**CHABTER TWO**

* 1. **Techniques of steganography:**
     1. **Spatial Domain:**

In this technique only the least significant bits of the cover object is replaced without modifying the complete cover object. It is a simplest method for data hiding but it is very weak in resisting even simple attacks such as compression, transforms **[15].**

* + - 1. **Least Significant Bit (LSB):**

This is the most common, simple approach for embedding data in a cover image. The least significant bit (8th bit) of one or all of the bytes inside an image is changed to a bit of the secret message. When we use 24-bit image, three colour bits components are used which are red, green, blue, each byte store 3 bits in every pixel. An 800 × 600 pixel image, can thus store a total amount of 1,440,000 bits or 180,000 bytes of embedded data. For example a grid for 3 pixels of a 24-bit image can be as follows:

(00101101 00011100 11011100)

(10100110 11000100 00001100)

(11010010 10101101 01100011)

When the number 200, which binary representation is 11001000, is embedded into the least significant bits of this part of the image, the resulting grid is as follows:

(00101101 00011101 11011100)

(10100110 11000101 00001100)

(11010010 10101100 01100011)

The number was embedded into the first 8 bytes of the grid, only the 3 underlined bits needed to be changed according to the embedded message. On average, only

half of the bits in an image will modified to hide a secret message using the maximum cover size. Since there are 256 possible intensities of each primary colour, changing the LSB of a pixel results in small changes in the intensity of the colours. These changes cannot be visible by the human eye due to the message hidden. In these consecutive bytes of the image data – from the first byte to the end of the message – are used to embed the information. And easy to detect, more secure system for the sender and receiver to share a secret key that specifies only some pixels to be changed In its simplest form, LSB makes use of BMP images, since they use lossless compression. It hide a secret message inside a BMP file, one would require a very large cover image. In BMP images of 800 × 600 Pixels are not often used on the Internet and might arouse suspicion. For this reason, LSB steganography has also been developed for use with other image file formats **[16].** It is a simple method for embedding data in a cover image. This is the simplest algorithm in which information can be inserted into every bit of image information. Given an image with pixels, and each pixel being represented by an 8-bit sequence, the watermarks are embedded in the last (least significant bit) of selected pixels of the image proposed a simple data hiding technique by simple LSB substitution. In this technique last bit of host data is randomly changed and produce the watermarked data at output. The cover LSB media data are used to hide the message **[17].**

* + - 1. **Pixel Value Differencing:**

It provides both high embed-ding capacity and outstanding imperceptibility for the stego-image; this segments the cover image into non overlapping **[6]** blocks containing two connecting pixels and it modifies the pixel difference in each pair for data embedding.

* + - 1. **Pixel Indicator:**

This method gives the stego images of better quality than the traditional method while maintaining a high embedding capacity and it also uses concept of hiding the data using the difference between the pixel values **[18]**. It’s more complex way of hiding information in an image. Transformations are used on the image to hide information in. Transform domain embedding can be termed as a domain of embedding techniques in frequency domain; image is represented in terms of its frequencies.

**2.1.2. Frequency Domain:**

**2.1.2.1. Discrete Cosine Transformation:**

These methods convert the uncompressed image into JPEG compressed type**[19]**It is based on data hiding used in the JPEG compression algorithm to transform successive 8x8-pixel blocks of the image from spatial domain to 64 DCT coefficients each in frequency domain**. [15]**The main advantage of this method is its ability to minimize the block like appearance resulting when boundaries between the 8x8 sub-images become visible (known as blocking artefact).

