pISSN 1976-913X
eISSN 2092-805X

J Inf Process Syst, Vol.9, No.3, September 2013 http://dx.doi.org/10.3745/JIPS.2013.9.3.405

A New Approach for Information Security using an Improved Steganography Technique

Mamta Juneja* and Parvinder Singh Sandhu**

Abstract—This research paper proposes a secured, robust approach of information security using steganography. It presents two component based LSB (Least Significant Bit) steganography methods for embedding secret data in the least significant bits of blue components and partial green components of random pixel locations in the edges of images. An adaptive LSB based steganography is proposed for embedding data based on the data available in MSB's (Most Significant Bits) of red, green, and blue components of randomly selected pixels across smooth areas. A hybrid feature detection filter is also proposed that performs better to predict edge areas even in noisy conditions. AES (Advanced Encryption Standard) and random pixel embedding is incorporated to provide two-tier security. The experimental results of the proposed approach are better in terms of PSNR and capacity. The comparison analysis of output results with other existing techniques is giving the proposed approach an edge over others. It has been thoroughly tested for various steganalysis attacks like visual analysis, histogram analysis, chi-square, and RS analysis and could sustain all these attacks very well.

Keywords—Adaptive LSB Steganography, AES; Hybrid Feature Detection, Random Pixel Embedding, Steganography, Two Component based LSB Steganography

1. INTRODUCTION

The initial work on LSB steganography was on LSB Substitution and was explored by (Chang et al., 2002), (THIEN et al., 2003), (Wang et al., 2000, 2001), (Chang et al., 2003, 2006) and (Chan et al., 2001, 2004), which substitutes the same number of bits of each and every pixel of input host images for hiding the secret text or message and give rise to PoVs. They are easily attacked by the Chi-square Test given by (West fled et al., 1999), (Provos et al., 2002) and (Stanley, 2005). LSB Matching was introduced by (Ker et al., 2004) and researched by (Mielikainen, 2006), (LI, 2009),(Luo et al., 2010),(Kumar et al., 2012) and was attacked by (Ker, 2005) based on the Center of Mass (COM) of the Histogram Characteristic Function (HCF). Adaptive LSB was worked on by (Lie et al., 2000), (Liu et al., 2004), (Kekre et al., 2008) and is based on variable number bits substitutions. However, it is unable to utilize HVS masking characteristics completely and is affected by the edge masking effect. The PVD methods are among the most popular methods. They were explored by (Wu and Tsai, 2003), (Park et al., 2005), (Wu et al., 2005), (Yang et al., 2006), (Jung et al., 2008), (Liu et al., 2008), (Wang et al., 2008), (Yang et al., 2008), (Maleki et al., 2011), (Liao et al., 2011) and (Mandal et al., 2011). PVD

Manuscript received March 29, 2013; accepted July 8, 2013.
Corresponding Author: Mamta Juneja
* University Institute of Engineering and Technology, Panjab University, Chandigarh, India (er_mamta@yahoo.com)
** Rayat and Bahra Institute of Engineering and Biotechnology, Mohali, India (parvider.sandhu@gmail.com)

techniques follows the principle that the edge areas that are high in contrast, intensity, and transitions can tolerate more changes than smooth areas. But the problem with these techniques is their inability to make the proper demarcation in edges and textures. Moreover, these techniques were complex to hide large amount of data in images. PVD techniques were easily attacked by (Zhang et al., 2004). The edge detection filter based technique was utilized by (Alwan et al., 2005), (Negi et al., 2006), (Hempstalk, 2006), (Singh et al., 2007), (Chen et al., 2010), (Hussain, 2011) and (Bassil et al., 2011) for steganography in gray images. But the advancement in image technology to RGB leads to steganography application for color images. Pixel indicator techniques introduced by (Gutub et al., 2008, 2009) and (Gandharba et al., 2011) for color images had the major drawback of treating all color components (red, green, and blue) equally. This contradicted the Hecht principle, which reveals that the visual perception of intensely red objects is highest, followed by intensely green objects, and is least for intensely blue objects. (i.e., red plays the most significant and blue plays a least significant role in color formulation.) So, we can integrate maximum changes in a blue component and average changes in a green component and the least amount of changes in a red component without making much of a difference to the color image. Color component based techniques researched by (Imran et al., 2007), (Chang et al., 2008), (Roque et al., 2009) and (Mandal et al., 2012) were not fully tested for all types of attacks, like targeted and universal ones, and were focused on a single component. They didn't utilize cryptography and random sequence generator techniques to make these techniques more resistant to attacks. (Chen et al., 2010) proposed a steganography technique using a hybrid filter, but tested it for gray images. Moreover it wasn't tested for targeted and universal attacks. A capacity of 2.8 bpp (bits per pixel) is good, but the highest PSNR value attained was only 28.6 db. (Mandal, 2011) achieved 49% PSNR but didn't even address the capacity factor and (Hussain et al., 2011) achieved the highest PSNR for very small text messages. (Liao et al., 2011) achieved 39db PSNR with good capacity but also explicitly mentioned in their research paper that they targeted quality and capacity as a tradeoff between capacity, quality, and robustness. They compromised resistance to attacks for quality and capacity. (Kekre et al., 2009) and (Husain et al., 2010) also worked on quality and capacity but this was successfully attacked by (Singh et al., 2012). The three most required evaluation criteria for any good steganography techniques are robustness, imperceptibility, and capacity. There is no technique so far for color images that would fully target all of these criteria. So there is an urgent requirement for a technique that would provide good capacity and high PSNR value and that is resistant to all targeted, as well as universal steganalysis attacks, for color images.

