International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 03 | Mar 2022 www.irjet.net p-ISSN: 2395-0072

Message Camouflage in an Image using EDGE-Based Steganography

Akkimsetti Mohana Sai Chandra

Student, Dept of Electronics and Communication Engineering, School of SEEE, SASTRA University, Thanjavur, Tamil Nadu, INDIA

Abstract - *In Modern day world, Data Security is considered a major concern. Data can be transmitted in numerous ways over the internet. So, the data needs to be secured between the sender and receiver. Steganography is one technique, that hides information inside other media in such a way that no one will notice. Generally, in Steganography Images are considered as Cover media because they have high redundancy in representation and most common used over the internet. Images are popular for cover media as they have small size, content adaptability, visual resilience make them good carrier to transmit secret messages over the internet. In this paper, the data is hidden in an image using Edge based steganography where the data is hidden in the edge pixels of the image. Canny edge algorithm is applied for edge detection. This method considers the amount of data to be embedded as an important factor on the selection of edges, i.e., more the amount of data to be embedded, larger the use of weak edges in the image for embedding*

Key Words: **Edge Steganography, Canny Edge detection, Encoding, Decoding, Edge pixel, MATLAB, least significant bit (LSB), XOR, Modulo Operation**

1.INTRODUCTION

Encryption is a process that encodes information or message or file so that it can be only be accessed by authorized people. Encryption uses algorithms to encrypt data at the transmitter using a key and then the receiver uses same or different key (based on type of encryption used) to decrypt the information. In an encryption technique, the secret information or message, referred to as plaintext, is encrypted using an encryption algorithm – a cipher– generating ciphertext that can be read only if decrypted. On the other hand, Steganography is the practice of hiding a message within another message or a cover object [1]. In computing/electronic contexts, a computer file, message, image, or video is hidden within another file, message, image, or video. Generally, In Steganography grayscale images are used. A grayscale image is one in which the RGB space colors are shades of grey. The reason for opting grayscale scale image is that less information needs to be provided for each pixel. In Gray images have red, green and blue components have all equal intensity in RGB space, so only a single intensity value is required rather than three intensity values.

Security of the steganography algorithm depends on the pixel selection. Noisy pixels are considered ideal as modelling them is difficult. Edge pixels are noisy because the intensity suddenly changes at edge. Edge Pixel is a pixel where the intensity changes suddenly in an image [2]. These sharp variations make the edge pixel difficult to model. Therefore, edge pixels are better choice for embedding the data.

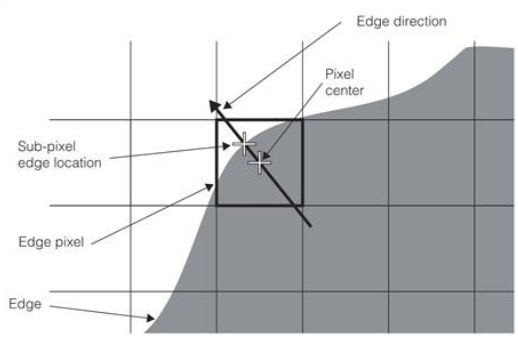

Fig 1. Edge Pixel in an image

1.1 Literature Review

As the technology increases the information exchange became insecure. A breach during this information exchange can cause a disaster (when it comes to exchange between military data or confidential data etc.). So, there is a need for protecting the information from third parties. The tool which helps to protect stored information and information transfer is Information Security [3]. Information security is crucial in safeguarding, securing and maintain the integrity of information. Steganography is an art of secure transmission of information from transmitter to receiver by ensuring that no other one can reliably conclude on the secret communication.

Information or secret message can be hidden into cover medium or cover data through several steganographic techniques. Steganographic techniques involves embedding and extracting of information. Image-based steganographic techniques can be classified as spatial domain and frequency (transform) domain [4].

Images are preferred cover medium for steganographic techniques because of their content adaptability, resilience, high redundancy and smaller size of images make them good carrier to transmit secret messages [5]. A secret message is generally embedded as, the bits of encrypted message are embedded in pixels of the cover image. LSB Steganography is the most used and renowned

Steganographic technique. LSB steganography is technique in which the least significant bit of the pixels is used for embedding the secret information. LSB steganography can be further classified as LSB replacement and LSB matching [7]. In LSB replacement the least significant bit of each pixel of cover image is replaced by next bit of secret message, whereas in LSB matching the pixel value of cover image is randomly increased or decreased by 1 except at boundary pixels if any mismatch occurs between least significant bit and secret message [8]. Edge adaptive image steganography is another technique based on LSB matching which calculates the difference between two adjacent pixels. If the difference is greater than the predefines threshold then both pixels are marked as edge pixels, then one bit of data is hidden in each of them. But the major limitation of this technique is if the image is smooth then detection of edge pixels will be difficult.

