IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-5, May 2018 E-ISSN: 2347-2693

# An Efficient and Secure Steganography Technique Using Edge Adaptive Technique

**Neha Singla[1], Khushil Kumar Saini [2*]**

[1]Division of Computer Engineering, Netaji Subhas Institute of Technology, University of Delhi, New Delhi, India
[2]Division of Computer Engineering, Netaji Subhas Institute of Technology, University of Delhi, New Delhi, India

[*]*Corresponding Author: khushil.nsit@gmail.com, Tel.: +91-9911176744*

**Available online at: www.ijcseonline.org**

Accepted: 17/May/2018, Published: 31/May/2018

*Abstract*— This paper presents an edge based embedding technique to achieve undetectability in steganography with minimal amount of distortion. In this method, desired information is embedded in the edges, as the edges are least susceptible to Human Visual System (HVS) and therefore, remains concealed. Further, the proposed technique uses Majority Parity Check to improve the PSNR of image. The prominent issue with steganography is the amount of distortion caused due to embedding. The proposed technique offers the flexibility of attaining high security with minimal distortion. Experimental results show that the technique provides reduction in the distortion by 20% in comparison to other well-known techniques available in literature.

***Keywords***— Steganography, Edge Adaptive, LSB, Majority Parity Check

## I. INTRODUCTION

Since the evolution of internet, security is the most difficult thing to maintain. Sometimes the message transferred by us may be read by intruders and the security of message is very crucial especially in case of military information. For the security of messages steganography is used. Steganography is a process of concealing a secret message behind a media which can be an image, video, audio, text file i.e. 'covered writing' is done where message is covered by digital media. Generally, image is preferred due to high redundancy and more accuracy in display.

Watermarking is a process similar to steganography but is used to verify the authenticity and integrity of owner [1]. Basic difference between watermarking and steganography is that in watermarking the information about details of owner is concealed on the other hand in steganography information that should be understood only by sender and receiver is concealed. Watermarking may be invisible and visible on the other hand steganography is only invisible. Invaders might not be able to even sense that a particular image contains a secret message.

Basic steganography model is as shown in figure 1.1. Embedding algorithm is where the embedding of secret message in cover image takes place. A pseudo random key is used in embedding algorithm so that message can be randomly inserted in image. When message, key and cover image are passed as an argument to embedding algorithm stego image is produced. To retrieve the message same pseudo key is used in decoding algorithm. Edges are preferred for concealing the message because the smooth/flat areas in the cover images will certainly be changed after concealing of data even at a small embedding rate, and this will result in poor visual quality and low security [2]. Figure 1.2(a) shows cover image, Figure 1.2(b) shows the cover image edge pixels, Figure 1.2(c) shows the stego image edge pixels Figure 1.2(d) shows the distinction between stego image and cover image edge pixels. As seen in figure 1.2(c) human cannot distinguish between cover image and stego image with naked eyes.

Structure of the paper is as follows: Section 2 contains the literature review. An effective edge-based steganography procedure is explained in Section 3. Section 4 describes the experimental results.

 

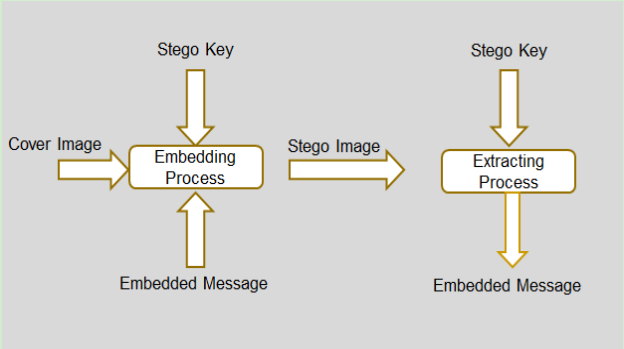


*Figure 1.1: Basic Steganography Model.*


*Figure 1.2: (a) cover image (b) cover image edges (c) stego image edges (d) difference between cover image edges and stego image edges [3].*

## II. RELATED WORK

There are many techniques to conceal message secretly in an image. These techniques are classified into two domains: Spatial domain and Temporal domain. In Spatial domain, the message is directly embedded in the pixels of the image without converting the image. Spatial domain consists of LSB replacement and LSB matching algorithms. In LSB replacement, the LSB of pixels are replaced according to the message to be embedded. LSB replacement may be 1-LSB or 2-LSB, in 1-LSB least significant bit of the pixel is replaced while in 2-LSB two least significant bit are replaced. 2-LSB is more secure than 1-LSB as it is not detectable by Steganalysis [4,5].

