ALGORITHM IMPLEMENTATION

5.1 Design Methodologies

5.1.1 Types of Algorithms in Image steganography

1) Spatial Domain Algorithm.
2) Transform Domain Algorithm.
3) Distortion Technique.
4) Masking And Filtering.

1) Spatial Domain Algorithm.

Spatial domain steganography remains a popular method, whereby images are represented as a matrix of intensity values. It is the individual values of the spatial domain that are commonly modified. This is because small changes to intensity values generally cause hardly recognizable visual differences to the stego-object as a whole. The early basis for this type of steganography was to exploit the limitations of the human visual system (HVS).

Methods in Spatial Domain Algorithm.

i) LSB Substitution Method.

LSB (Least Significant Bit) substitution is considered the most well-known method of embedding in digital steganography. It relies on the principle that the least significant bit of any byte is frequently indistinguishable from random noise and can therefore be replaced by bits from a secret message. For example, Below Figure shows the full representation of a greyscale image (on the left) paired with its representation in the LSB bit-plane. The LSB bit-plane appears random and should therefore accommodate bit replacement with an encrypted message. Advancements in steganalysis, however, revealed this naive assumption to be completely untrue, leading to steganography methods exploiting this false concept.

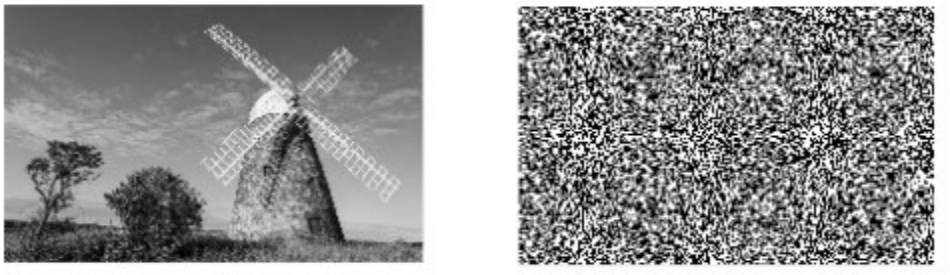

Fig: Comparison of gray scale and LSB bit plane.

The early work on image steganography was by Bender et al., Adelson, and those working in spread-spectrum steganography. Initially, these researchers based their schemes on exploiting the limitations of the Human Visual System (HVS) and its sensitivity to contrast versus spatial frequency. In the naive applications of LSB steganography, there was one qualifying characteristic, that the change of the LSB of a byte would result in a stego-object being visually indistinguishable from the cover-object. The concept of LSB is still widely applied in modern steganography schemes. However, these methods are typically supported by complex operations and cover codes to produce a secure scheme with improved embedding efficiency, that is significantly harder to detect. One of the first known implementations for digital steganography was a spatial domain technique proposed by Kurak and Mchugh. This method embedded secret data within 4-LSB components and was the first case of steganography exploiting detailed knowledge of the HVS. Typically, spatial techniques modify bits within an image's pixel values, replacing them directly with the secret message bits. The concept of bit replacement would be that, for each pixel in an image, it can be assigned a corresponding integer value, based on the intensity of a colour that is displayed by the pixel. Once converted into a binary stream, individual bits can be extrapolated to determine whether they are suitable for substitution. For example, functions can be included to prevent even values from being decreased and odd values from being increased. Below Figure demonstrates the process for LSB bit substitution steganography.

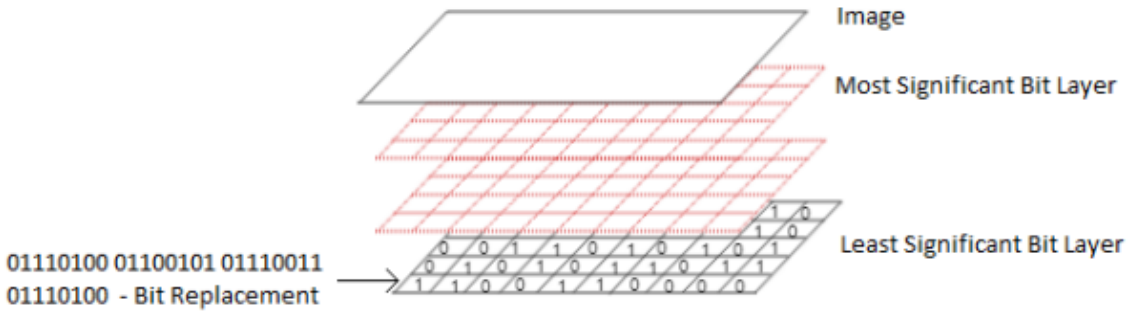

Fig: Break down of bit-plane of an image.