**2.1.2.2. Discrete Wavelet Transformation:**

It gives the best result of image transformation .it splits the signal into set of basic functions .there are two types of wavelet transformation one is continuous and other is discrete **[16]** This is the new idea in the application of wavelets, in this the information is stored in the wavelet coefficients of an image, instead of changing bits of the actual pixels. It also performs local analysis and multi-resolution analysis. DWT transforms the object in wavelet domain and then processes the coefficients and performs inverse wavelet transform to show the original format of the stego object **[20].**

* 1. **Applications of Steganography:**

Applications of information hiding have been proposed in the context of multimedia applications. In many cases they can use techniques already developed for copyright marking directly; in others, they can use adapted schemes or shed interesting light on technical issues. They include the following **[21]:**

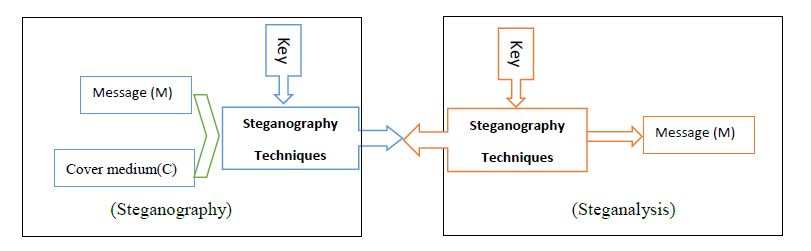
* Secret and Invisible Communication. It has requirements for security, invisibility and insertion of high volumes of secret data
* Copyright Protection.
* Authentication. It has security and invisibility requirements. Digital signature fits in this area.
  1. **The Different between Cryptography, Steganography and Watermark [22]:**

1. Cryptography which refers techniques for change data to become unreadable.
2. Steganography which hides data into a cover media so that it becomes invisible,
3. Watermarking is the process of embedding a message on a host signal Watermarking, as opposed to steganography, has the additional requirement of robustness against possible attacks. A watermark can be either visible or invisible.

2.4. **Components of Steganography system:**

1. The cover medium(C) can be Text, Image, Audio/Video or Network protocol.
2. The secret message (M), Text, Image, Audio/Video.
3. A stego-key (K) may be used to hide and unhide the message.
4. The steganographic techniques.

The Figure1 below show related between this components in steganography and in Steganalysis.



**Figure [2.1] Steganography and Steganalysis system**

**2.5. Steganography key:**

1. Pure steganography where there is no stego-key. It is based on the assumption that no other party is aware of the communication.
2. Secret key steganography it's as symmetric.
3. Public key steganography where a public key and a private key is used for secure.

**2.6. Requirements of steganography technique:**

There are many different protocols and embedding techniques that enable us to hide data in a given object. However, all of the protocols and techniques must satisfy a number of requirements so that steganography can be applied correctly **[23]** the following is a list of main requirements that steganography techniques must satisfy:

1. The integrity of the hidden information after it has been embedded inside the stego-object must be correct.
2. The stego-object must remain unchanged or almost unchanged to the naked eye.
3. In watermarking, changes in the stego-object must have no effect on the watermark.
4. Finally, we always assume that the attacker knows that there is hidden information inside the stego-object.

**2.7. Steganography in image:**

Image steganography has been widely studied by researchers. There are a variety of methods used in which information can be hidden in images. In the following section, we present the most common methods. There are three common methods of steganography: Replacing Moderate Significant Bit, Transformation Domain Techniques, and Replacing Least Significant Bit (LSB) in this project will focus on Replacing Least Significant Bit (LSB).

**2.8. Least Significant Bit Substitution:**

In LSB steganography, the least significant bits of the cover media’s digital data are used to conceal the message. LSB replacement steganography flips the last bit of each of the data values to reflect the message that needs to be hidden, in RGB image each pixel consist value between (0-256), 8-bit (00000000-11111111) The mechanism of this algorithm depend on last bit of pixel if the LSB of the cover pixel matches the bit of secret data no changes are done otherwise, one is added or subtracted from the cover pixel value, at random. Basic method of data hiding in an image is given as **[24]**:

Let C be the original 8-bit grayscale cover-image of Mc× Nc pixels represented as:

**C = {xij | 0≤ i <Mc, 0≤ j <Nc}**

**xij∊ {0, 1… 255}…. ………………………….(1)**

M be the n-bit secret message represented as

**M = {mi|0≤ i <N , mi ∊{0,1}}………………... (2)**

For embedding the n-bit secret message M into the k-rightmost LSBs of the cover-image C, the secret message M is rearranged to form a conceptually k-bit virtual image M’ represented as:

**M’ = {m’i |0≤ i <n’, m’i ∊ {0, 1,….. 2k− 1}}….(3)**

**Where n’<Mc×Nc.**

The mapping between the n-bit secret message M = {mi} and the embedded message

M’= {m’i}

Can be defined as follows:

∑ =  **i×k+j×2k-1-j**

After that, a subset of n’ pixels {xl1, xl2 ,………. xln} is chosen from the cover-image C in a predefined sequence. The embedding process is completed by replacing the k LSBs of xli by m’i . Mathematically, the pixel value xli of the chosen pixel for storing the k-bit message m’i is modified to form the stego-pixel x’li as follows:

**x’li = xli- xli mod 2k + m’i**

Algorithm for LSB [9] Based embedding and extracting process is given as-: A LSB-based Embedding Algorithm

**Input** -: cover C

**for** i = 1 to Length(c), doSjCj

**for** i = 1 to Length(m), do

Compute index ji where to store the ith message bit of m SjiLSB(Cji) = mi

**End for**

**Output -: Stego image S**

In the extraction process, given the stego-image S, the embedded messages can be directly extracted. Using the same sequence as in the embedding process, the set of pixels

{xl1, xl2 ,………. xln}storing the secret message bits are selected from the stego-image. The k LSBs of the selected pixels are extracted and lined up to reconstruct the secret message bits.

**A LSB-based Extracting Algorithm**

**Input** -: Secret image s

**For** i = 1 to Length (m), do

Compute index ji where to store the ith message bit of

m MjLSB(Cji)

End for

**Output -: message**

**2.9. Improved LSB method for hiding text in an image:**

In this method each character of message including special character such as space, enter etc. is converted in ASCII code then each value is converted in 8 bit binary number. Then, a subset of 8pixels {xl1, xl2 ,………. xl8} is chosen from the cover-image C in a predefined sequence. In this method RRGGGBBB sequence is chosen for embedding each bit of a character. The embedding process is completed by replacing the last LSB of cover image by a bit of secret image using key. This process is continued till all secret message bits get inserted in cover image. In the extraction process, given the stego-image S, the embedded messages can be directly extracted from stego image with the help of key, by using the same sequence as in the embedding process, the set of pixels{RGBBGRRG }, Mathematically, the embedded message bits m’ can be recovered by : **m’i= x’li mod 2k**

**Algorithm for embedding secret message-:**

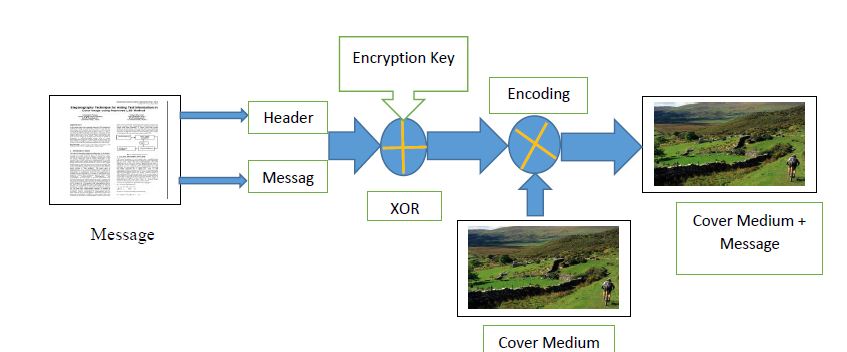
1. Steps to be carried out in this technique and implement in MATLAB are-:
2. Select a colour image as cover image
3. Now choose the text file containing secret information
4. Enter the key of 8 bit that is key must be lies between 0 – 255
5. Convert each character of secret message in ASCII code i.e. 97 for ‘a’, 48 for ‘0’, 32 for ‘space bar’ etc. In MATLB this can be achieve by “double” command.
6. Determine the length of the message and padded zero to make it 8 characters long and make it header of the message, which means to add message length at the header of message.
7. Convert each ASCII code to its 8 bit binary equivalent.
8. Take cover image and convert in unsigned 8 bit integer so as to bring its pixels value in range [0 255]
9. Separate cover image in RGB plane and insert message bits last bit of Red, Green, Blue, Blue ,Green ,Red, Red ,Green respectively So 8 bits of first character of secret text message get embedded in 8 pixels of cover image Each time when a bit is embedded to a pixel of a plane increase its position by 1 so as to go on next pixel of that plane. This process continues till all bits get embedded in cover image.
10. Resulting image is stego image containing information.