2. THE PROPOSED SYSTEM

The present research provides improved LSB based steganography techniques, which will work in a spatial domain, for hiding data in a 24-bit bitmap color image. It integrates the following three new techniques:

- The hybrid feature (line/edge/boundary/circle) detection technique integrates the canny and enhanced hough transform for bifurcating an input cover image into an edge and smooth areas.
- The two components based LSB substitution technique for hiding encrypted messages in the

edges of given cover image.
- An adaptive LSB Substitution technique for hiding messages in in the smooth areas of given cover image.
- In addition to the above techniques, AES is used to encrypt input text files and random pixel embedding is incorporated to embed data at random pixel locations.

The various algorithms utilized in the proposed system are as explained below.

P1. The Hybrid Feature Detection Technique for Extracting Edges and Smooth Areas

A new hybrid feature detection technique for extracting edges and smooth areas from an image is being proposed. It integrates the canny edge detection proposed by (Canny, 1986) and the enhanced hough transform edge linking technique given by (Hough, 1962).

P11. Edge Detection Using Canny Edge Detector: The canny edge detector is widely considered to be the standard edge detection algorithm in the industry. It is known as an optimal edge detector due its good detection abilities, good localization, and minimal amount of false edges. It uses a multi-stage algorithm to detect a wide range of edges in images.

Algorithm:
1. Smoothing: blurring of the image to remove noise.
2. Finding gradients: the edges should be marked where the gradients of the image have large magnitudes.
3. Non-maximum suppression: Only local maxima should be marked as edges.
4. Double thresholding: potential edges are determined by thresholding.
5. Edge tracking by hysteresis: final edges are determined by suppressing all of the edges that are not connected to a very certain (strong) edge.

P12. The Enhanced Hough Transform:

The hough transform technique is used to refine the output of the canny edge detector. It is used to detect peaks, edge links, lines, and circles in the output image retrieved from the canny edge detector**.** The various steps that need to be followed in the enhanced hough transform are as listed below:

a. Conversion from the xy plane to ρθ plane: this function used to transform xy plane to ρθ plane in which the hough transform works.

Syntax:

[htm, θ, ρ] = convert_xy_ ρθ (eps, δθ, δρ)

Inputs: eps is the edge pixels image that is retrieved after applying the canny edge detector on the given input cover image. δθ and δρ represents the hough transform bins along the θ axis and the ρ axis, respectively

Outputs: convert_xy_ρθ () converts the xy-plane to the ρθ-plane. A matrix called the hough transform matrix, which is denoted by htm, represents the output obtained after carrying this plane transformation. This output hough transform matrix is nρ x nθ in which nρ = 2*ceil (norm (size (eps))/ δρ) -1 and nθ = 2*ceil(90/ δθ). Here θ and ρ are element vectors specifying

the angle in degrees corresponding to each column and row of htm respectively.

b. Peak Detection: The function hough_peaks_detect () is used to detect peaks in the hough transform matrix retrieved in a. The steps that are to followed are as listed below:

i Locate all of the cells in the hough transform matrix with a maximum value, as compared to other cells, and store their positions.

ii Suppress all the rest of the Hough transform cells of the hough transform matrix, which are in the neighborhood with the cells in step i to zero.

iii Repeat steps i and ii until the desired threshold is reached or until the target number of peaks is not reached.

Syntax:

[row_cord, col_cord, htm_p]= hough_peaks_detect (htm, max_num_of_peaks, threshold, neighborhood)

Inputs: Max_num_of_peaks specifies the maximum number of peaks to be traced from htm (output matrix from step a). Any value for a hough transform cell in htm that is below the given threshold value is not to be taken as a peak. A neighborhood is a two element vector that defines the location of hough transform cells in htm and is to be provided with suppression of values. These are the cells around each identified peak, which are suppressed to zero.

Outputs: htm_p is the output matrix that successfully represents all of the peak locations and the suppressed neighborhood. Each and every peak is identified by their row and column coordinates row_cord and col_cord, respectively.

c. Line Detection and Linking:

a. After a successful detection of all the peaks in the hough transform matrix, the next step is to determine the line segments associated with those peaks along with the start and end of those segments. So, the locations of all non-zero pixels, which formulate that peak, are determined. The same is done using hough_pixels_detect ().

Syntax:

[row_cord, col_cord] = hough_pixels_detect (htm_p, θ, ρ, row_bins, col_bins)

Inputs: htm_p is the hough transform matrix with all peak locations returned from Step b. θ and ρ are the axis' to work in and row_bins and col_bins are the row and columns bins positions returned by hough_peaks_detect().

Outputs: hough_pixels_detect returns the row and column coordinates for all of the pixels that formulate the line segments.

b. The next step is to trace the line segments formulated from the pixels determined in Step a. The pixel coordinates returned by using the hough_pixels_detect () are joined together to formulate the line segment. Function hough_lines_detect () work according to the following steps:

1. Rotate the pixels along the vertical line.
2. Sort the pixel locations.
3. Locate gaps in different line segments and merge line segments whose gap is less than the minimum gap that has been specified.
4. Return only those line segments with a length that is greater than the minimum length specified.

Syntax:

Lines_detected=hough_lines_detect (htm_p, θ, ρ, row_bins, col_bins, min_gap_in_lnsegments, minimal_length)

Inputs: htm_p is a hough transform matrix with all of the peak locations returned from Step a. θ and ρ are the axis' to work in given by function convert_xy_ ρθ (); row_bins and col_bins are rows and columns bins positions returned by hough_peaks_detect ().

Outputs: Line segments

hough_lines_detect () detects all line segments in the output matrix htm_p are traced using the given row_bins and col_bins. Any two retrieved line segments from the same hough transform bin which are separated by a gap of less than the specified min_gap_in_lnsegments are combined into a single segment. Now, if the length of this newly formulated segment is less than the minimal_length, all of pixels forming it are discarded.

d. Circle Detection: It is done using the function hough_circle_detection (), which detects the multiple circles in an image using hough transform.