As mentioned earlier steganography can be classified into spatial and transform domain, Adaptive Steganography is a special case of both spatial and transform domain techniques. In the spatial domain the secret data is hidden in pixels of cover image using LSB steganography [6]. This technique is widely used due to its potentiality of embedding secret information in an image with high capacity. In transform domain (frequency domain), the secret data is hidden in the transformed coefficients of the cover image using Discrete Cosine Transform [9]. Edge detection is a technique where the edges in an image are found out by using a suitable algorithm. The edge which can be defines as a set of pixels positions where a sudden or abrupt change of intensity values occurs. Edges represent boundaries between objects and background. Edge detection involves method of segmenting an image into regions of discontinuity. Some edge detection techniques are Sobel, Prewitt, LoG, Robert, Canny etc. [11].

1.2 Classic Edge Detection Techniques

Sobel's algorithm: When there is a change in the intensity value of the image pixel, then the edge can detect. There are several techniques to detect an edge. Sobel is one of the former edge detection techniques, which uses the gradient of the image intensity to detect edge. In this algorithm or technique, the image intensity gradient is taken at each and every point in the image. This gives the magnitude and direction of intensity of the image. Then by comparing the resultant threshold values, the presence of an edge can be determined. Sobel operator can be used to find the gradient of the image intensity values [10]. Basically, Sobel operator is a discrete differentiation operator. It is very simple and time efficient computation. Smooth edges can be detected very easily. But it is very sensitive to noise and not very accurate in edge detection as it needs comparison with a threshold. Thick and rough edges do not give appropriate results.

Prewitt Algorithm: Prewitt algorithm is similar to the Sobel algorithm, but the difference is the source image point is convolved with two 3x3 kernels vectors to obtain the horizontal and vertical derivatives. By using the "EDGE" function in MATLAB and the Prewitt algorithm results in a binary image where bit 1 corresponds to edges and bit 0 corresponds to non-edges. Prewitt operator is the best operator to detect the orientation of an image, but diagonal direction points can't be determined.

Roberts Edge Detection Algorithm: This technique also detects the edges by computing intensity gradient just like Sobel and Prewitt. Besides, this algorithm simplifies the complexity as it uses a simple 2×2 kernel vector for gradient computation. In addition to this, every pixel gradient intensity is compared only with that of those diagonally adjacent pixels. So, its main disadvantage is that since it uses such a small kernel, it is very sensitive to noise. Diagonal direction can be preserved by this technique

Laplacian of Gaussian Algorithm (log) and Zero Crossing Edge detection Algorithm: Laplacian of Gaussian Algorithm (log) is different from the Sobel, Prewitt as it does not use the gradient of intensity values (like the other edge detection techniques), instead it uses zero crossing technique. Initially, Gaussian function is employed to minimize the effect of noise and smoothness. Then the Laplacian of the resultant is calculated. If any sign change occurs in the Laplacian, we can conclude that an edge is detected, in other words when the Laplacian crosses zero then an edge is detected and the resultant binary image has a high value - '1' in that point to indicate the edge. If the intensity gradient starts increasing or decreasing then the zero crossing reports an edge, and this may happen at places which are not true edges. So, the log method uses other alternative methos for edge detection. One such method is utilizing threshold of the log value so that a value higher than the threshold would report an edge. There is a chance of multiple edges are being detected. Another method by comparing the pixel log value with that of its adjacent pixels and choosing those points that are having lesser log magnitude than that of its four neighboring points. But there is a risk of missing few edges in this technique.

Canny Edge Detection Algorithm: The canny algorithm is one of the most famous and most used edge detection techniques as it is less prone to the noise. In this algorithm, convolution of original image with Gaussian filter eliminates the effect of noise [12]. The resultant image is then utilized to calculate the intensity gradient and the direction of the gradient based on the gradient angle. If the strength of the intensity at that point is higher than that of its adjacent points along the direction of the gradient then the edge can be pointed out on the image. This algorithm employs two threshold values-the higher value to include the true genuine edges and the lower value to trace the very small details of

the image. Hence, the canny edge detection algorithm gives a more accurate, elaborate and detailed binary image.

