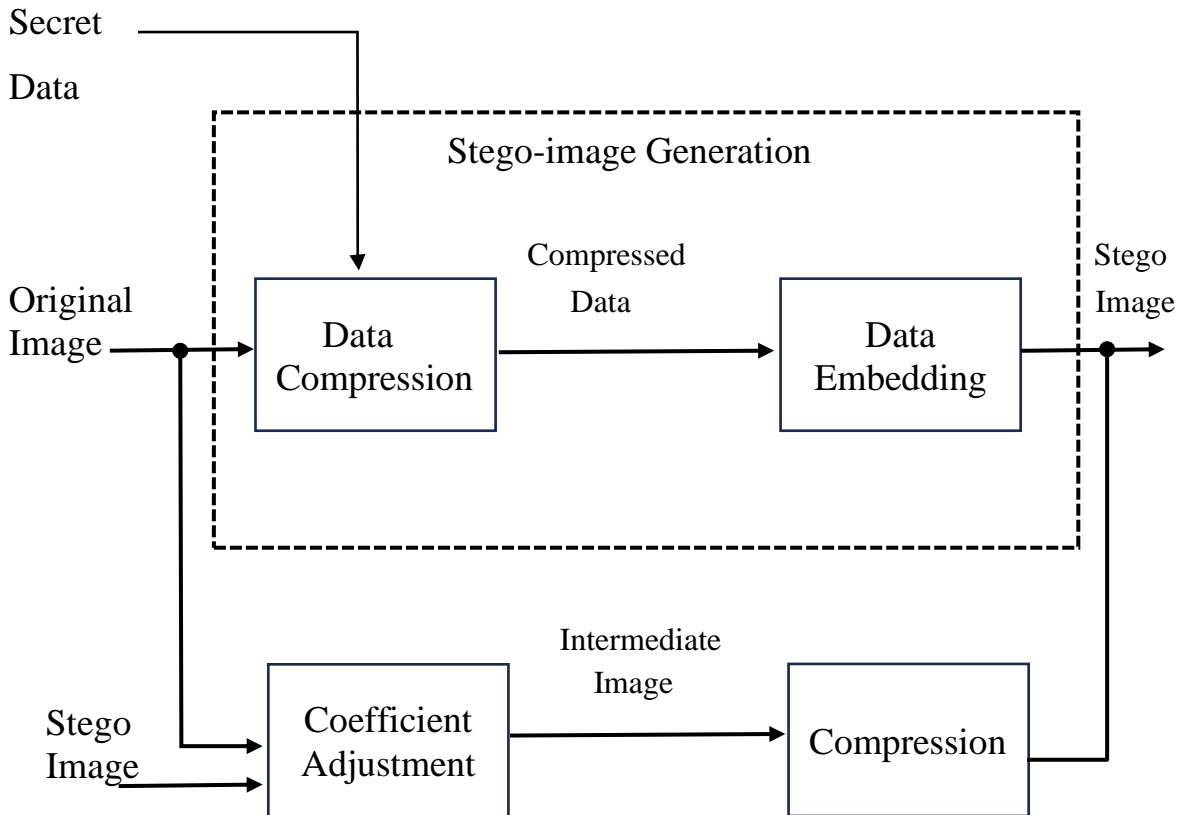
We have analysed the performance of F5 A steganography algorithm and Efficient exploiting by modification direction in detail of accuracy with JPEG images. The Efficient Embedding by Modification Direction framework has high capability of embedding secret data into images compared to F5 Steganography Algorithm. The main metric that differ both the frameworks is the amount of distortion that is created after embedding the secret data into images. Here the distortion rate after using Efficient Embedding by Modification Direction is least compared to F5 algorithm. Because the calculation will be done for selecting the direction of embedding, means on which direction the data is to be embedded in order to avoid high distortion rate. Also F5 has lower accuracy while retrieving the secret data. In case of Efficient Embedding by Modification Direction the accuracy is very high.

## CHAPTER 3

### TOWARDS ROBUST IMAGE STEGANOGRAPHY

#### 3.1 INTRODUCTION

The flowchart of our proposed framework is illustrated in Fig 3.1. Given an original image and the secret data to be hidden, our purpose is to generate an intermediate image whose channel compressed version is exactly the same as the stego-image. To this end, we first obtain the stego-image by data embedding on the channel compressed original image using any of the existing JPEG steganographic schemes. Then, we propose a coefficient adjustment scheme to produce the intermediate image based on the stego-image and the original image. This scheme ensures that the channel compressed version of the intermediate image is exactly the same as the stego-image.



**Fig 3.1: proposed framework for robust image steganography**

## 3.2 PROPOSED WORK

The proposed algorithm consists of Two procedures:

- i. Stego-Image Generation Partition schemes
- ii. Coefficient Adjustment

## 3.3 METHODOLOGY

### **Stego-Image Generation:**

Let's assume the quality factor of the JPEG compression of the communication channel is  $Q_c$  (within the range of  $[1,100]$ ), which can be obtained according to the image received from the output of the channel. Most of the time, the original image is already JPEG compressed, we denote the corresponding quality factor as  $Q_o$ . Given the quantized DCT coefficients of an original JPEG image, the communication channel first performs an inverse DCT quantization according to  $Q_o$  to estimate the original DCT coefficients, which are then quantized again according to  $Q_c$ .

The quantized DCT coefficient located at  $(i, j)$  after the channel compression is computed as  $C(i, j) = \text{round}[O(i, j) * Mo(i, j) / Mc(i, j)]$ , where  $\text{round}(\cdot)$  is the rounding operation,  $O(i, j)$  is the quantized DCT coefficient of the original image at  $(i, j)$ ,  $Mo(i, j)$  and  $Mc(i, j)$  are the corresponding quantization steps at  $(i, j)$  for quality factors  $Q_o$  and  $Q_c$ , respectively. Once the channel compressed original image is available, we generate the stego-image by embedding the secret data into this image using any of the existing JPEG steganographic schemes. In our implementation, we adopt two popular JPEG steganographic schemes: the J-UNIWARD [17] and UERD [18], for the generation of stego-images. These schemes incorporate the STC framework with different distortion functions.

The J-UNIWARD measures the distortion by a sum of relative changes between the original and stego-images in the wavelet domain, the UERD computes the



distortion by multiplying the quantization steps and the energy of the DCT block. Sometimes, the original image might be uncompressed. In such a case, we perform JPEG compression on the original image using  $Q_0 = 100$  to obtain an original JPEG image. Then, we obtain the channel compressed original image according to  $Q_c$  for generating the stego-image. The stego-image is eventually the image received from the output of the communication channel for covert communication. Upon receiving the stego-image, the secret data can be extracted based on the data extraction of the JPEG steganographic scheme adopted in the framework.

➤ **Universal Wavelet Relative Distortion (UNIWARD):**

1. Algorithm takes cover image and secret data to be embedded as input.
2. The cover image is preprocessed by applying JPEG compression. This can be done by performing the Discrete Cosine Transform (DCT) on each block.

$$F(u, v) = \frac{1}{4} C(u) C(v) \left[ \sum_{x=0}^7 \sum_{y=0}^7 f(x, y) \cos\left(\frac{(2x+1)u\pi}{16}\right) \cos\left(\frac{(2y+1)v\pi}{16}\right) \right]$$

- $F(u, v)$  represents the DCT coefficient at frequency position  $(u, v)$ .
- $C(u)$  and  $C(v)$  are normalization factors that depend on  $u$  and  $v$ .
- $f(x, y)$  represents the pixel intensity at position  $(x, y)$  of the block

3. After obtaining the DCT coefficients, we perform Quantization. Quantization involves dividing the DCT coefficients with Quantization matrix.