Syntax:

Circles=hough_circle_detection (eps, min_rad, max_rad, min_edge_pxs_ratio, max_circle_diff);

Inputs: Here eps is the input image for the hough_circle_detection () function, which is retrieved after application of the canny edge detector on the input cover image and so a gray image min_rad minimal radius is possible and max_rad is the maximum radius that is possible for the circle in pixels. The min_edge_pxs_ratio is the ratio of the minimal amount of edge pixels of a circle that are detected out to the perimeter of a circle. It is generally between 0 to1 (0<min_edge_pxs_ratio>1) and its default value taken is 0.3 and so this makes it an optional argument. max_circle_diff is the maximum possible difference between two circles for them to be considered as the same one whose default value taken is 12 and so is an optional argument. For example, if fst_circle (p1, q1, r1) and sec_circle (p2, q2, r2) are the two circles detected then max_circle_diff = |p1-p2|+|q1-q2|+|r1-r2|.

Outputs: It is the m-by-4 array of m circles (p, q, r, no_of_pixels) where p and q are the center coordinates and r is the radius and no_of _pixels gives the count of pixels covered.

The description of steps for hough_circle_detection () are as listed below.

a. The input validation step:

- If argument max_circle_diff is missing, use its default value of 12.
- If the min_edge_pxs_ratio is missing, use its default value 0.3.
- If the number of arguments<3, display an error message.
- Make sure all argument values are positive.

b. Create a three dimensional hough array in which the first two dimensions specify the coordinates of the circle center and the third argument specifies the radius.

c. Edges are detected using the canny edge detector, while resetting the value for threshold to balance between the performance and detection quality.

d. (ep_x-max_rad to ep_x+max_rad, ep_y-max_rad to ep_y-max_rad) are the possible locations for circle centers for an edge pixel (ep_x, ep_y) so mapping is done in the same manner.

e. Formulate the grid (0 to d_max_rad, 0 to d_max_rad), and then compute the distances between the center and all the grid points to form a radius. Here d_max_rad is double the maximum radius max_rad.

f. Increment the corresponding elements in the hough array for each edge pixel determination.

g. Mark these circles highlighted on input cover image while deleting duplications.

P13. Hybrid Feature Detection:

The algorithm reads a 24-bit color image denoted by CI= {cp1, cp2, cp3,……..cpn-1}, where cpi is the ith pixel in the image and n is the total number of pixels.

In the same context, every color pixel cpi can be represented as cpi={rci[rc0,….rc7], gci[gc0,…..gc7], bci[bc0,……bc7]} where i is the index of the ith pixel, rci is the ith bit of color component rc, gci is the ith bit of color component gc, and bci is the ith bit of color component bc. Here CI is a 24-bit image so each of its pixels is made up of 3 color components—each of which is 8 bits in length.

2. The algorithm converts CI into a gray image, as denoted by f(CI)=GI, where GI stands for gray image. The purpose of this conversion is to ease the processing of subsequent steps.

3. Three parameters: 1) the size of the Gaussian filter, 2) a low threshold, and 3) a high threshold are automatically chosen where the results of the filter are optimal.

4. The canny edge detection algorithm is executed on GI as described above in Section P11 using the three parameters selected in Step 3. The results are a collection of lines, curves, and points denoting the edges or the boundaries of the objects in the image GI. The pixels that constitute the extracted edges are represented as EP= {ep1, ep2, ep3 …epr-1} where ej is the jth pixel that makes up the edges and r is the total number of these pixels.

Thereafter we applied the enhanced hough transform to extract various other features like shapes, lines, and circles.

The output image after applying the canny edge detector has distorted edges, which are not properly joined with each other and so edge linking is required to fill in the edge gaps. Therefore, a global edge linking technique (i.e., the hough transform) has been applied, as described above in P12, on the output image obtained by the canny so as to obtain more refined edges. The hough transform can even detect lines, edges, links and shapes that were not traceable through the Canny. The edge pixels that were retrieved after applying the hough transform are represented as:

HEP= {hep1, hep2, hep3 …hepq-1}, where hepj is the jth pixel in the image and q is the total number of edge pixels.

P2. An Adaptive LSB Substitution for Smooth Areas

We have used the algorithm provided by (Kekre, 2009) and modified it to get smoother areas. In this approach a variable number of LSBs would be utilized for embedding secret message bits, in accordance with this algorithm.

For all pixels across smooth areas:

1. If the value of the red component (sp_rc), green component (sp_gc), and blue component (sp_bc) of the smooth pixel spi is in the range of 240 to 255 (i.e., 240<=spi_rc<=255, 240<=spi_gc<=255, and 240<=spi_bc<=255) then utilize all of the bits of the blue component for embedding data. This is done by first checking that the 4 MSBs all equal 1.

2. If the value sp_rc, sp_gc, sp_bc is in the range of 224 to 239 (i.e., 224<=spi_rc<=239, 224<=spi_gc<=239 and 224<=spi_bc<=239) then utilize 6 LSBs of the blue component for embedding of data This is done by checking that the first 3 MSBs all equal 1.

3. If the value sp_rc, sp_gc, sp_bc is in the range of 192 to 223 (i.e., 192<=spi_rc<=223,

192<=spi_gc<=223 and 192<=spi_bc<=223) then utilize 5 LSBs of the blue component for embedding of data. This is done by checking that the first 2 MSBs both equal 1.

4. If the value sp_rc, sp_gc, sp_bc is in the range of 0 to 191 (i.e., 0<=spi_rc<=191, 1<=spi_gc<=191 and 0<=spi_bc<=191) then utilize 4 LSBs of the blue component for embedding for embedding of data. This is done by checking that the 1st MSB equals 1.

This is illustrated in Table 3.1.

For all of the components (red, green, blue) of each and every pixel in a color image across the smooth areas (except those already embedded by the above algorithm), the following embedding process is employed:

1. If the value of the current pixel component (the first 4 MSB's are all 1's) say for example, the cpci, is in the range of 240 <= cpci>=255, then we utilize 4 LSBs of that component for embedding.