In LSBM (LSB matching) there is a slight change from LSB replacement. If the LSB of probable pixels do not match with the bits of message, then message bit is either subtracted or added to LSB randomly or according to the algorithm to be used. Statistically, the probability of decreasing or increasing for each modified pixel value is identical, so the observable asymmetry changes done by LSB replacement can be easily evaded. Therefore, the techniques used to detect LSB replacement fails to detect the LSBM.

In transform domain, concealing of secret message is done by adjusting the Least Significant Bit of non-zero DCT coefficients of the cover image. In place of discrete wavelet transform or DCT Fourier transform, or any other representation of images could be used. W. r. t. robustness, transform domain is better against attacks than the spatial domain but it takes more computational time and has limited embedding capacity [6].

The goal of the steganographic techniques are imperceptibility, undetectability and embedding capacity. Towards this goal the first technique was pixel value differencing which converts a 2-D image to 1-D and calculate the difference between intensity of two adjacent pixels, more the difference of intensity more is the number of bits inserted [7]. This technique lacks security from attacks as the pixels could be easily traced out due to noticeable change in adjacent pins of histogram [8]. Generally, edges are considered for embedding the message because human visual system is quick to detect changes in smooth region as compared to edges. So, it's logical that we embed secret message at edges. Various edge adaptive techniques are there, Sobel introduced based on the largest number of gradients among R, G and B planes [9] But sobel operators are sensitive to noise and inaccurate results while extracting because data is embedded sometimes more than once. Prewitt introduced prewitt operators [10] for finding edge pixels that uses same equation as sobel operator varying some constant value, prewitt operators have problem similar to sobel operator. The Laplacian of Gaussian (LoG) operator is also used for finding edge pixels using the Laplacian filter for Marr's edge detection. The downside is that the accuracy for detecting edge pixels reduces at curves and corners [11] . [12] introduced an approach based on Gaussian filter called as canny method, it gives better edge detection due to noise resistant. The advantage of canny is better signal to noise ratio. Table 2.1 shows comparison between various techniques.

*Table 2.1: Comparison between various techniques.*

| Edge | Advantage | Disadvantage |
|---|---|---|

 

| Operator | | |
|---|---|---|
| PVD | • Simplicity.<br>• High Embedding Capacity. | • Possibility of detecting the message is increased.<br>• Structural symmetry between edges is there. |
| Sobel Method | • Simplicity. | • Does not assure a high embedding rate.<br>• Sensitivity to noise.<br>• Data Extraction is sometimes incorrect. |
| LOG | • It is easier to find the correct position edges. | • Reduces accuracy at corners and curves. |
| Prewitt | • Results are more accurate than Sobel. | • Sensitivity to noise.<br>• Inaccuracy as gradient magnitude of edge decreases. |
| Canny | • Better edge detection in presence of noise.<br>• Improving SNR.<br>• Randomization. | • Time Consuming. |

## III. METHODOLOGY

Here, we proposed a new steganography procedure in which we are using canny edge detection to find the probable pixels for embedding the secret message and for further reducing the distortion caused to image, we are using a majority vote strategy [13] that causes minimal distortion. The main drawback of using traditional edge detection techniques is that the edge pixels found in stego image does not completely match with the edge pixels found in the cover image. This happens because of the minor modification produced in the image due to embedding. To overcome this drawback, we have used masking. We have calculated the edges after masking the 2 LSB of each pixel. So, when we will find the edges in stego image during decoding then also we will find the edges after masking 2 LSB. This will lead to exactly matching of edges in stego image and cover image. Various stages of the proposed embedding technique are depicted in the flow chart in Figure 3.1 and proposed decoding technique is shown in Figure 3.2.