**Note:** For a particular quality factor, the Quantization matrix is Fixed

$$\text{Quantization Coefficient} = \frac{F(i,j)}{\text{Quantization Matrix}(i,j)}$$

4. Next the LSB'S of the Quantized Coefficients are modified to embed the secret information.

5. Next **Inverse Quantization** is performed. This involves multiplying each Quantized coefficient with corresponding value in quantization matrix.

$$\text{Inverse Quantization} = \text{Quantized Coefficient}(i,j) * \text{Quantization Matrix}(i,j)$$

6. Next step is to apply inverse DCT (Discrete Cosine Transform) to reconstruct the image in spatial domain.

$$f(x,y) = \frac{1}{4} \sum_{u=0}^7 \sum_{v=0}^7 C(u) C(v) F(u,v) \cos\left(\frac{(2x+1)u\pi}{16}\right) \cos\left(\frac{(2y+1)v\pi}{16}\right)$$

7. The modified cover image, known as the stego-image, is generated and can be saved or transmitted.

8. Finally, a stego-image is resulted which contains secret data. Here the image is converted from spatial domain to frequency domain.

9. Frequency domain makes it possible for manipulating the pixel values directly. Again, after embedding the secret data the image is converted back into spatial domain.
10. By using the stego-image we can retrieve the secret data by performing the steps in reverse order.

➤ **Uniform Embedding Revisited Distortion (UERD):**

1. Algorithm takes cover image and secret data to be embedded as input.
2. Convert the cover image into Discrete Cosine Transform (DCT). And divide the DCT coefficients into two groups: DC coefficients and AC coefficients.

**DC coefficients:** DC coefficients are the low frequency components of the image. Modifying the DC coefficients will not produce any distortion in the image.

$$DC(i, j) = DCT(i, j)[0, 0]$$

Here, DCT (i, j) [0,0] represents the top-left element of the DCT.

**AC coefficients:** AC coefficients are the high frequency components of the image. Modifying the AC coefficients will produce distortions in the image.

$$AC(i, j, k, l) = DCT(i, j)[k, l]$$

Here, DCT(i,j)[k,l] represents the (k,l)-th element of the DCT.

3. Now choose a distortion function that determines how much noise is added to the DC coefficients.

$$f(x) = x * \left( 1 - e^{\left(\frac{-x^2}{\sigma^2}\right)} \right)$$

Where, x is the DC coefficient &  $\sigma$  is Standard deviation.

4. For each DC coefficient, add noise to the coefficient using the distortion function.

$$y = x + f(x) * N(0, \sigma^2)$$

\* y is the new DC coefficient & x is original DC

\* f(x) distortion function,  $N(0, \sigma^2)$  random variable

5. Convert the DCT coefficients back to the spatial domain.
6. Output the Stego-Image.

### **Coefficient Adjustment:**

Generate an intermediate image before the channel transmission such that its channel compressed version is exactly the stego-image. To do so, the below steps are followed in order to achieve coefficient adjustment.

**Step 1:** First, we have to generate the intermediate image. For generating the intermediate image, we add each pixel of original image with some extra noise.

$$I(i, j) = O(i, j) + \alpha(i, j)$$

Where,  $I(i, j)$  represents Intermediate image and  $O(i, j)$  as Original image.

**Step 2:** Calculate the extra data to be added for the original image.

$$\alpha(i, j) = \underset{\alpha}{\operatorname{argmin}} \left| [O(i, j) + \alpha] \cdot \frac{Mo(i, j)}{Mc(i, j)} - S(i, j) \right|$$

**Step 3:** Here in the process of calculating the extra data we are using the stego-image. So, for calculating the pixel values of stego-image we use the below equation.

$$S(i, j) = \operatorname{round} \left[ I(i, j) * \frac{Mo(i, j)}{Mc(i, j)} \right]$$

Where,  $Mo$  represents Quantization matrix for Original Image and  $Mc$  represents Quantization matrix for Communication Channel.

After the adjustment, we are able to generate the intermediate image (which is a JPEG image) by using the same quantization table as the original JPEG image, where  $I(i, j)$  are the quantized DCT coefficient. According to the lemma, we can

always produce an intermediate image whose channel compressed version is exactly the same as the stego-image. It should be noted that our proposed scheme is not robust when the coefficients of the intermediate image are changed by a middleman. In such a case, the channel compressed version of the intermediate image may not be exactly the same as the stego-image, which will result in incorrect data extraction.

### **3.4 SUMMARY**

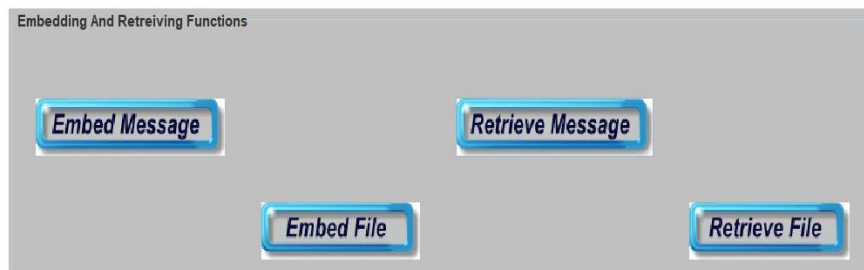
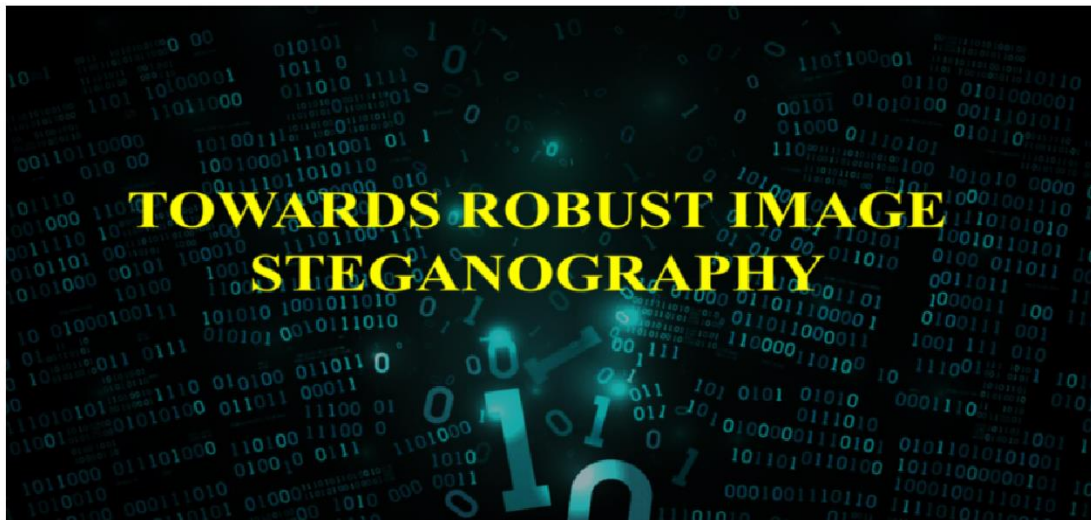
In this paper, we consider a cover image and first it made as a stego-image by using the steganographic schemes. Upon getting the stego-images these are transmitted through the channel in which they get compressed. By putting them in consideration we generate the intermediate images by the coefficient adjustment scheme proposed in the paper. This will modify the LSB bits of the image. We will embed the secret message into the LSB bits of the Coefficients. These coefficients are obtained by performing the Discrete Cosine Transform (DCT) on the images. Due to this embedding in LSB bits of the image there will be no large impact on the image features. After getting the intermediate images we can transmit them through the channels. As we have embedded the secret message using the proposed methods in the paper the probability of detection of secret message is low. So, the concept of Coefficient Adjustment works efficiently in the Image Steganography.