**2.10. Algorithm for extracting secret message:-**

1. Steps to be carried out in this technique for extraction of the secret message are-:
2. Select stego image, and enter the key.
3. Separate stego image in RGB plane take mode 2 of first and 2nd pixel of red plane so as to get first 2 bit of first character of secret message.
4. Now take mode of last bit of 8-pixel as sequence colour (Red, Green, Blue, Blue, Green, Red, Red, Green) to get bits of first character of secret message respectively. Each time mode 2 of pixel is taken increase its position value by 1 so as to go to next pixel. Run this step up to 8 times so as to get header which contain the information of secret message length.
5. Now repeat the same process again utile length of message.
6. Resulting message containing information was hide.

**2.11. Steganographic Encoding Process:**

Encoding hidden messages using steganography follows the basic process outline in Figure [2.2].

****

**Figure [2.2]: Stenographic Encoding Process. The message and header are XOR encrypted and then encoded into the cover medium image.**

The message is first analysed to determine whether it is a text or image message type. The message type and dimensions of the message (overall length for text and height/width values for an image) form an 8-bit Header that is used to reconstruct the message during the Decoding Process. The Header is concatenated to the beginning of the message and this new combined message is encrypted using a simple symmetric Exclusive OR (XOR) encryption key, which follows the bit logic outlined in Table and Example 1.

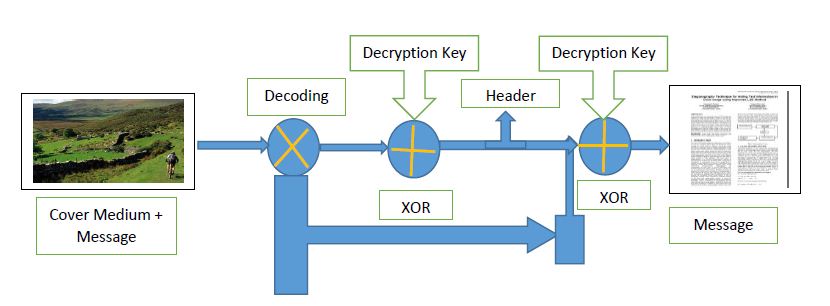
|  |  |  |
| --- | --- | --- |
| **Message value** | **Encryption Key value** | **Encrypted value** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

|  |  |  |  |
| --- | --- | --- | --- |
| Encoding |  | Decoding |  |
| Plaintext: | 10101010 | Cipher text: | 10100101 |
| Encryption Key: 00001111 | | Encryption Key: | 00001111 |
| Cipher text: 10100101 | | Recovered Plaintext: 10101010 | |
|  | |  | |

From Example 1 we can clearly see that we need to use the same encryption key during the decryption process if we want to successfully recover the original message. Unfortunately, this symmetric encryption key is a shared secret that must be sent separately from the encrypted message to ensure it remains uncompromised and secure.

Finally, the encrypted Header and Message is encoded onto the Cover Medium Image’s least significant bits. The Sequential Encoding method which we will describe in more detail in the Method Section below:

**2.11.1. Stenographic Decoding Process:**

 Decoding and recovering the hidden message follows the basic process outlined in Figure [2.3]. This process reverses the effects of the encoding process and reveals the secret message to an authorized user.