2. If the value of the cpci (First 3 MSB's are all 1's), is in the range of 224 <= cpci>=239 then we utilize 3 LSBs of that component for embedding.

3. If the value of the cpci (the first 2 MSB's are all 1's), is in the range of 192 <= cpci>=223 then we embed 2-bits of secret data into the 2 LSB's of that component.

4. And in all other cases where values are in the range of 0 <= cpci>=192 we embed 1-bit of secret data into 1 LSB of that component.

This is illustrated in Table 3.2.

A similar procedure is adapted for extracting the hidden text from the image.

Table 3.1. Per Pixel Embedding Chart

sp_rc	sp_gc	sp_bc	Utilized bits	Total bits available/pixel
240-255	240-255	240-255	sp_rc -4 LSBS sp_gc-4LSBs **sp_bc-8LSBs**	16 bits
224-239	224-239	224-239	sp_rc -3 LSBS sp_gc-3 LSBs **sp_bc-7 LSBs**	13 bits
192-223	192-223	192-223	sp_rc -2 LSBS sp_gc-2 LSBs **sp_bc-6 LSBs**	10 bits
0-191	0-191	0-191	sp_rc -1 LSB sp_gc-1 LSB **sp_bc-5 LSBs**	7 bits

Table 3.2. Per Pixel Component Embedding Chart

range of the smooth pixel components (sp_rc or sp_gc or sp_bc)	Utilized bits/Component
240-255	4
224-239	3
192-223	2
0-191	1

P3. The Two Components-Based LSB Substitution for Edge Areas

In this method, the 8-bits of the first component (blue component) for the image pixels would be replaced with secret text message bits followed by a secret message being embedded into the 4 least significant bits of the green component.

For all edge pixels retrieved through hybrid feature detection:

a. Read each edge pixel provided by the hybrid feature detection that was described in Section 3.1. Each edge pixel can be represented as:

epi={ Ri[r0, r1, r2, r3, r4, r5, r6, r7], Gi[g0, g1, g2, g3, g4, g5, g6, g7],
Bi [b0, b1, b2, b3, b4, b5, b6, b7]}

Here, I is a 24-bit image so each of its pixels is made up of 3 color components red (R), green (G), and blue (B). Each of which is 8-bits in length. i is the index of the ith pixel, ri is the ith bit of the color component R, gi is the ith bit of color component G, and bi is the ith bit of the color component B.

b. Embed message bits in epi (Gi[g4,g5,g6,g7],Bi[b0, b1, b2, b3, b4, b5, b6, b7]) in the sequence b7 to b0 and then g7 to g4.

c. Embed data until:

i The end of the message data is not reached.
-OR-
ii The end of the edge pixels is reached.

d. If the end of the message data is reached then the embedding is over and in the other case embed the rest of the message data using by using the adaptive LSB approach given in section P2.

The utilizations bits in this technique are 12-bits out of a total of 24-bits of pixels.

A similar procedure is used for extracting message bits from edge pixels.

P4. Encryption and Decryption Using AES

An input text file is encrypted using the standard encryption technique AES, as defined in (FIPS, 2001), in order to provide two tier security for the proposed system. It ensures that the message will not be understood by any, even in the case that its existence is disclosed due to being encrypted.

1. Encryption using AES:

For each round (except the final round) with the state and key as inputs, repeat the following steps:

a. Substitute_State_Bytes()
b. Shift_State_Rows()
c. Mix_State_Columns()
d. Add_Round_key()

For the final round, Step c is not performed.

Step 1: Substitute_State_Bytes (State)

(a) Each byte of the block is replaced with its substituent in the S Box.
(b) Each byte is taken as independent.
(c) A single S Box is used for all of the states.

Step 2: Shift_State_Rows (State)

(a) Each row of the state is moved a certain number of steps in a cyclical manner.
(b) The number of shifts a row undergoes is different for different rows.

Step 3: Mix_State_Columns (State)
(a) State columns are considered as polynomials over the Galois Field (2^8).
(b) Each and every state column is a multiplication modulo of a pair of polynomials.

Step 4: Add_Round_Key (State, Key[i])
Perform XOR operation between received round key and state.

2. Decryption using AES

For each round (except the final round) with the state and key as inputs, repeat the following steps:

a. Inverse_ Substitute_State_Bytes()
b. Inverse_Shift_State_Rows ()
c. Inverse_Mix_State_Columns()
d. Inverse_Add_Round_key()

For the final round, step c is not performed. All of the above decryption steps are the same as for encryption, except they should be done in reverse.

P5. Random Pixel Embedding and Extraction

Random pixel embedding, as given by (Schneier et al., 2003), has been implemented in the present work by using the Linear Congruential Generator (LCG). It is also known as the Pseudo Number Generator (PRNG), which is probably one of the most commonly used techniques out of the family of pseudorandom techniques, and is used for generating random numbers. With a given seed it will keep generating random numbers, this is why it selects random pixels to embed in both edge and smooth areas.

1. Random Pixel Embedding:

Before being embedded into any pixel, a random pixel is selected using the function random_pixel_generator (), which is defined below.

a. Initialize the values for the increment, multiplier, initial seed, and the maximum_ possible_value. (An initial seed is any value from 0 to 16,777,215 for a 24-bit color bitmap image and the maximum possible value is 1,677,215).
b. Assign values for the seed, increment, multiplier, and the maximum_possible_value as long as the 4 conditions listed below are satisfied.
   1. The increment must be relatively prime to maximum possible value.
   2. multiplier-1 must be a multiple of every prime p that divides the maximum_possible_value.
   3. multiplier-1 must be a multiple of (a number) if the maximum_possible_value is a multiple of a number.
   4. The seed, increment, multiplier, and maximum_possible_value must all be greater than 0.
c. Compute the Random_Pixel_Value= Multiplier * Seed + Increment mod (maximum possible value).
d. Repeat step c while keeping the values of the increment, multiplier, and maximum pos-

sible value as the same. However, the random_pixel_value generated becomes the seed for the next random number, in order to generate all values from 1 to the maximum possible value.

e. Return the Random_Pixel_Value.
Do this so that the output will be a random pixel number.