## CHAPTER 4

### RESULTS AND DISCUSSIONS

#### 4.1 Results

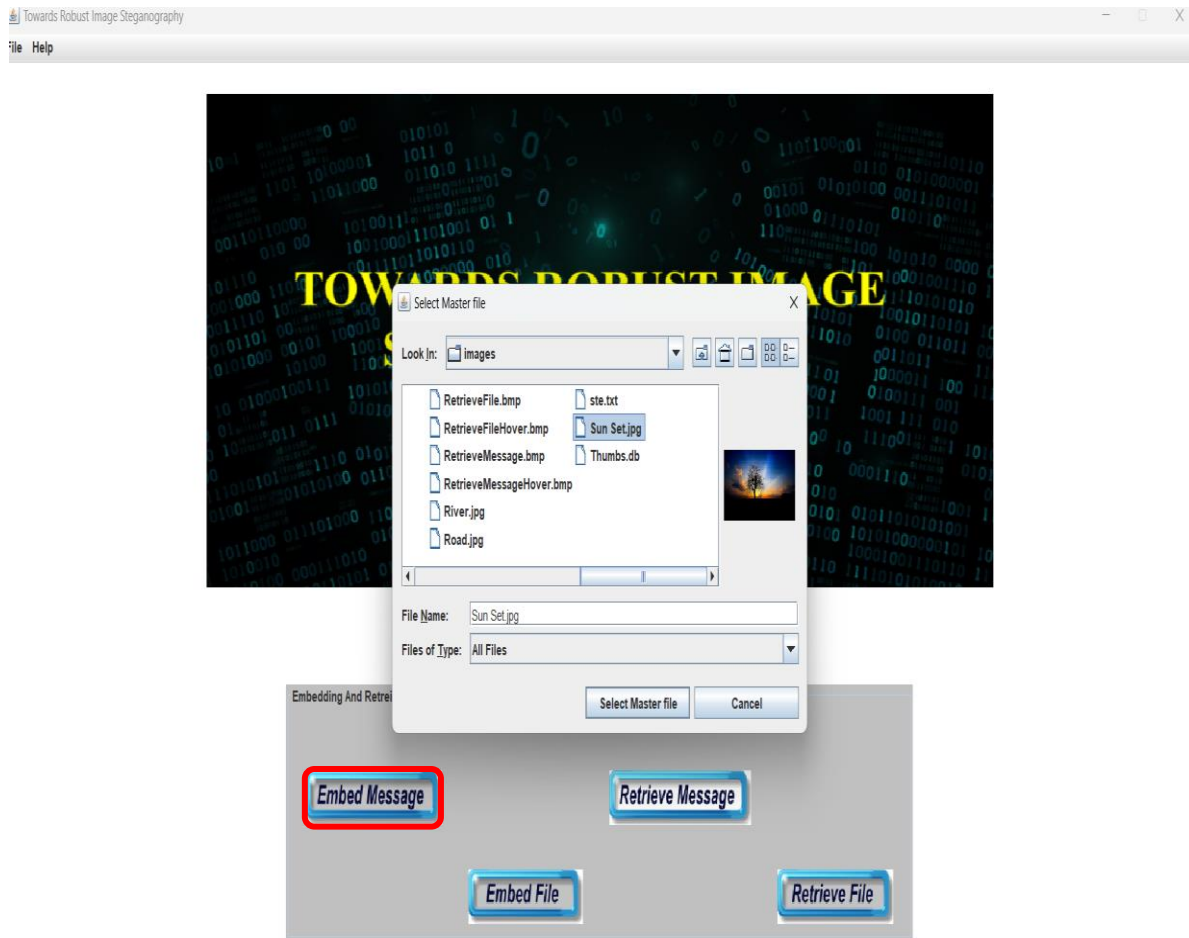
##### 4.1.1 USER INTERFACE



**Fig 4.1: Interface for Embedding and Retrieving Functions.**

We can embed secret message or we can embed a file that contains the secret information. Fig 4.1 shows the interface that contains the available Embedding and Retrieving Functions.

## 4.1.2 SELECTING THE ORIGINAL IMAGE

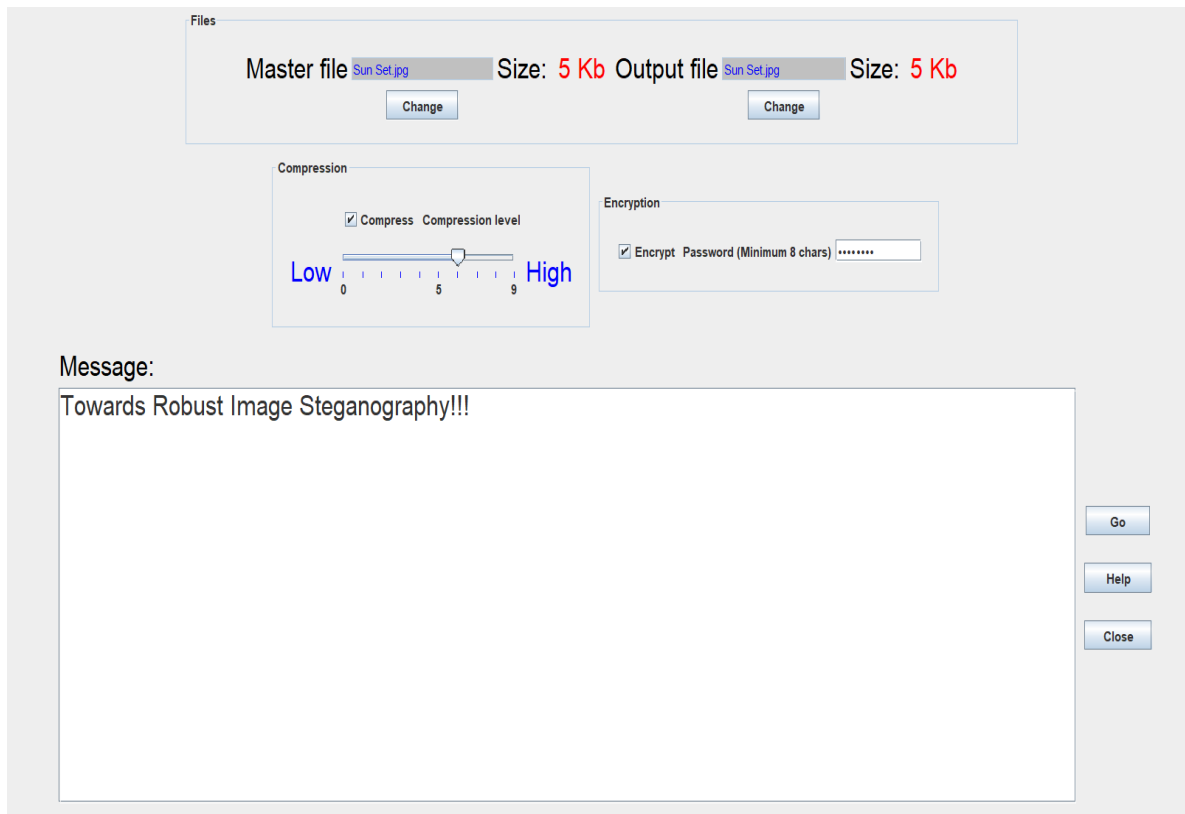


**Fig 4.2: Selection of Original Image.**

Here for embedding the secret information we need to select the original image on to which the secret information has to be embedded. The fig 4.2 depicts the selection of the original image and initiating the embedding process. Here at this instant of time we are using the image “SunSet.jpg”. Also, before embedding the size of the image is 6159 Bytes.



### 4.1.3 EMBEDDING THE MESSAGE INTO IMAGE



**Fig 4.3: Embedding secret message into image.**

Here we embedded the secret message. We gave secret message as **“Towards Robust Image Steganography!!!”**. Also, we can set the compression level between 0 to 9. And we can also use password protection while retrieving. The original and Stego image are shown in Fig 4.4.