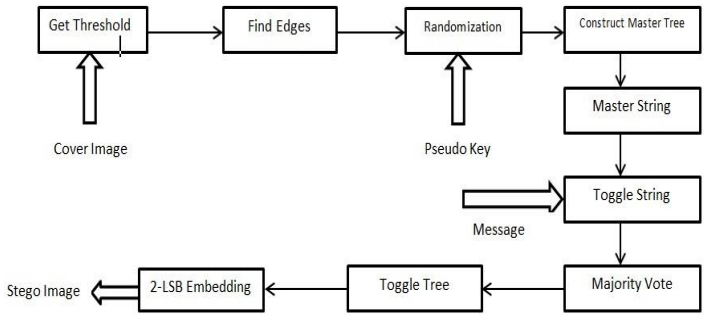


*Figure 3.1: Embedding Algorithm (Proposed)*

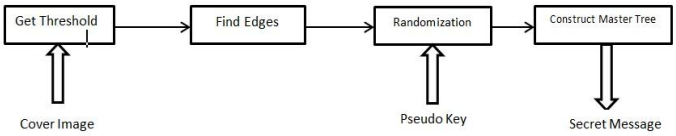


*Figure 3.2: Decoding Algorithm (Proposed)*

### A. *Calculating Threshold*

Canny introduced by [12] uses a two threshold for calculating the edges: upper threshold ($t_h$) and a lower threshold ($t_l$). Upper threshold is used in Identifying strong edges and lower threshold is used in finding the weaker edges. The intensity of the edge found depends on the threshold used. Upper threshold is calculated using message size such that the edge pixels found are sufficient for the message. Experimentally, ($t_l$) is taken to be 0.4* $t_h$.
Outcome of threshold is shown in Figure 3.3. Algorithm 1 is used to calculate the threshold for finding edges. Binary search is used to calculate the threshold. Binary search will return a value for which number of edges found will be more than the size of message and the value should not be very large, if the value will be large then weaker edges will also be included. A variable limit is used to set the upper limit on the number of edges, which is 1% of message size is sufficient.

**ALGORITHM 1**: Get threshold (I, m, w)

// I is cover image
// m is message information to be embedded
// w is Gaussian kernel width used as a canny parameter.
RESULT: threshold ($t_h$) for canny edge detection
limit = 0.1* | m |;
$t_{max}$ = 1;

 

$t_{min}$ = 0;
Bool = false;
while ( bool == false)
{
$t_h$ = ($t_{max}$ + $t_{min}$ )/2 ;
$n_e$ = getEdgePixelCount (Canny(I, $t_h$ , $t_l$ ,w))
diff = ne − |m| ;
if(diff > limit)
$t_{min}$ = $t_h$ ;
else if (diff < 0)
$t_{max}$ = $t_h$ ;
else
bool = false ;
}

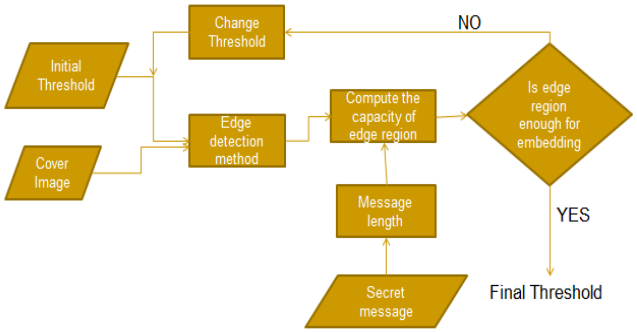


*Figure 3.4: Flowchart of get threshold.*

The flow of algorithm of calculating threshold is shown in Figure 3.4. Threshold calculation is done in such a way that only required edge region obtained, this is done to have a preference of strong edges over weaker edges.