2. Random Pixel Extraction

During extraction the same procedure is repeated while facilitating the selection of the same pixels. Random pixel extraction will select the same set of pixels for the same set of value increments, multipliers, initial seeds, and the maximum range.

3. IMPLEMENATION OF THE PROPOSED SYSTEM

The proposed system is comprised of the two components, as shown in Figure 3.1.

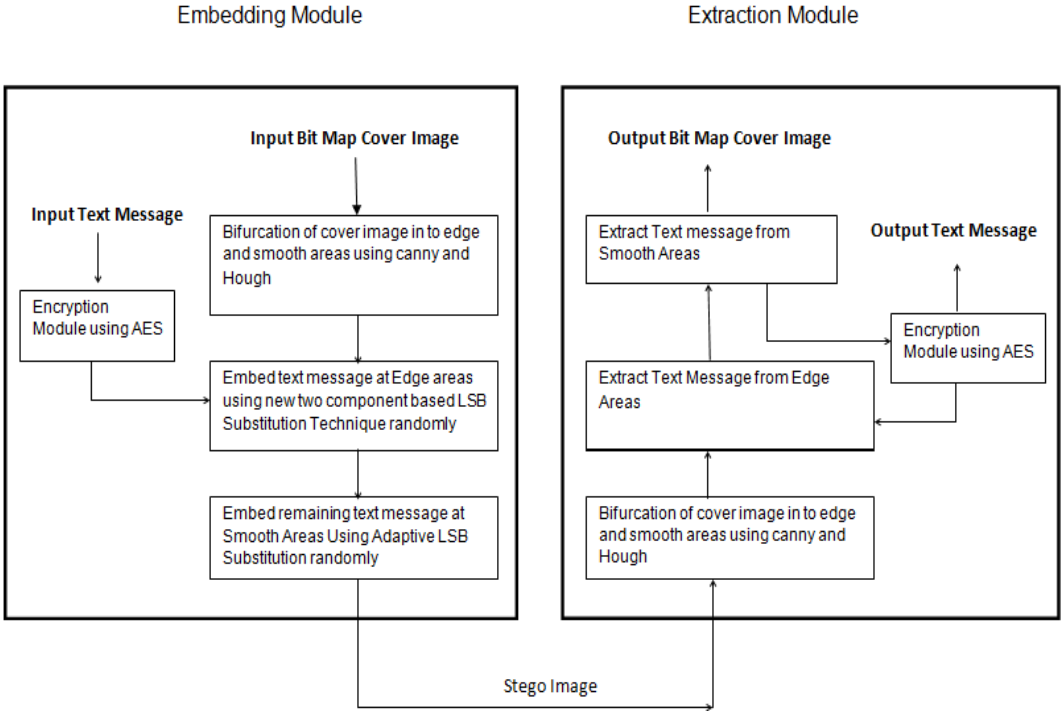

Fig. 3.1. Proposed System

3.1 The Embedding Module

The main algorithm for the embedding stage can be explained as follows:

1. Input the secret text/image file that is to be hidden in the cover image.
2. Select the cover image (BITMAP file) from the list of stored image files and the text

files.

3. Extract the input cover image to the edge and smooth areas by using the new hybrid feature detection filter described in Section P1.
4. Calculate the size of the secret text file.
5. The secret text data is first encrypted using the advanced encryption standard described in Section P4.
6. Substitute the encrypted secret characters from Step 5 to cover the image that was randomly obtained from Step 3 as explained in Section P5.
   For edge areas, embed secret text data using new two component based LSB substitution technique described in section P3 and procedure 3.1.1.
   For smooth areas, embed the secret text data using adaptive LSB method, as described in Section P2 and Procedure 3.1.2.

3.1.1 Procedure for Embedding Edge Areas

- Extract all of the pixels marked as edges, as explained in section P1, of the given cover input image and store values retrieved in the array called the Cover_Edge_Pixel_Array.
- Extract all the characters of the given text file (to be hidden in the cover image) and store the values retrieved in an array called the Message_Character_Array.
- Extract all of the characters from the stego key in the Key_Array.
- Choose the first pixel and pick characters from the Key_Array and place characters of Key_Array in 8 bits of the blue component of a pixel. If there are more characters in the Key_ Array, then place the rest of them in the 4 bits of green component and then into the next pixel and so on until there are characters in the Key_Array.
- Place a terminating symbol to indicate end of the key. 'EOF' has been used as a terminating symbol in this algorithm.
- Place the characters of Message_Character_Array in 8 bits of the blue component and 4 bits of the green component of pixel while repeating it until all of the characters have been embedded or until we have reached the last edge pixel.
- If it is applicable, you will then need to place a terminating symbol to indicate the end of data.
- The obtained stego output image will hide all of the characters that we input.

3.1.2 Procedure for Embedding the Smooth Areas

a. Extract all of the remaining pixels of the given input image, which are not marked as edges, as explained in section P1, and store them in Cover_Smooth_Pixel_Array.
b. For each red, green, blue component of each and every pixel stored in Cover_ Smooth_Pixel_Array, repeat the following steps till we reach end of Message-Character-Array or end of Cover_Smooth_Pixel_Array.
   i Calculate number of bits available as given in section P2.
   ii Place the remaining characters from Message-Character-Array into available bits of Cover_Smooth_Pixel_Array.

3.2 Extraction Module

The main algorithm for the extraction stage can be listed as follow:

- Extraction of Input cover image to Edge and smooth areas using new hybrid feature detec-

tion filter as in section P1.
- Extraction of secret text message from stego image is carried from random pixels of cover image using Function defined in section P5.
- For edge areas: Extract data from 8 bits of blue component and 4 least significant bits of green component as described in section P3 and Procedure 3.2.1
- For smooth areas: Extract data from adaptive number of bits as described in section P2 andProcedure 3.2.2.
- Apply AES Decryption method described in section P4.