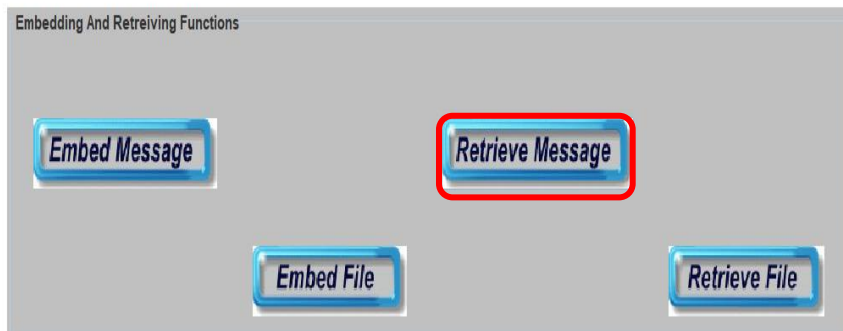
Original Image ( 6159 Bytes )



Stego Image ( 6336 Bytes )

**Fig 4.4: Comparison of Original and Stego Image.**

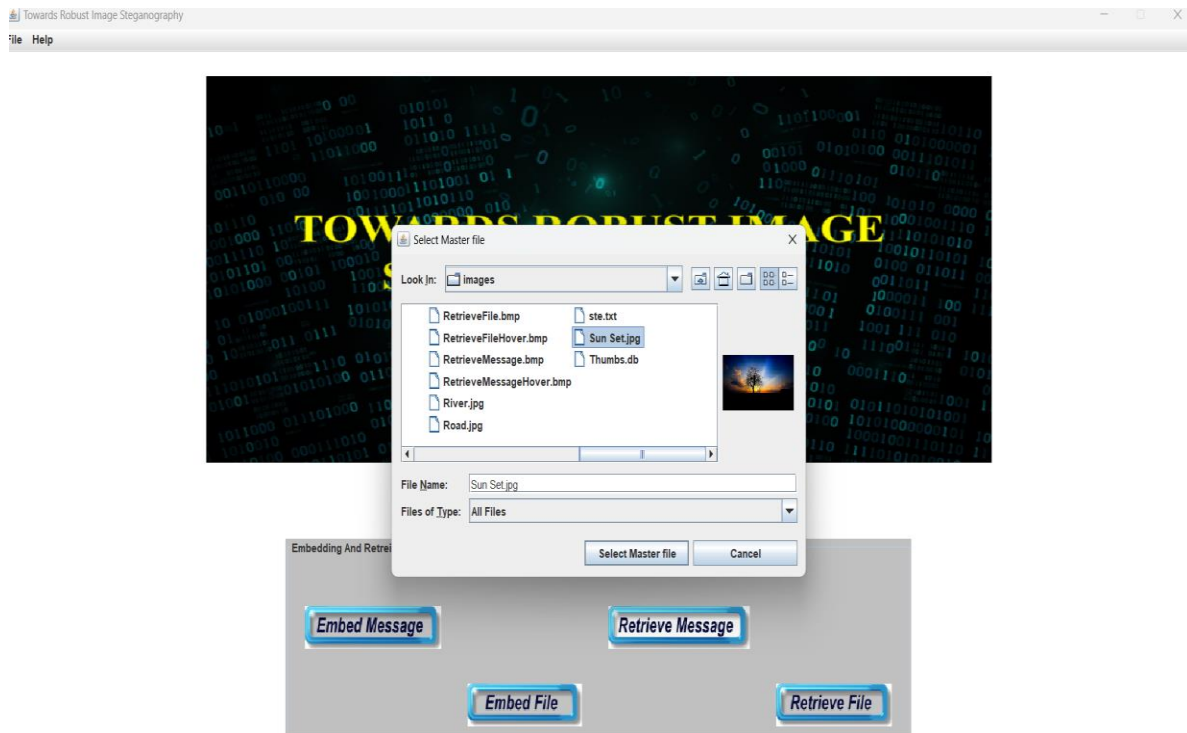
#### 4.1.4 RETRIEVING THE MESSAGE FROM STEGO IMAGE



**Fig 4.5: Retrieving of Message.**

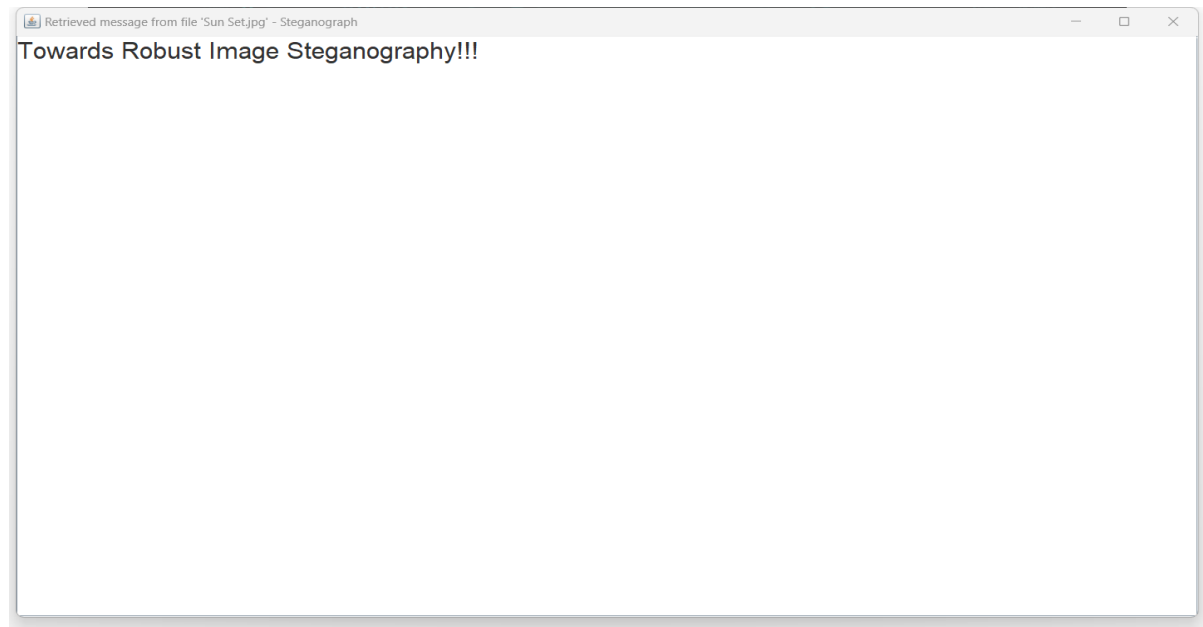
We successfully completed the embedding process and generated a stego-image that is carrying the secret information. Now we need to retrieve the embedded information from the stego-image. Fig 4.5 depicts the initiation process of the Retrieving the message.