**Figure [2.3]: Steganographic Decoding Process. First, the Header is decoded, XOR decrypted, and analyzed to determine the dimensions and message type. Next, the Message is decoded and XOR decrypted before finally being reconstituted using the dimension data from the Header**

First the encrypted Header values are recovered from the Cover Medium Image and decrypted using the same XOR encryption key used during the encoding process. The Header dimension value (length for text message; width times height for image message) is used to determine the stop value for the recovery algorithm, thereby reducing the complexity and speeding up the recovery process.

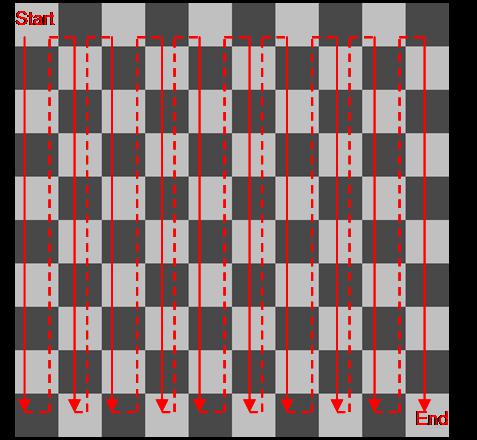
Next, the program recovers the entire encrypted Message from the Cover Medium Image by using the Header Dimensions to determine when to stop. The recovered Message is decrypted using the same XOR encryption key used during encoding. Finally, if the message type is an image, the program uses the height/width dimensions from the recovered Header to reconstruct the original image from the decrypted message values for display.

**2.12. Method:**

For our project we implemented Sequential and Decoder functions using MATLAB script files. Our functions follow the code outline provided in [**25**], but also include message transposition and XOR encryption for enhanced security. We also developed a rudimentary user-input function that allows a user to encode and decode their own hidden messages without preloading variables in MATLAB.

**2.12.1. Sequential encoding (stegancoder Function) and decoding (stegandecoder Function):**

Sequential steganography is relatively simple to implement and provides the most basic level message hiding capabilities. Sequential, like the name implies, begins at some point and then follows some set pattern to encode the message across the least significant bits of the Cover Medium Image. The stegancoder function begins with the top left pixel of the Cover Medium Image and proceeds to encode the message down the columns of the image from left to right, like is shown in Figure [2.4].



**Figure [2.4]: Sequential Encoding Method**

After the message has been analyzed, the Header added, and the new message encrypted, each value of the message is converted from a value in the range of 0-255 and is represented in its 8-bit binary equivalent, which we will call a message word. For example, the message word for the number 15 would be represented as 00001111.

Next, to enhance message security we implement a layer of scrambling, called transposition, during the encoding process. Each bit of every message word is stored in the following pattern: RGBBGRRG; where R, G, and B mean the message value is stored in the Red, Green, and Blue Channels of the next available pixel. This means that every message word uses three pixel’s worth of Red and Green Channel values while only require two pixel’s worth of Blue Channel values. To implement this pattern the stegancoder and stegandecoder

For improved Message security our program requires the usage of an Encryption Key Value that is between 0-255.Finally, the steganography function outputs the resulting Cover Medium Image with encoded Message or decoded Message directly to the

variable workspace as well as prompts the user for a filename to save a copy of the results in the appropriate file format. Images are saved in the lossless bitmap (.bmp) format and text messages are saved into text file (.txt) format.