3.2.1 Procedure for Extraction for Edge areas

a. Extract all pixels marked as edges as described in P1 of given cover input image and store in the array called Cover_Edge_Pixel_Array.
b. Now, start scanning pixels from Cover_Edge_Pixel_Array and keep extracting key characters from first and second (partial) components of all pixels to Key-Array until you get the terminating symbol.
c. If this extracted key matches with the key entered by the receiver, then again start scanning next pixels and extract secret message characters from first (blue) and second (partial green) component of next pixels and place it in Message_Character_Array until you the get terminating symbol or you reach the last edge pixel.

3.2.2 Procedure for Extraction for Smooth Areas

a. Extract all remaining pixels of input cover image which are not marked as edges as explained in section P1 and store them in the Cover_Smooth_Pixel_Array.
b. Calculate the number of bits that available in each red, green, and blue component of each pixel, respectively, that is being used for embedding, as explained in Section P2.
c. Now, start scanning pixels from the Cover_Smooth_Pixel_Array and keep extracting characters from the number of bits, as determined by Step a, in the character array until you get the terminating symbol.

4. RESULT ANALYSIS

R1. Evaluation Criteria: Steganography techniques are broadly evaluated in 3 aspects, as mentioned by (Peticolas et al., 2000), (Wang et al., 2004), and (Morkel et al., 2005).

1. Evaluation Criteria I: Imperceptibility/Quality
2. Evaluation Criteria II: Capacity or Payload
3. Evaluation Criteria III: Robustness or Resistance to Attacks

1. Evaluation Criteria I: Imperceptibility/Quality

This is a quantifiable measurement of how finely a secret message has been embedded into a given cover image without anyone knowing about its existence at all. It is scaled to measure the invisibility of hidden information in a stego image. The higher the imperceptibility, the higher its invisibility, and the better the quality of the image is. It is one of the most widely used criteria to evaluate the performance of any steganography technique, which is measured using the Peak Signal to Noise Ratio (PSNR)

PSNR is a metric used for the measurement of imperceptibility or for the quality of a stego

image. Its units are decibels (dB). The higher the value of the PSNR, the higher the imperceptibility or the quality of the image will be. So, all steganography techniques work to acquire more and more PSNR value. It is measured as:

$$PSNR = 10 LOG_{10} \frac{255^2}{MSE} dB$$

$$MSE = \left(\frac{1}{MXN} \right) \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left(P(x,y) - P'(x,y) \right)^2$$

Here, M designates number of rows and N is the number of cover image columns. P(x, y) and P'(x, y) are the pixel values for the cover and stego image, respectively. MSE stands for "Mean Square Error."

2. Evaluation Criteria II: Capacity or Payload

The payload is the scale for measuring the capacity of a cover image to hide a secret message inside it without being noticed by anyone. It provides the measure of the maximum number of bits that can be hidden into the cover image with an acceptable stego image quality.

3. Evaluation Criteria III: Robustness or Resistance to Attacks

The robustness of a steganography system is defined as its immunity against all types of attacks that are carried out by an intruder to extract hidden information.

The success of any steganography system is attained for all three major criteria viz. imperceptibility (calculated through PSNR), capacity and robustness (immunity to visual analysis, histogram analysis, chi square, and RS analysis). Any steganography approach is successful if it is able to provide a higher PSNR and capacity value than existing techniques along with immunity to attacks. So, the proposed approach was thoroughly tested with many test images to test for an attainment of these criteria and the results are given in next sections.

R2. Outcomes for Criteria I and II

The outcome results for PSNR and capacity are shown in Table 4.1 on various cover images after embedding a confidential text file, while varying the size of the text file and cover image.

Table 4.1(a). Embedded Data , % of pixels used in the Image, % of Changed Bytes, Average Number of Bits per Pixel, MSE, and RMSE for Test Images

	Embedded data	% of used Pixel in Image	% of Changed Bytes	Avg. # Bits/ Pixel	MSE	RMSE
LEENA	261,121	0.4545	42.65	1.47	0.055	0.235
BABOON	267,134	0.4552	39.54	1.13	1.447	1.2.33
PEPPER	267,135	0.4545	41.39	1.13	1.433	1.197
FAMILY	270,936	0.4545	41.73%	1.5	0.01	0.122

Table 4.1(b). Embedded Data, Mean, Standard Deviation,PSNR, Capacity for Images

	EmbeddedData	Mean	Standard Deviation	PSNR	Capacity
LEENA	261,121	256.5	361.3	60.70	806,912
BABOON	267,134	91	129.4	46.52	106,496
PEPPER	267,135	92.5	129.4	46.57	102571
FAMILY	270,936	213	299.8	66.37	544,768

R3. Comparison Analysis with Existing Techniques for Evaluation Criteria I and II

The comparison of existing techniques with proposed approach on the basis of PSNR and capacity is shown in Table 4.2.

Table 4.2(a). Capacity and PSNR Value Comparison of Existing Techniques

Technique	OPAP(3 bits)		LSB(3bits)		OLSB(3Bits)		ALSB, HSV	
Author	(Chan et al., 2004)		(Chan et al., 2004)		(Wang et al., 2001)		(Lie et al., 1999)	
	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR
LEENA	786,432	40.7	786,432	37.92	786,432	40.7	786,432	37.92
BABOON	786,432	40.7	786,432	37.92	786,432	40.7	786,432	37.92
PEPPER	786,432	NA	786,432	NA	786,432	NA	786,432	NA

Table 4.2(b). Capacity and PSNR Value Comparison of Existing Techniques (cont'd.)