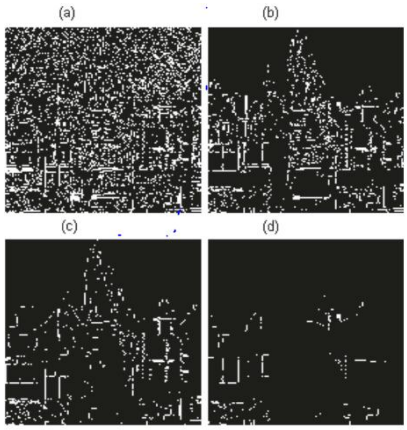


*Figure 3.3: Effect of threshold (a) threshold = 0 (b) threshold = 0.25 (c) threshold = 0.50 (d) threshold = 0.75*

### *B. Proposed Embedding algorithm*

**Algorithm**: embedding (I, p, m, w, n)
// I is the cover image
// p is the pseudo key
// m is message information to be embedded
// w is Gaussian kernel width used as a canny parameter.
// n is used in constructing master tree and toggle tree
RESULT: stego image (s)
S = I
**Step 1**: mask the image s, for masking bitand(s, 252)
**Step 2**: Compute the threshold, get threshold(s, m, w)
**Step 3**: Calculate e, e is the edges acquired by using canny edge detector
e = canny (s, $t_h$, $t_l$, w)
**Step 4**: Random permute edges e = random permute (e, p) and random permute image s = random permute(s, p)
// Shuffle edges and image.
**Step 5**: Construct n-ary master tree similar to TBPC [14] , add three fields in node of master tree : LSB from each edge pixel , which LSB is embedded i.e. LSB or second LSB as 2-LSB embedding is used ,pixel number from image it will help in identifying which pixel is to be modified (Figure 3.5).
**Step 6**: Master string is constructed from master tree by performing exclusive-or operation to all leaf to root path (Figure 3.6).
**Step 7**: Performing exclusive-or on master string and message will give toggle string.
**Step 8**: Apply majority parity check algorithm [13] on toggle string to construct toggle tree.
**Step 9**: Check for the node with value 1 in toggle tree and modify the corresponding pixel data value LSB in master tree.
**Step 10**: Embed width and threshold in non-edge pixels, non-edge pixels e` = complement (e) then random permute (e`, p).
**Step 11**: Modify the LSB of non-edge pixel to embed width and threshold of Gaussian kernel.
// width and threshold are 16 bits IEEE 754 half precision floating point.
**Step 12**: s = random permute(s, p). Image s is stego image.
// reshuffle image to get stego image.

| Pixel Number | Data Value | Bit No. |
|---|---|---|

*Figure 3.5: Tree node structure*

 

### C. *Proposed Decoding algorithm*

**Algorithm**: Decoding (I, p, w, n)
// I is the cover image
// p is the pseudo key
// w is width of Gaussian kernel
// n is used in constructing master and toggle tree
RESULT: message (m)
**Step 1:** mask the image I, for masking bitand (I, 252).
**Step 2:** Calculate e, the edges by using canny based edge detection algorithm
e = canny (s, $t_h$, $t_l$, w).
**Step 3:** Random permute edges e = random permute (e, p) and random permute image s = random permute(s, p)
// Shuffle edges and image.
**Step 4:** Construct n-ary master tree similar to TBPC, add three fields in node of master tree: LSB from each edge pixel, which LSB is embedded i.e. LSB or second LSB as 2-LSB embedding is used, pixel number from image it will help in identifying which pixel is to be modified.
**Step 5:** Master string is constructed from master tree by performing exclusive-or operation to all leaf to root path, master string is the message.