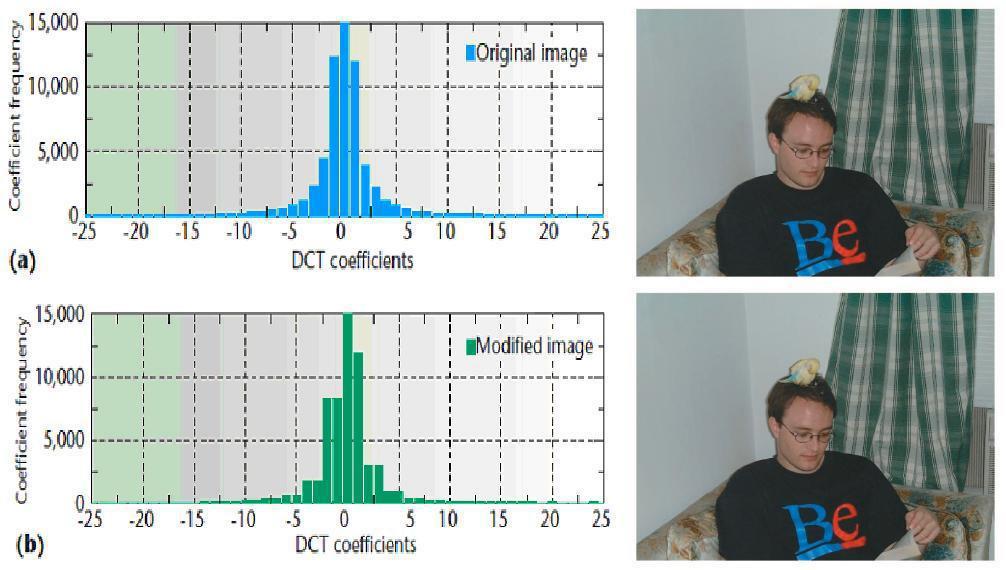
**2.13. Attacks on steganography:**

Steganography algorithms provide stealth and security to information. The degree of stealth and security is hard to measure. One way to judge the strength of a steganographic algorithm is to imagine different attacks and then assess whether the algorithm can successfully withstand them. Attacking staganographic algorithms is very similar to tracking cryptographic algorithms**.[25]**

Here’s a list of some possible attacks:

* File Only: The attacker has access to the file and must determine if there is a message hidden inside.
* File and original Copy: If the attacker have a copy of the file with the encoded message and a copy of the original, pre-encoded file, then detecting the presence of some hidden message is a trivial operation. The real question is what the attacker may try to do with the data (destroy hidden information, extract the information, replace...).
* Multiple Encoded Files: The attacker gets n different copies of the files with n different messages. This situation may occur if a company is inserting different tracking information into each file. Some attackers may try to replace the tracking information with their own version of the information.
* Compression Attack: One of the simplest attacks is to compress the file. Compression algorithms try to remove the extraneous information from a file, and “hidden” is often equivalent to “extraneous”.
* Destroy Everything Attack: An attacker could simply destroy the message.
* Random Tweaking Attacks: An attacker could simply add small, random tweaks to all files in the hope of destroying whatever message may be there.
* Reformat Attack: One possible attack is to change the format of the file. Different file formats don’t store data in exactly same way (BMP, GIF, and JPEG).

**2.14. Steganalysis - Detecting Hidden Messages:**

 Even though the human visual system is unable to differentiate between subtle colour differences or changes, steganographic messages are still detectable. Steganographic messages are typically encoded by altering the least significant bit of a pixel colour value in a specific order or pattern, leaving them vulnerable to statistical analysis tools which can be used to detect and provide information about messages hidden within cover media. For messages Sequentially Encoded, where the message starts at the top left corner of an image and proceeds to the next pixel until the message is completely encoded, histogram analysis can be completed to identify the presence and length of the message hidden in an image**.** In Figure [2.5], the first chapter of Lewis Carroll’s The Hunting of the Snarky was Sequentially Encoded into the image in (b) leaving no visual indication that anything has been changed, but when we examine the Discrete Cosine Transform (DCT) Histogram of the images we can clearly see that the message has altered the distribution of the DCT coefficients from a normal distribution **[24].**

**Figure [2.5]: Histogram Analysis of Sequentially Encoded Message [25]**

* 1. DCT Coefficients Histogram (left) and Image (right) without message.

1. DCT Coefficients Histogram (left) and Image (right) with Sequentially Encoded message.

As the message size increases it uses a higher percentage of the DCT Coefficients and is easier to detect.

Unfortunately, we were unable to get the Outguess program, used in the by the authors of **[25]**, to work with any of our image results. Consequently, we are unable to draw any conclusions about the effectiveness of our functions compared to the commercially available products analysed by the authors of **[24].**