## 4.1.5 SELECTING THE STEGO IMAGE



**Fig 4.6: Selection of Stego Image.**

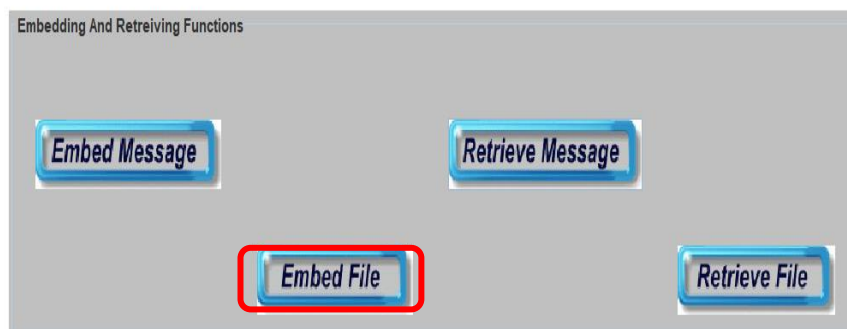
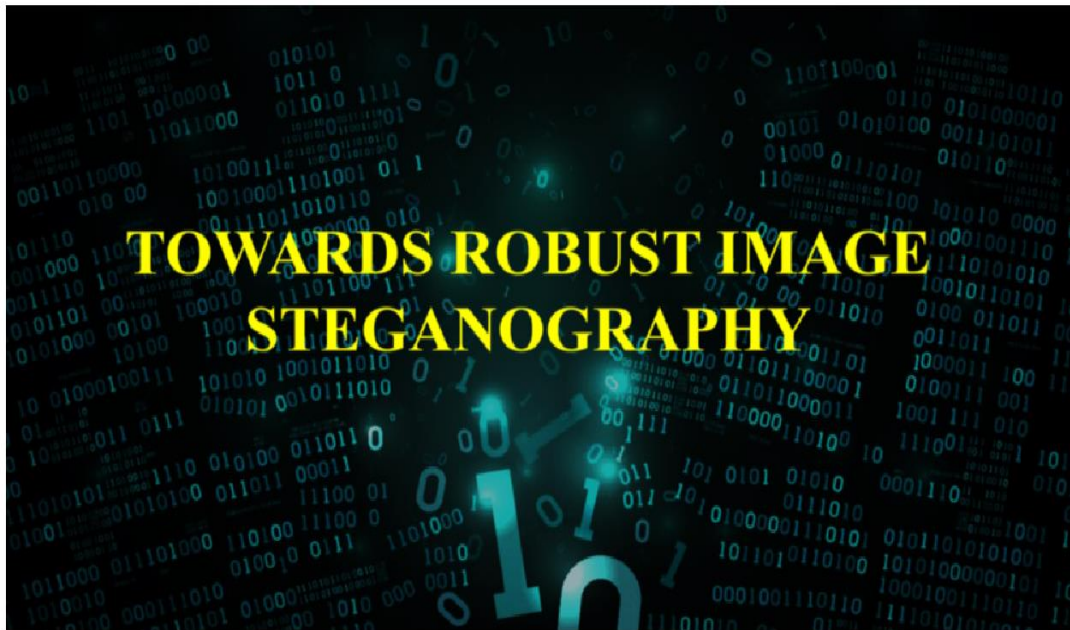
## 4.1.6 RETRIEVED MESSAGE



**Fig 4.7: Retrieved Message.**

Finally, the we retrieved the embedded secret message with 100% accuracy.

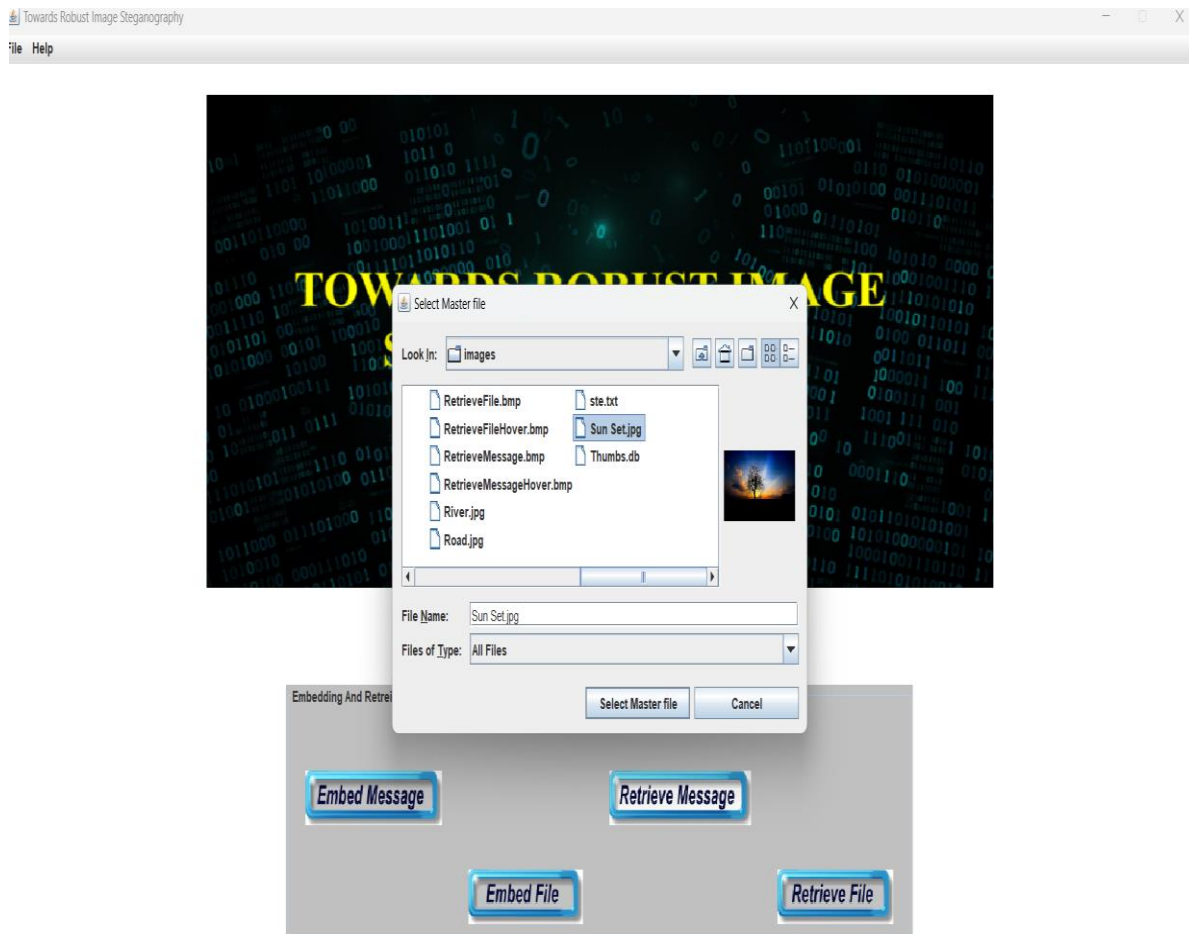
#### 4.1.7 EMBEDDING SECRET DATA AS A FILE



**Fig 4.8: Embedding a Secret File**

We can embed secret file that contains the secret information. Fig 4.8 shows the interface that contains the available Embedding and Retrieving Functions.

## 4.1.8 SELECTING THE ORIGINAL IMAGE



**Fig 4.9: Selection of Original Image.**

Here for embedding the secret information we need to select the original image on to which the secret information has to be embedded. The fig 4.9 depicts the selection of the original image and initiating the embedding process. Here at this instant of time we are using the image “**SunSet.jpg**”. Also, before embedding the size of the image is 6336 Bytes.