Algorithm	Total Pixels	Edge Pixels (Average)
Canny	262,144	23,818
Sobel	262,144	8,451
Prewitt	262,144	8,407
LOG	262,144	18,220

Table1. Comparison of detected pixels values of different algorithms

Fig 2.1 RGB Image

Fig 2.2 Grayscale Image

2. METHODOLOGY

The image in which the secret message will be embed ca be referred as Cover Image. In this technique, the preprocesses all will be done on the cover image. At first, boundary will be traced for the cover image and then the edge pixels will be detected using Canny algorithm. The Cover image is initially converted into grayscale image. The secret message or information will be embedded into the edge pixels of the cover image, so an edge detection algorithm will be applied to cover image so as to detect the edges in the cover image. Canny edge detection will be considered among other algorithm (from Table 1) since it can detect a greater number of edges compared to other algorithms. Canny detection uses convolution of original image with Gaussian filter eliminates the effect of noise, so it is more resistant to noise.

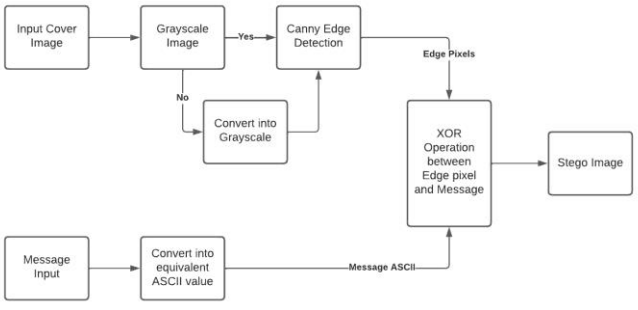

Fig.3 Message Hiding into Edge pixel process

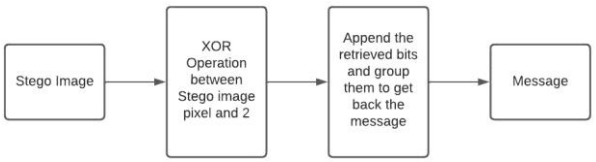

Fig.4 Message Retrieving process from the Stego Image

To embed the data into image, the data must be in binary to perform XOR operation. So, initially the data will be converted into binary data using ASCII values for characters and then decimal to binary conversion, which gives binary data for the secret message. XOR operation will be performed between the edge pixel value of cover image and binary data of secret message. The resultant XOR value will be added to the pixel value of Stego image. Stego image will be formed form the resultant of cover image pixel value and secret message data.

```
1.  LSB=mod (cover (i, j),2);
2.  a=str2double(binary_all(count));
3.  temp=double (xor (LSB, a));
4.  stego (i, j) =cover (i, j) +temp;
```

The stego image will be sent over internet from transmitter to receiver. At the receiver's end the secret message should be retrieved back. The message will be retrieved back by computing modulo 2 operations (remainder) between the stego image pixels values. This will result in binary format of the secret message, which can be converted into characters using 'bin2dec' function.

2.1. Algorithm Implementation

```
1.  % Read in original RGB image.
2.  rgbImage = imread('godavari1.jpg');
3.  subplot (2, 2, 1);
4.  imshow(rgbImage)
5.  axis ('on', 'image');
6.  title ('Original Image')
7.
8.  % Convert to gray scale.
9.  grayImage = rgb2gray(rgbImage);
10. subplot (2, 2, 2);
```

```
imshow(grayImage)
axis ('on', 'image');
title ('Grey Scale Image')

% Get edges
Canny_img = edge (grayImage, 'Canny');
%Canny_img= cannydetector(grayImage);
subplot (2, 2, 3);
imshow (Canny_img, [])
axis ('on', 'image');
title ('Edge Detected Image')
```

Code 1. Canny Edge Detection on the Cover Image

```
%=====================================
%Encoding the message
%=====================================
ascii=uint8(message);
binary_separate=dec2bin(ascii,8);
binary_all='';
for i=1: strlength(message)
    binary_all=append (binary_all, binary_separate (i, :));
end
count=1;
for i=1: row
  for j=1: column
    if count<=len
      LSB=mod (cover (i, j),2);
      a=str2double(binary_all(count));
      temp=double (xor (LSB, a));
      stego (i, j) =cover (i, j) +temp;
      count=count+1;
    end
  end
end
imwrite(stego,'Stego_Image.jpg');
```

Code 2. Message Hiding into the edge pixels Implementation

```
%=====================================
%Decoding the message
%=====================================
message_in_bits='';
for i=1: row
  for j=1: column
    if count<=len
      LSB=mod (stego (i, j),2);
      message_in_bits=append (message_in_bits, num2str (LSB));
      count=count+1;
    end
  end
end
i=1;
original_message='';
while i<=len
original_message=append (original_message, char (bin2dec (message_in_bits (1, i: i+7))));
  i=i+8;
end
```