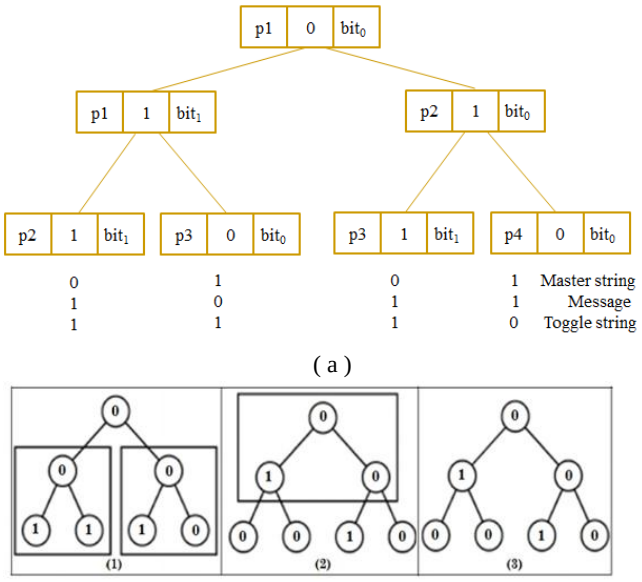


*Figure 3.6: (a) Master tree (b) Construction of toggle tree*

## IV. RESULTS AND DISCUSSION

The proposed technique was developed in C++ using the openCV library. The result of proposed algorithm has been shown in table below varying the value of n (amount of branching of tree constructed). To evaluate efficiency of proposed steganography method we have used four parameters. First is percentage saving in modification obtained due to MPC combined with canny edge detection. Second is visual quality evaluation which will include PSNR, WPSNR and SSIM calculation. Third is security evaluation i.e. how secure is our proposed algorithm.

### A. *Percentage of pixel modification saving*

Generally, most data hiding algorithm leads to 50% modification in embeddable sites of cover image. Using MPC with canny edge detection leads to saving in modifications. When n = 2 then MPC acts as TBPC.

$$P_{saving} = \frac{(Modification_{canny} - Modification_{MPC})}{Modification_{canny}}$$

We have used 5 jpeg images (shown in figure 4.1) and run the code with MPC and without MPC. Data collected is shown in Table 4.1, Comparison of modification with MPC and without MPC and percentage saving in modification is shown.

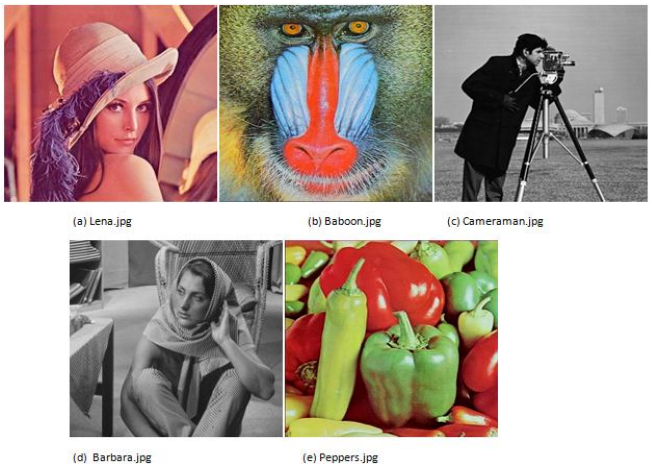


*Figure 4.1: Test Images*

### B. *Visual Quality Evaluation*

The assessment of visual quality is done by calculating PSNR (peak signal to noise ratio), wPSNR (weighted peak signal to noise ratio) and SSIM (structural similarity index). Higher the PSNR, better the image quality. PSNR is calculated as:

$$PSNR = 20\log_{10}\left(\frac{255}{MSE}\right)(dB)$$

Where MSE is mean square error between cover image and stego image.

 

$$MSE = \frac{1}{WH}\sum_{i=1}^{W}\sum_{j=1}^{H}(c_{ij} - s_{ij})^2$$

Where h and w are the height and width of cover image, both cover and stego images have same width and height. $S_{ij}$ and $C_{ij}$ are the value of pixel of Stego image and cover image respectively. PSNR simply measures the distortion between two images, it does not take human visual system into deliberation. For that purpose, wPSNR is used

$$wPSNR = 10\log_{10}\left(\frac{\max(C)^2}{\|NVF(S-C)\|^2}\right)(dB)$$

$$NVF = NORM\left\{\frac{1}{1+\delta^2}\right\}$$

NVF is as defined [5] NORM is a normalization function which that change the value of pixels of image in range 0 to 1.