### 4.1.9 SELECTING THE SECRET FILE

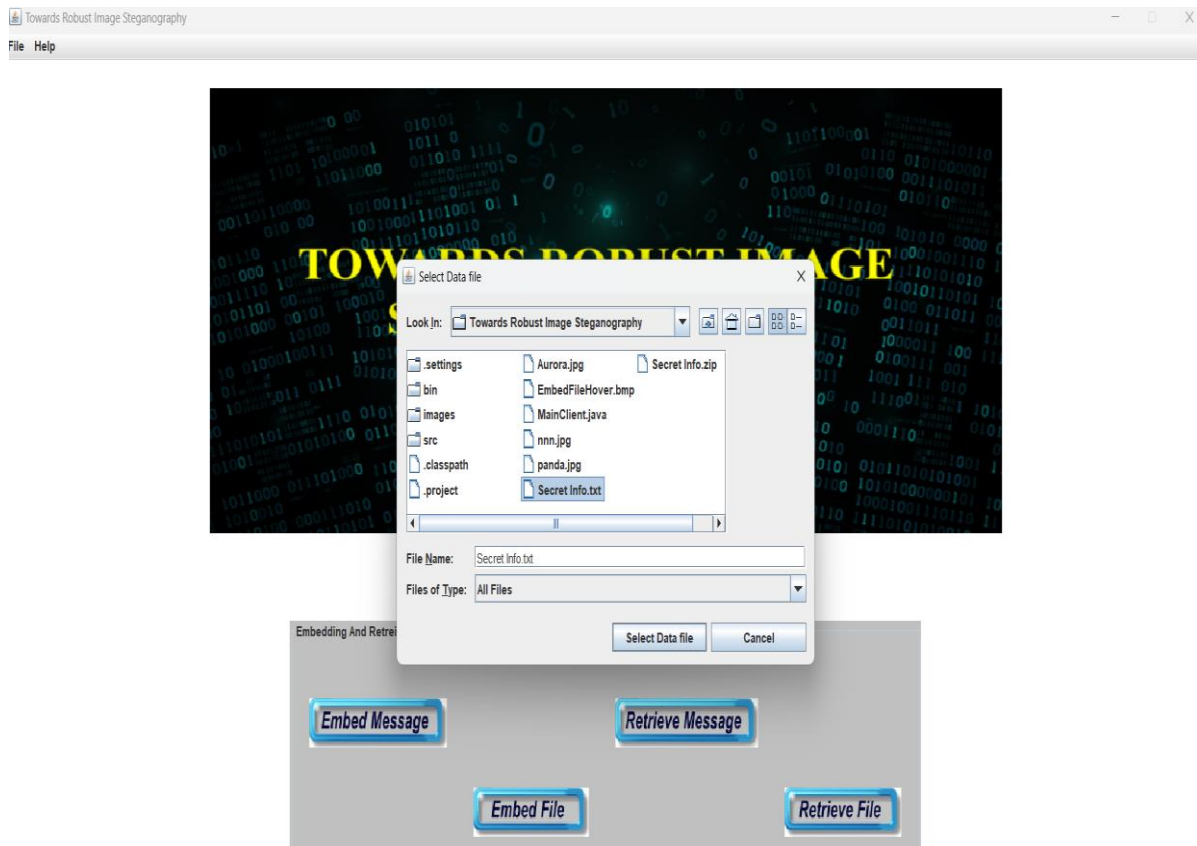


Fig 4.10: Selecting Secret File

We can select the secret file for embedding into the image. Here we are using the secret file “**Secret Info.txt**”. The secret file is around 656 Bytes. The fig 4.10 depicts the selection of secret file for embedding the data into image. Here at this instant of time we are using the image “**SunSet.jpg**”.



#### 4.1.10 EMBEDDING THE SECRET FILE

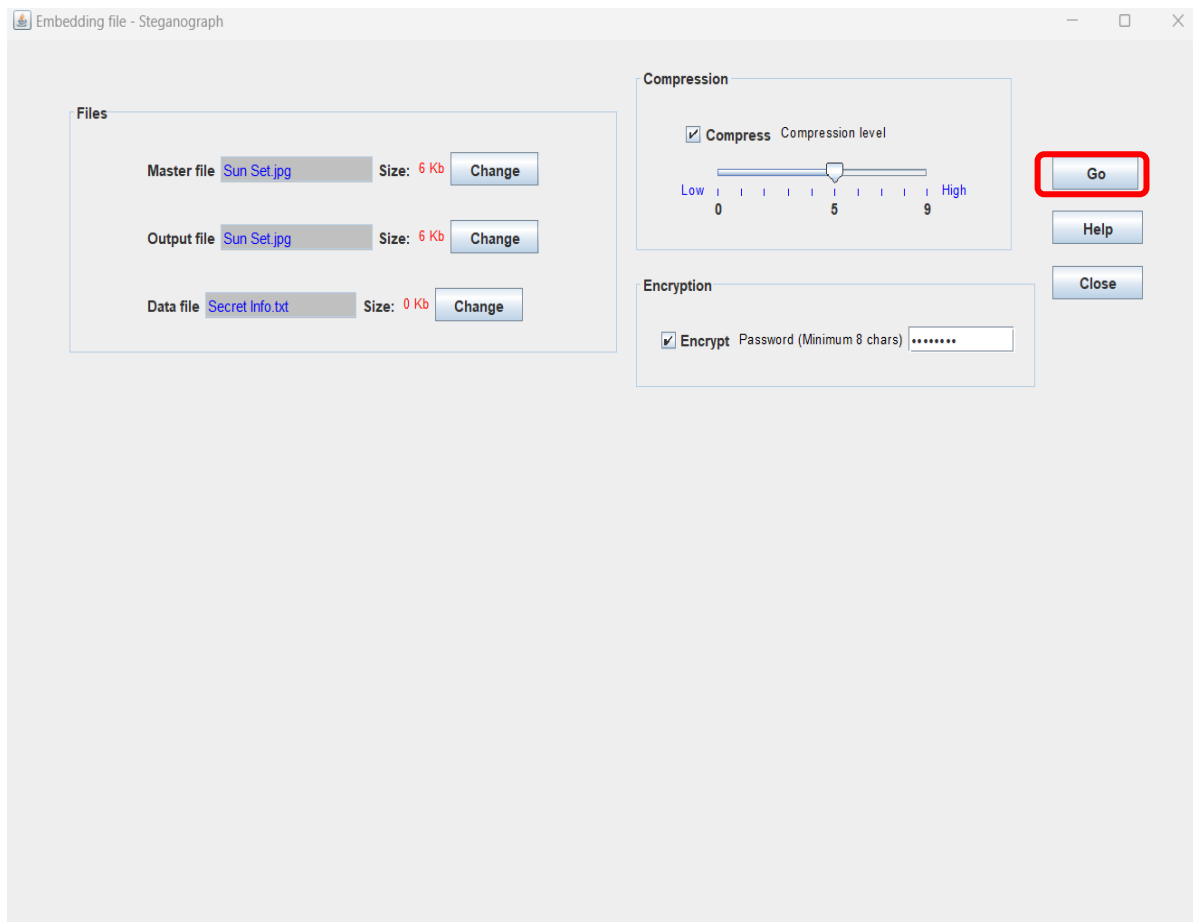


Fig 4.11: Embedding Secret File

Here the secret data file will get compressed. Also, we are using password protection. Below are original and stego images.