Technique	Side Match		PVD		Adaptive LSB		PVD Modulus	
Author	(Chang et al., 2004)		(Wu et al., 2003)		(Kekre et al., 2008)		(Wang et al., 2007)	
	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR
LEENA	48,626	41.2	35,827	59.1	35,827	59.1	786,432	37.92
BABOON	57,146	37	34,235	59.4	34,235	59.4	786,432	37.92
PEPPER	50,907	40.8	60,317	56.2	60,317	56.2	786,432	NA

Table 4.2(c). Capacity and PSNR Value Comparison of Existing Techniques (cont'd.)

Technique	PVD, LSB Replacement		Adaptive LSB, PVD		High pay Load		Adaptive LSB Replacement	
Author	(Wu et al., 2005)		(Yang et al., 2008)		(Chen et al., 2010)		(Luo et al., 2010)	
	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR
LEENA	774,970	37.6	807,256	41.39	774,970	37.6	807,256	41.39
BABOON	720,288	34.3	854,096	38.58	720,288	34.3	854,096	38.58
PEPPER	776,160	37.5	800,168	42.42	776,160	37.5	800,168	42.42

Table 4.2(d). Capacity and PSNR value Comparison of Existing Techniques (cont'd.)

Technique	MPVD, Adaptive		DHPVD		Adaptive, Floor, Modulus		ADH, modulus	
Author	(Liao et al., 2011)		(Mandal et al., 2011)		(Joo et al., 2011)		(Chen et al., 2011)	
	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR
LEENA	810,564	39.6	NA	49.45	810,564	39.6	NA	49.45
BABOON	903,580	36.9	NA	46.54	903,580	36.9	NA	46.54
PEPPER	805,492	39.8	NA	48.78	805,492	39.8	NA	48.78

Table 4.2(e). Capacity and PSNR Value Comparison of Existing Techniques via the Proposed Approach

Technique	Adaptive Modulus		NPI		Color PVD		Proposed Technique	
Author	(Maleki et al., 2011)		(Imran et al., 2007)		(Mandal et al., 2012)			
	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR
LEENA	1,055,620	34.4	1.48	42.3	145,787	42.3	806,912	60.7
BABOON	1,108,708	32.2	1.47	38.4	144,916	38.4	756,496	56.5
PEPPER	1,057,584	34.3	1.48	42.3	145,995	42.3	802,571	52.5

From above mentioned results we can say that our proposed approach has proven better in Evaluation Criteria I and II than already exisiting techniques. It provides better imeperceptibilty/ quality and hiding capacity than previous known techniques.

R4. Evaluation Analysis for Criteria III- Robustness/Resistance to Attacks

The robustness of the proposed technique was thoroughly tested through various steganlysis attacks. These included visual analysis and statistical analysis. The various results are explained below.

R41. Visual Analysis

The results of the proposed technique on a 24-bit color image (FAMILY.bmp) can be seen in Figure 5.1. This approach successfully resisted visual attacks as visual difference in the input cover image and the stego image could not be traced as demonstrated in Figure 4.1.

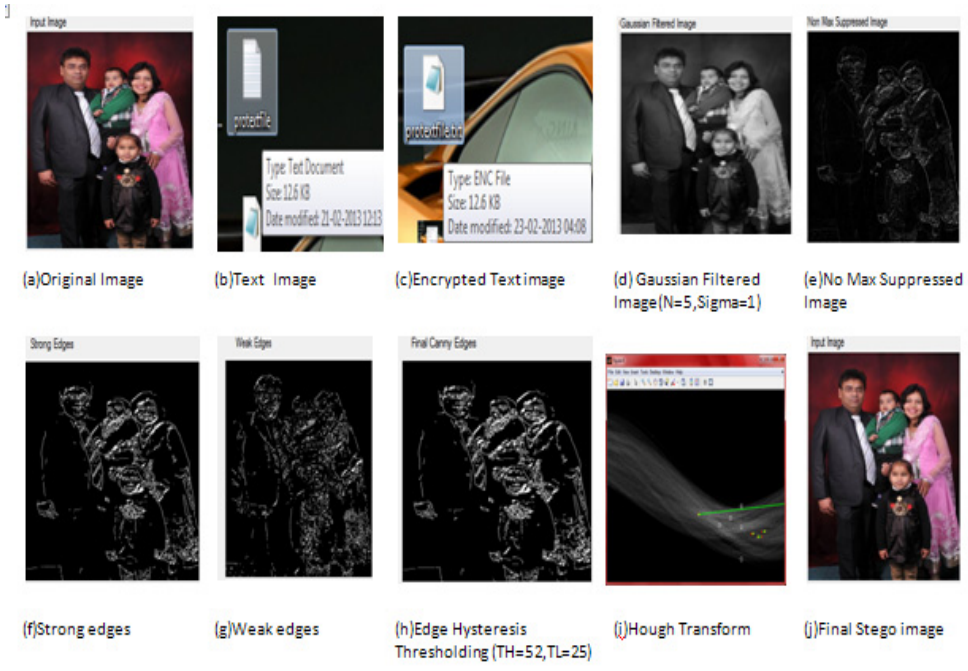

Fig. 4.1. No Visual Differences in the Original Cover Image and the Stego Image

R42. Statistical Analysis: Histogram Analysis

The results of the histogram analysis tecnhnique are shown in Figure 4.2 and no differences were found in the histograms of the original and stego images, so they could not be attacked.

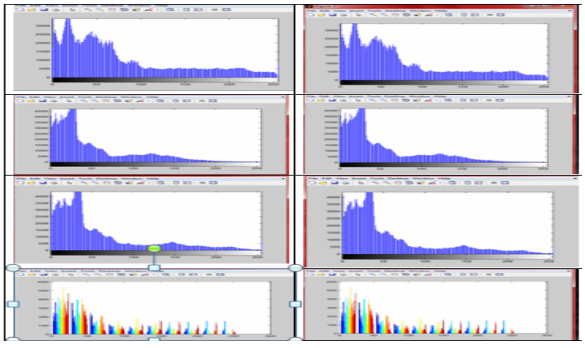

Fig. 4.2. Histogram Analysis of the Red, Green, Blue for all Components of the Original Image and Stego Image (FAMILY.bmp)

R43. Statistical Analysis: Chi-Square Attack

The proposed approach was even tested by the chi square attack and it could succesfully withstand these attacks, as shown in Figure 4.3.