Code 3. Message decoding from the edge pixels Implementation

3. RESULTS AND CONCLUSIONS

Fig 5 is the given input image, the corresponding grayscale image is generated and referenced as Cover image in Fig 6. After the embedding the message into the Cover image the resultant Stego Image, Fig 7 is generated which is used for transmission over the internet. At the receiver side the Stego image is take as input and the message is retrieved using the decryption process. In this present implementation, two images godavari1.jpg of 2816 x 2112 pixels and lena512.jpg of 512 x 512 pixels color digital images has been taken as cover image, tested for full embedding capacity and the results are given. The effectiveness of the proposed Steganography technique has been studied by calculating MSE and PSNR for the message 'SASTRA UNIVERSITY'

The MSE is calculated by using the equation,

$$MSE = \frac{1}{MN} \sum_{i=0}^{M} \sum_{j=0}^{N} (Xij - Yij)2$$

Fig 5. Cover Image

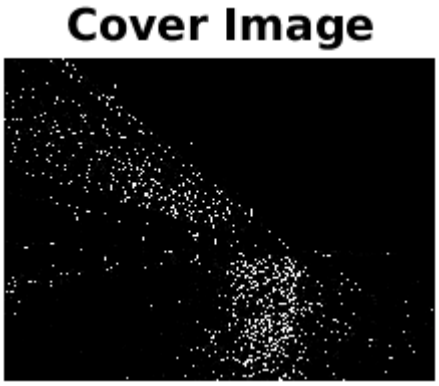

Fig 6. Edge Detection Image

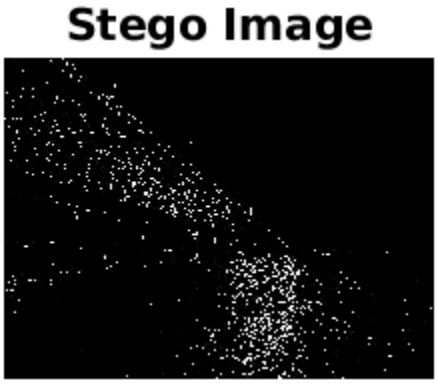

Fig 7. Final Image with Message hidden

where X_{ij} is Stego value and Y_{ij} is the cover object.

The PSNR is calculated using the equation

$$PSNR = 10 \log \left(\frac{I^2}{MSE} \right) dB$$

where "I" is the intensity value of each pixel which is equal to 255 for 8-bit gray scale images. Higher the value of PSNR better the image quality

Table 2.1 Test Image – godavari1.jpg (2816 x 2112)

Metrices	Results
Entropy	0.9125
SNR	23.4931
MSE	7.4037
PSNR	39.4363

The message is hidden/camouflaged into an image using edge detection and XOR operation. Here Canny edge detection algorithm is used for edge pixel detection. Output Image (Stego image) pixels values are modified based on the XOR operation between cover image pixels and secret message data. Large data can also be embedded into an image using this technique.

REFERENCES

[1]. Rengarajan Amritharajan, Benita Bose, Sasidhar Imadbathuni and John Bosco Balaguru Rayappan. "Security Building at the Line of Control of Image Stego", International Journal of Computer Applications, 2010.

[2]. R. Amritharajan, Akhila R, P. Deepikachowdaarapu. "A computer Analysis of Image Steganography", International Journal of Computer Applications, May 2010.

[3]. Rupali Bharadwaj, Vaishali Sharmab. "Image Steganography Based on Complemented Message and Inverted bit LSB Substitution".

[4]. Jhilik Guha, Orchi Saha, Riya Roy. "Improved edge-based Image Steganographic Technique".

[5]. Saiful Islam, Mangat R Modi, Phalguni Gupta. Edge-based Image Steganography. EURASIP Journal of Information Security, 2014.

[6]. Ismail KICH, El Bachir AMEUR, Youssef Taouil. "Image Steganography on Edge-Detection Algorithm, IEEE, 2018 International Conference.

[7]. Ramandhan J. Mastafa, Chrostian Bach. "Information hiding in Images using Steganography Techniques", Northeast Conference of the American Society of the American Society for Engineering Education (ASEE), 2013.

[8]. Rupali Roy. "Image Steganography using Python", May 7, 2020.

[9]. Reem Abdulrahman Alomirah. "An Edge-based Steganography Algorithm for Hiding text into images", Unitech Institute of Technology, Auckland, New Zealand, 2019.

[10]. M Saritha, Vishwanath M Khadabdi, M Sushravya. "Image and text Steganography with Cryptography using MATLAB", International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), 2016.

[11]. Larry S. Davis. "A survey of edge detection techniques", Computer Graphics and Image Processing, Volume 4, Issue 3.

[12]. Paul bao, Lei Zhang and Xiaolin Wu. "Canny Edge Detection Enhancement by Scale Multiplication", IEEE Transactions on pattern analysis and Machine Intelligence, Vol 27, No 9, September 2005