Table 4.1: Percentage of pixel modification savings

| | N = 2 | | N = 3 | | N = 4 | |
|---|---|---|---|---|---|---|
| | CANNY | PROPOSED | CANNY | PROPOSED | CANNY | PROPOSED |
| LENA | 4493 | 4092 | 4493 | 3374 | 4493 | 3727 |
| CAMERAMAN | 4517 | 3980 | 4517 | 3299 | 4517 | 3709 |
| BARBARA | 4468 | 4003 | 4468 | 3322 | 4468 | 3742 |
| PEPPERS | 4562 | 4021 | 4562 | 3357 | 4562 | 3672 |
| BABOON | 4508 | 3998 | 4508 | 3427 | 4508 | 3722 |
| Average % of pixel modification saving | 10.87 % | | 25.56% | | 17.612% | |

Table 4.2 shows the value of assessment measures of image and compared it with other popular technique. The results of technique (pvd plus tbpc [5]) are almost similar to our proposed technique only for n = 2, but our algorithm is more secure as in PVD pixels could be effortlessly traced by visible change in histogram and the results slightly depend on the message taken into consideration. The security of message is equally important as least distortion in the image. Table 4.3 shows the visual quality assessment of images we have varied the embedding rate for each image and for each embedding rate we for different value of n have calculated PSNR, wPSNR and SSIM. In Table 4.3, the average values of result of five images are written.

Table 4.2: Comparison with other techniques

| Embedding Rate | Algorithm | PSNR | wPSNR | SSIM |
|---|---|---|---|---|
| 30 | PVD | 54.287 | 68.476 | 0.9994 |
| | PVD with TBPC | 57.287 | 71.254 | 0.9994 |
| | Proposed | 62.3058 | 77.2821 | 0.9995 |
| 50 | PVD | 52.515 | 67.415 | 0.9982 |
| | PVD with TBPC | 55.35 | 68.007 | 0.9988 |
| | proposed | 57.65 | 70.658 | 0.9990 |

Table 4.3: Visual Quality Assessment

| Embedding rate | Value of n | PSNR | wPSNR |
|---|---|---|---|
| 5 | 2 | 67.4709 | 81.3629 |
| | 3 | 68.7708 | 82.4785 |
| | 4 | 67.5459 | 80.2187 |
| 10 | 2 | 67.4040 | 81.1741 |
| | 3 | 67.8431 | 81.3971 |
| | 4 | 67.3111 | 80.6824 |
| 15 | 2 | 65.1030 | 79.2915 |
| | 3 | 66.1117 | 80.2633 |
| | 4 | 65.4866 | 79.1379 |
| 20 | 2 | 63.8091 | 77.9852 |
| | 3 | 64.8096 | 78.9432 |
| | 4 | 64.3166 | 78.3146 |

### *C. Security Evaluation*

The main characteristic of steganography is undetectablity and embedding capacity. By undetectablity what we mean is that intruders should not be able to detect that message is embedded in image. Steganalysis is breaking the stegaonographic system i.e. detecting that whether steganography has been used or not. Steganalysis tools are generally of three types: visual detectors, structural detector and non-structural detector. Visual detectors are those in which human visual system is able to detect that image contains secret message. Proposed algorithm has used edge based technique and in edge based technique human visual system is not sensitive to embedding sites. Structural detectors are those that use structural properties of image to analyze if there is any distortion or not. Non-structural uses classifier model to detect distortion, in classifier model classifier is trained using the training set. Here, we have used 2-LSB embedding that is not noticeable by structural detectors and non- structural detectors, as 2-LSB embedding

 

violates essential hypothesis of structural detectors. So, proposed technique is much secure from Steganalysis and moreover we have used randomization in our algorithm that makes it more non-vulnerable.

## V. CONCLUSION

Here we have explored a different technique for steganography. The technique was aimed at minimum distortion of a cover image while embedding. The minimum distortion in cover image helps in preserving the basic principle of steganography i.e. undetectability, lesser is distortion lesser is the possibility of detection by intruders. The pixel chosen for embedding are the edge pixels which are not sensitive to human visual system. In addition to minimum distortion this approach is able to resist targeted attacks while keeping an acceptable state of the stego image. We have experimentally verified the approach is effectively reducing the distortion in image approximately to 20% less distortion.