This concept formed the initial works in LSB steganography. However, naïve LSB schemes are highly susceptible to Chi-square tests and statistical steganalysis. LSB encoding within 24-bit colour images was the most popular form of early spatial domain steganography. The embedding technique was used across many software products. Capacity became the focus of these

techniques, as 24- bit images allowed the user to store up to 3-bits for each modified pixel. The early work in spatial domain steganography for colour based images was broken by the efforts of Fridrich et al. . To detect LSB steganography, Fridrich and her team considered a calculation of the ratio between close colour pairs and all colour pairs in an image. Their idea was that for an image with no secret messages, the number of close pairs of colours relative to the number of all possible pairs of colours will be smaller than for an image that has a message already embedded in it. To determine the ratio threshold, their team observed that further modifying a stego-object with a new secret message will not modify the ratio in any significant manner. Whereas if the image does not contain a secret message, then embedding will drastically change the ratio. This knowledge can be used to determine the presence of LSB colour encoded steganography and develop a detection algorithm. A test was used where a message was randomly distributed throughout the LSB of selected pixels. The calculation is given to the new stego-object and if the two ratios are almost identical, LSB steganography is suspected.

ii)LSB Matching Method.

LSB matching is a method of LSB steganography first proposed by Sharp. The concept was that instead of sequentially and predictably substituting bits of target LSBs within an image, an adaptive scheme would be constructed to be selective of the features present in any particular cover-object. If the message bit fails to match the LSB of the cover-image, then a '1' value is either added or subtracted from the value of the target pixel. To ensure that the scheme for AoSo (Add or Subtract One) would be invertible, an algorithm would be implemented to stop pixel values from being modified outside of a predetermined range. While simple to implement, it is surprisingly and significantly more challenging to detect. than basic LSB substitution. A similar method was proposed by Mielikainen who proposed that instead of using a random function to determine the use of the +1 or −1 principle, a selection feature can be introduced to set a binary function of two cover pixels to the desired value. This improved method of LSB matching maintained the same embedding efficiency as the original but provided greater resistance to known attacks, such as the one proposed by Westfeld. Westfeld's attack relied upon a count of arbitrary neighbor colours introduced by the embedding algorithm for LSB matching. However, this means that the detection method was only applicable to colour images and would not work for a grayscale representation. A stronger method for greyscale images was presented by Andrew Ker.

iii)Pixel Value Difference.

The pixel-value differencing (PVD) scheme provides high imperceptibility to the stego image by selecting two consecutive pixels and designs a quantization range table to determine the payload by the difference value between the consecutive pixels. Besides, it offers the advantage of conveying a large number of payloads, while still maintaining the consistency of an image characteristic after data embedding.

In recent years, several studies have been proposed to improve the PVD method. Wu et al.'s presented a method combining pixel-value differencing and the LSB replacement method. Yang and Weng proposed a multipixel differencing method that uses three difference values in a four-pixel block to determine how many secret bits should be embedded, and Jung et al.'s proposed an image data hiding method based on multipixel differencing and LSB substitution. Liu and Shih proposed two extensions of the PVD method, the block-based approach and Haar-based approach, and Yang et al. proposed an information hiding technique based on blocked PVD. Liao et al.'s proposed a four-pixel differencing and modified LSB substitution, and Yang et al.'s proposed a data hiding scheme using the pixel-value differencing in multimedia images.

Some studies focused on increasing the capacity using LSB or a readjusted process to improve the embedding capacity or image quantity. Few studies focus on the range table design. Besides, it is intuitive to design it using the width of the power of two.