**Original Image ( 6336 Bytes )**



**Stego Image ( 6866 Bytes )**

#### 4.1.11 RETRIEVING THE SECRET MESSAGE FROM FILE

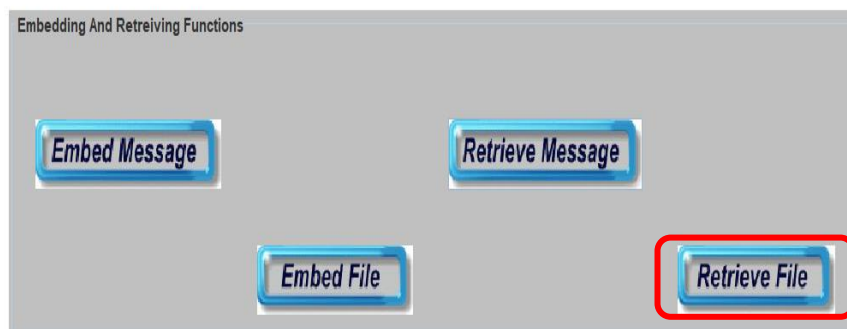
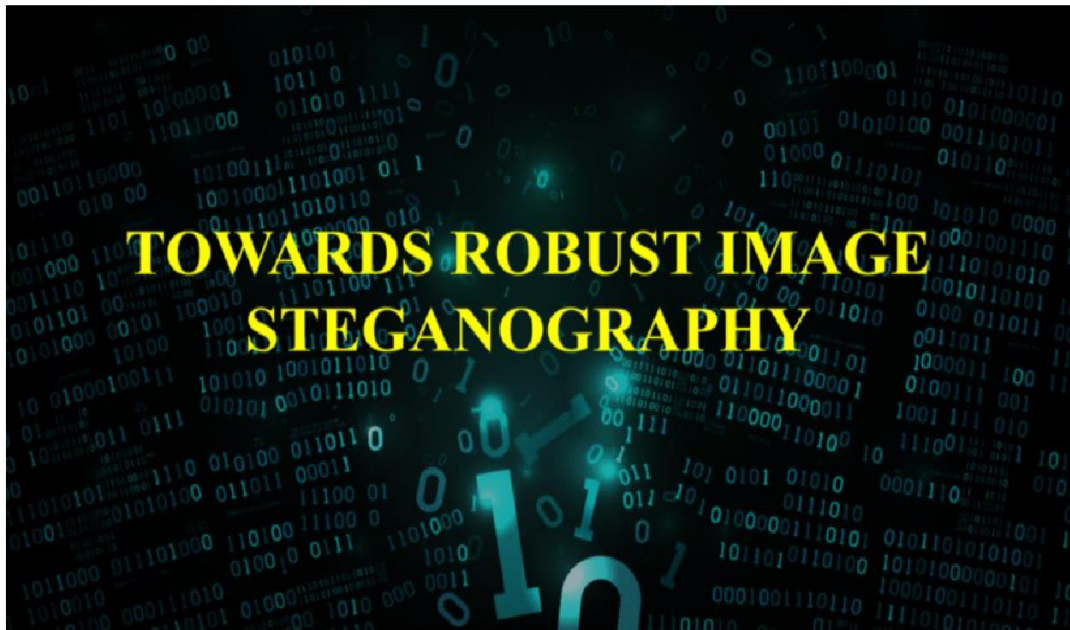
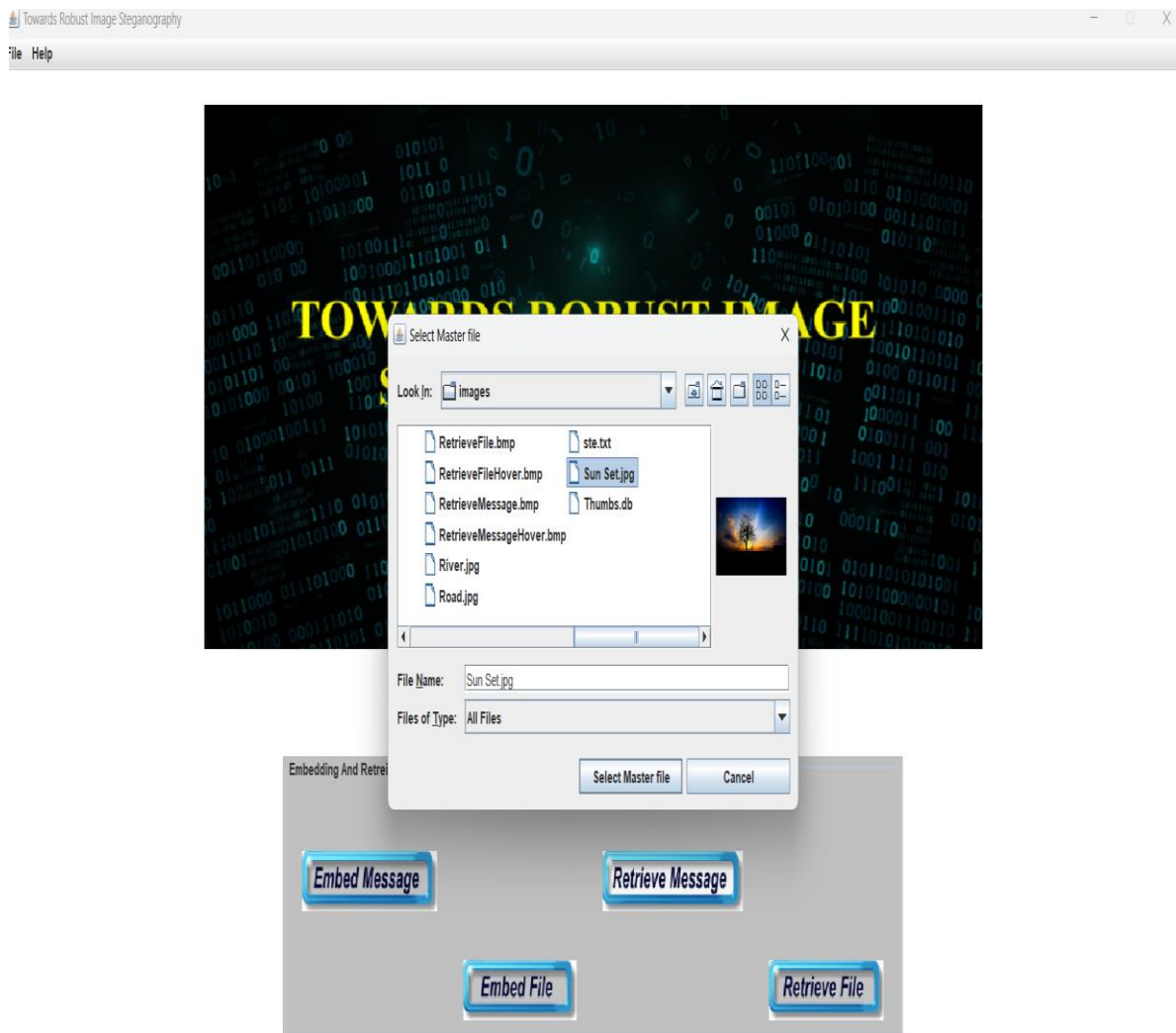


Fig 4.12: Retrieving the Secret Message

We successfully completed the embedding process and generated a stego-image that is carrying the secret information. Now we need to retrieve the embedded information from the stego-image. Fig 4.12 depicts the initiation process of the Retrieving the message.