## REFERENCES


[1] R. Agrawal, *"Hippocratic Databases"*, In the Proceedings of the 28th International conference on Very Large Data Bases, VLDB Endowment, 2002.

[2] W. Luo, F. Huang, J. Huang, *"Edge adaptive image steganography based on LSB matching revisited"*, IEEE Transactions on Information Forensics and Security, Vol. 5, No. 2, pp. 201-214, June 2010.

[3] S. Islam, P. Gupta, *"Revisiting least two significant bits steganography"*, In the Proceedings of 8th International conference on intelligent information processing (ICIIP), Seoul, Republic of Korea, pp. 90-93, April 2013.

[4] A. Ker, *"Steganalysis of embedding in two least-significant bits"*, IEEE Transactions on Information Forensics and Security, pp. 46-54, April 2007.

[5] H. Al-Dmour, N. Ali, A. Al-Ani, *"An efficient hybrid steganography method based on edge adaptive and tree based parity check"*, In the Proceedings of 21st International Conference, MMM 2015, Sydney, NSW, Australia, January 2015,

[6] H. Al-Dmour, A. Al-Ani, *"A steganography embedding method based on edge identification and XOR coding"*, Expert Systems With Applications, Vol/ 46, pp. 293-306, March 2016.

[7] W. Da-Chun, W. H. Tsai, *"Steganographic method for images by pixel-value differencing"*, Pattern Recognition Letters, Vol. 24, Issues 9–10, pp. 1613-1626, June 2003.

[8] X. Zhang, S. Wang, *"Vulnerability of pixel-value differencing steganography to histogram analysis and modification for enhanced security"*, Pattern Recognition Letters, Vol 25, Issue 3, pp. 331-339, Feb 2004.

[9] N. Kanopoulos, N. Vasanthavada R. L., *"Design of an image edge detection filter using the Sobel operator"*, IEEE Journal of Solid-State Circuits, Vol. 23 , Issue 2, pp. 358-367, Apr 1988.

[10] A. Seif, M. M. Salut, M. N. Marsono, *"A hardware architecture of Prewitt edge detection",* In the Proceedings of IEEE Conference on Sustainable Utilization and Development in Engineering and Technology", pp. 99 – 101, Nov. 2010.

[11] L. Chen, I. W. Tsang, D. Xu, *"Laplacian Embedded Regression for Scalable Manifold Regularization"*, IEEE Transactions on Neural Networks and Learning Systems, Vol. 23, Issue: 6, pp. 902 – 915, June 2012.

[12] J. Canny, *"A Computational Approach to Edge Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence",* Vol. PAMI-8 , Issue 6, pp. 679 – 698, Nov. 1986.

[13] K. More, J. Rao, *"Hiding Data in Images with Least Distortion using Majority Vote Parity Check."*, In the Proceedings of Third International Conference on Computational Intelligence and Information Technology – CIIT 2013, Mumbai, India, Oct. 2013.

[14] Hou, Chung-Li, et al, "An optimal data hiding scheme with tree-based parity check", *Image Processing*, IEEE Transactions on Image Processing, Vol. 20, Issue 3, p. 880-886, March 2011.


## Authors Profile


*Ms. Neha Singla* received her M. Tech in Information Systems from Division of Computer Engineering, Netaji Subhas Institute of Technology, Delhi University, Delhi, India. Her area of research lie in Steganography. She completed her M. Tech dissertation under guidance of Mr. Khushil K. Saini.

*Mr. Khushil K. Saini* received his B.Tech. in Instrumentation Engineering from Kurukshetra University, Kurukshetra and M. Tech. in Computer Technology from Indian Institute of Technology, Delhi, India. Presently, He is working as Assistant Professor in Division of Computer Engineering, Netaji Subhas Institute of Technology, Delhi University, Delhi, India. His research areas lie in the area of Steganography, Digital watermarking, image processing.