In this work, we design a new quantization range table based on the perfect square number to decide the payload by the difference value between the consecutive pixels. It differs from the design of Wu and Tsai's scheme, in which the quantization range table is based on the range width of the power of two. The perfect square number provides an elegant mathematical model to develop a new quantization range table, which divides each range into two subranges for embedding different numbers of secret bits.

2) Transform Domain Algorithm.

This is a more complex way of hiding information in an image. Various algorithms and transformations are used on the image to hide information in it. Transform domain embedding can be termed as a domain of embedding techniques for which a number of algorithms have been suggested. The process of embedding data in the frequency domain of a signal is much stronger than embedding principles that operate in the time domain. Most of the strong steganographic systems today operate within the transform domain Transform domain techniques have an advantage over spatial domain techniques as they hide information in areas of the image that are less exposed to compression, cropping, and image processing. Some transform domain techniques do not seem dependent on the image format and they may outrun lossless and lossy format conversions. Transform domain techniques are broadly classified into:

1. Discrete Fourier transformation technique (DFT).
2. Discrete cosine transformation technique (DCT).
3. Discrete Wavelet transformation technique (DWT).
4. Lossless or reversible method (DCT)
5. Embedding in coefficient bits.

3) Distortion Technique.

Distortion techniques need knowledge of the original cover image during the decoding process where the decoder functions to check for differences between the original cover image and the distorted cover image in order to restore the secret message. The encoder adds a sequence of changes to the cover image. So, information is described as being stored by signal distortion. Using this technique, a stego object is created by applying a sequence of modifications to the cover image. This sequence of modifications is use to match the secret message required to transmit. The message is encoded at pseudo-randomly chosen pixels. If the stego-image is different from the cover image at the given message pixel, the message bit is a "1." otherwise, the message bit is a "0." The encoder can modify the "1" value pixels in such a manner that the statistical properties of the image are not affected. However, the need for sending the cover image limits the benefits of this technique. In any steganographic technique, the cover image should never be used more than once. If an attacker tampers with the stego-image by cropping, scaling or rotating, the receiver can easily detect it. In some cases, if the message is encoded with error correcting information, the change can even be reversed and the original message can be recovered.

4) Masking And Filtering.

These techniques hide information by marking an image, in the same way as to paper watermarks. These techniques embed the information in the more significant areas than just hiding it into the noise level. The hidden message is more integral to the cover image. Watermarking techniques can be applied without the fear of image destruction due to lossy compression as they are more integrated into the image.

Advantages of Masking and Filtering Technique:

1. This method is much more robust than LSB replacement with respect to compression since the information is hidden in the visible parts of the image.

Disadvantages of Masking and Filtering Technique:

1. Techniques can be applied only to gray scale images and restricted to 24 bits.

5.2 Encoding Algorithm:

Step 1: START

Step 2: Get the image and message as the parameters for Encoding function.

Step 3: Calculate the maximum bytes of the image.

Step 4: IF length of Secret image > number of bytes of image:

Raise “value error”

Step 5: Set the Delimeter as the user key.

Step 6: Convert the secret message into binary format using the MessageToBinary function in program.

Step 7: Find the length of the Binary secret message.

Step 8: Initialize Data_Index=0

Step 9: Start FOR LOOP values in image:

Start FOR LOOP pixel in values:

#Convert the RGB values into binary format.

IF Data_index < length of binary secret message:

Modify the Least Significant bit of RGB colours to store our data.

IF data_index > length of binary secret message:

Break

Step 10: END of FOR LOOP pixel in values.

Step 11: END of FOR LOOP values image.

Step 12: Return Image.

Step 13: END

5.3 Decoding Algorithm.

Step 1: START

Step 2: Get the Image as a parameter for decoding algorithm.

Step 3: Start FOR LOOP values in image:

Start FOR LOOP pixels in values:

#Convert RGB values into binary format.

#Extract the data of the least significant for each pixel.

Step 4: End FOR LOOP pixels in values.

Step 5: End FOR LOOP values in image.

Step 6: Divide the extracted data into 8-bits each.

Step 7: Start FOR LOOP Bytes in 8-bit data

#Convert each byte into characters.

IF Decoded data includes delimeter:

Return Decoded data

Step 8: End of FOR LOOP bytes in 8-bit data

Step 9: ENDs