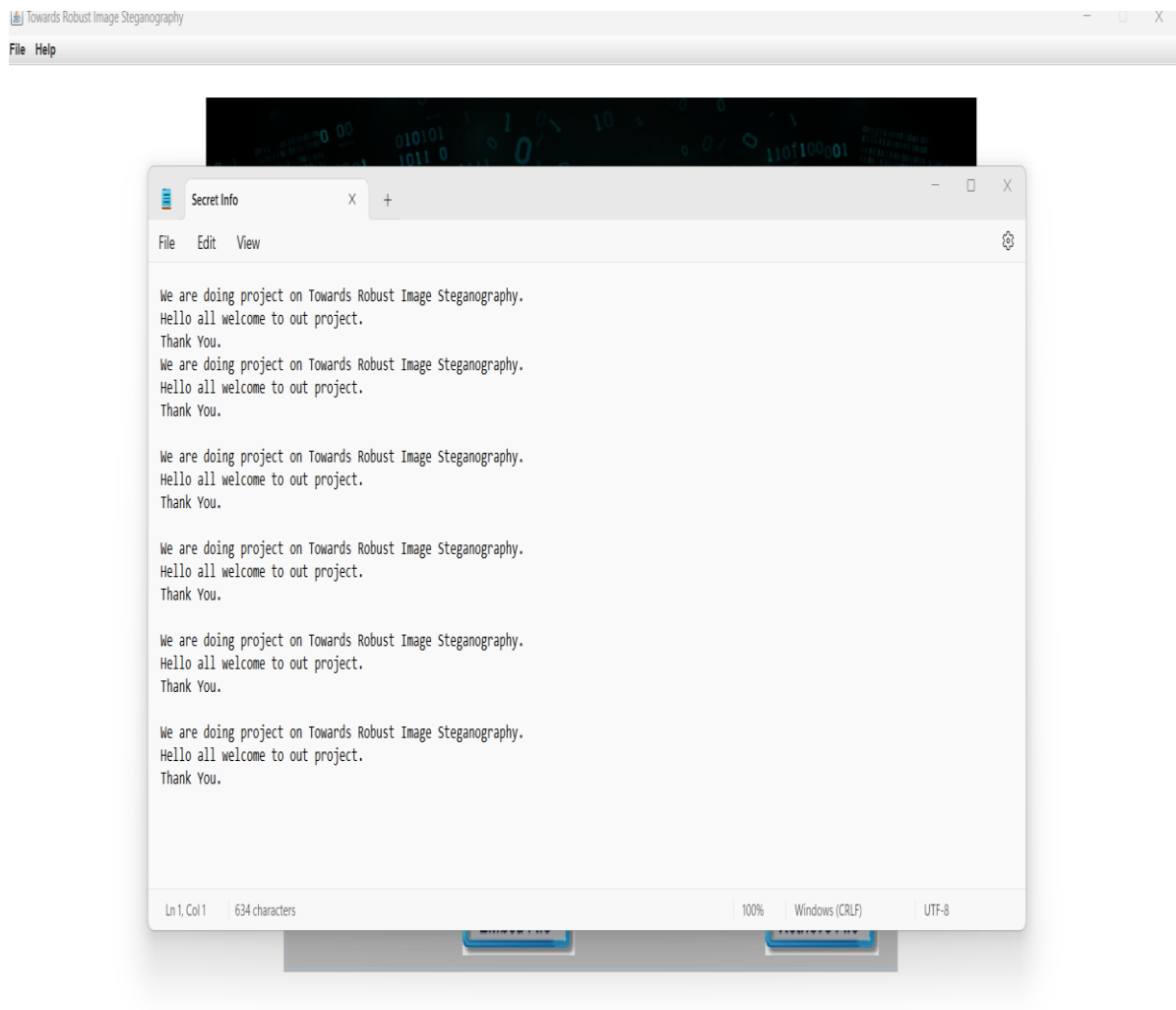
### 4.1.12 SELECTING THE STEGO IMAGE



**Fig 4.13: Selecting the Stego Image**

Here we are selecting the stego image in which we have embedded the secret information by giving a file as input. Fig 4.13 depicts the selection of the stego image. Here stego image is “**Sun Set .jpg**”.

### 4.1.13 RETRIEVED MESSAGE FROM STEGO IMAGE



**Fig 4.14: Retrieved Message**

Here we have successfully retrieved the secret message which we have embedded in the form of secret file. Finally, we retrieved the embedded message with 100% accuracy.

## **CHAPTER 5**

### **CONCLUSION**

A novel framework for robust image steganography is proposed in this paper, which is able to resist the Data compression of the communication channel. In this framework, we first obtain the stego-image by embedding data into the image using any of the existing steganographic schemes. According to the stego-image, we propose a coefficient adjustment scheme to slightly modify the original image to produce an intermediate image. We proof that we can always generate an intermediate image whose channel compressed version is exactly the same as the stego-image. Therefore, it is guaranteed that the secret data can be recovered from our stego-image with 100% accuracy. Meanwhile, it is able to achieve high non-detectability when an advanced steganographic scheme is adopted in our framework.

## REFERENCES

- [1] M. Asikuzzaman and M. R. Pickering, “An overview of digital video watermarking,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 9, pp. 2131–2153, 2018.
- [2] M. Asikuzzaman, M. J. Alam, A. J. Lambert, and M. R. Pickering, “Imperceptible and robust blind video watermarking using chrominance embedding: A set of approaches in the DT CWT domain,” *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 9, pp. 1502–1517, 2014.
- [3] M. Asikuzzaman, M. J. Alam, A. J. Lambert, and M. R. Pickering, “Robust DT CWT-based DIBR 3D video watermarking using chrominance embedding,” *IEEE Trans. Multimedia*, vol. 18, no. 9, pp. 1733–1748, 2016.
- [4] J. Fridrich, *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
- [5] T. Y. Liu and W. H. Tsai, “A new steganographic method for data hiding in microsoft word documents by a change tracking technique,” *IEEE Trans. Inf. Forensics Security*, vol. 2, no. 1, pp. 24–30, Mar. 2007.
- [6] B. Li, S. Tan, M. Wang, and J. Huang, “Investigation on cost assignment in spatial image steganography,” *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 8, pp. 1264–1277, Aug. 2014.

- [7] C. Qin, C.-C. Chang, Y.-H. Huang, and L.-T. Liao, “An inpainting assisted reversible steganographic scheme using a histogram shifting mechanism,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 7, pp. 1109–1118, Jul. 2013.
- [8] Y. Huang, C. Liu, S. Tang, and S. Bai, “Steganography integration into a low-bit rate speech codec,” *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 6, pp. 1865–1875, Dec. 2012.
- [9] D. Xu, R. Wang, and Y. Q. Shi, “Data hiding in encrypted H.264/AVC video streams by codeword substitution,” *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 4, pp. 596–606, Apr. 2014.
- [10] S. Li and X. Zhang, “Towards construction-based data hiding: From secrets to fingerprint images,” *IEEE Transactions on Image Processing*, doi:10.1109/TIP.2018.2878290.
- [11] A. Westfeld, “F5-a steganographic algorithm,” in *Proc. Int. Workshop Inf. hiding*. Springer, 2001, pp. 289–302.
- [12] P. Sallee, “Model-based steganography,” in *Proc. Int. Workshop Digital Watermarking*. Springer, 2003, pp. 154–167.
- [13] J. Fridrich and D. Soukal, “Matrix embedding for large payloads,” *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 3, pp. 390–395, 2006.
- [14] H. Wu and J. Huang, “Secure JPEG steganography by LSB+ matching and multi-band embedding,” in *Proc. IEEE Int. Conf. Image Process.*, 2011, pp. 2737–2740.

- [15] J. Fridrich and J. Kodovsk`y, “Rich models for steganalysis of digital images,” *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 868–882, 2012.
- [16] T. Filler, J. Judas, and J. Fridrich, “Minimizing additive distortion in steganography using syndrome-trellis codes,” *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 920–935, 2011.
- [17] V. Holub, J. Fridrich, and T. Denemark, “Universal distortion function for steganography in an arbitrary domain,” *EURASIP J. Inf. Security*, vol. 2014, no. 1, p. 1, 2014.
- [18] L. Guo, J. Ni, W. Su, C. Tang, and Y.-Q. Shi, “Using statistical image model for JPEG steganography: Uniform embedding revisited,” *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 12, pp. 2669–2680, Dec. 2015.
- [19] P. Bas, T. Filler, and T. Pevný, “‘Break our steganographic system’: The ins and outs of organizing BOSS,” *J. Amer. Statis. Assoc.*, vol. 96, no. 454, pp. 488–499, 2011.
- [20] G. Schaefer, “UCID: An uncompressed color image database,” *Proc. SPIE Electron. Imag. Storage Retr. Methods Appl. Multimedia*, vol. 5307, pp. 472–480, 2003.
- [21] W. Zhang, Z. Zhang, L. Zhang, H. Li, and N. Yu, “Decomposing joint distortion for adaptive steganography,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 10, pp. 2274–2280, Oct. 2017.

- [22] V. Holub and J. Fridrich, “Low-complexity features for JPEG steganalysis using undecimated DCT,” *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 2, pp. 219–228, Feb. 2015.
- [23] X. Song, F. Liu, C. Yang, X. Luo, and Y. Zhang, “Steganalysis of adaptive JPEG steganography using 2D Gabor filters,” in *Proc. ACM Workshop Inf. Hiding Multimedia Secure.*, 2015, pp. 15–23.
- [24] Y. Zhang, X. Luo, C. Yang, D. Ye, and F. Liu, “A framework of adaptive steganography resisting JPEG compression and detection,” *Secure. Communication. Network.*, vol. 9, no. 15, pp. 2957–2971, 2016.
- [25] Y. Zhang, X. Zhu, C. Qin, C. Yang, and X. Luo, “Dither modulation based adaptive steganography resisting JPEG compression and statistic detection,” *Multimedia Tools Appl.*, vol. 77, no. 14, pp. 17913–17935, 2017.