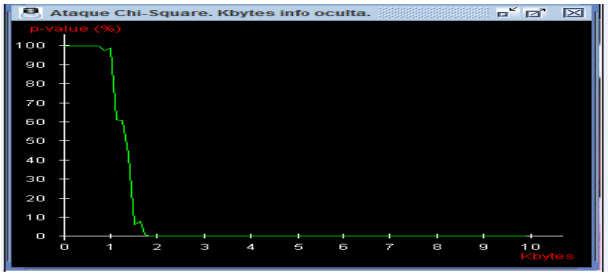

Fig. 4.3. Chi Square Attack on FAMILY.bmp

R44. Statistical Analysis: RS-Analysis

The results of the RS analysis are shown in Figure 4.4 and the proposed approach could not be attacked by this attack.Figure 4.4 and Table 4.3 show the results of RS analysis for the family cover image. The analysis predicts that the difference between RM (Positive Regular) and R-M (Negative Regular) are less than 10%. Furthermore, the difference between SM (Positive Singular) and S-M (Negative Singular) is also less than 10%. This indicates that the image is secured.

Table 4.3. RS Analysis on FAMILY.bmp

	Red	Green	Blue
Positive Regular	23.1582	23.3256	24.9753
Positive Singular	11.5232	11.3452	11.2234
Negative Regular	23.2852	24.2313	25.9876
Negative Singular	10.3221	10.2254	10.1123

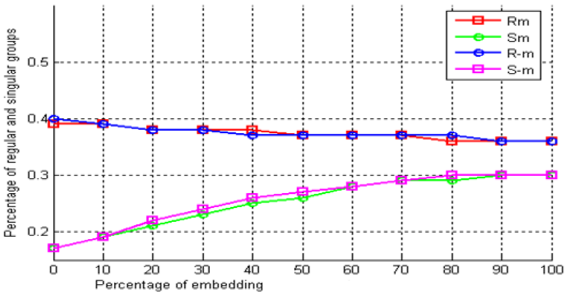

Fig. 4.4(a). RS Analysis of FAMILY.bmp

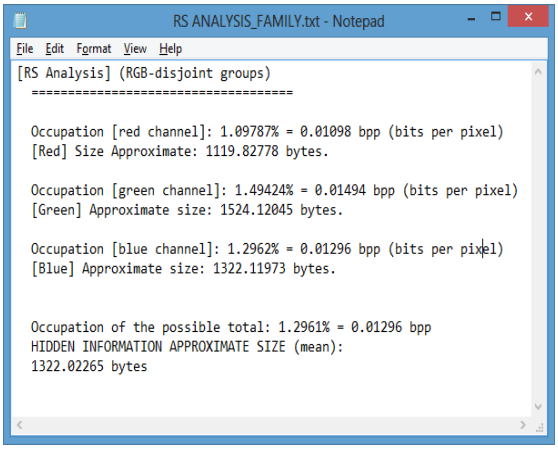

Fig. 4.4(b). RS Analysis of FAMILY.bmp

6. CONCLUSION

This research achieves the goal of implementing a new steganography approach for images. It integrates three new techniques viz. the hybrid feature detection technique; two components based LSB substitution technique and the adaptive LSB substitution technique. It achieved the target of improved imperceptibility (calculated by using PSNR) with minimum MSE (mean square error) as compared with existing techniques. It achieved an improved hiding capacity while utilizing 12 bits out of the total 24 bits of each pixel of RGB color image in the edge areas as well as smooth areas of the cover image. In addition, it also provides a better form of the feature detection technique, which is a hybrid of the canny edge detector and hough transform while providing more refined results. Noise and other disturbances had fewer effects on our results, due to the hybrid feature detector, which was based on a combination of the canny and enhanced hough transform. More robustness is provided to our proposed technique as it is integrated with an advanced encryption standard (AES). A better resistance to attacks is met by combining it with a random pixel embedding technique. The robustness of the proposed approach has been tested through the application of various steganalysis attacks like visual analysis, histogram analysis, chi square analysis and RS analysis. The overall results are satisfactory.

REFERENCES

[1] A. Ker (June, 2005), "Steganalysis of LSB Matching in Gray scale Images", IEEE Signal Processing Letter, vol. 12, no.6, pp. 441– 444.

[2] A. Ker, (May 23-25, 2004), "Improved Detection of LSB Steganography in Grayscale Images". In Proc. 6th International Workshop. Toronto (Canada), Springer LNCS, vol. 3200, p. 97–115.

[3] Adnan Gutub, Ayed Al-Qahtani, AbdulazizTabakh (May 10-13, 2009), "Triple-A: Secure RGB Image Steganography Based on Randomization" Proceedings of IEEE/ACS International Conference on Computer Systems and Applications (AICCSA, 2009), Rabat, Morocco. pp.400-403.

[4] Adnan Gutub, Mahmoud Ankeer, Muhammad Abu-Ghalioun, AbdulrahmanShaheen, and AleemAlvi (18-20 March,2008), "Pixel Indicator high capacity Technique for RGB image Based Steganography", Proceedings of 5th IEEE International Workshop on Signal Processing and its Applications (WoSPA 2008), University of Sharjah, Sharjah, U.A.E.

[5] Ali Shariq Imran, M. YounusJaved, and NaveedSarfrazKhattak (2007) "A Robust Method for Encrypted Data Hiding Technique Based on Neighborhood Pixels Information", World Academy of Science, Engineering and Technology 31 2007.

[6] Arjun, N Santosh.; AtulNegi, (14-17 Nov. 2006), "A Filtering Based Approach to Adaptive Steganography," TENCON 2006, IEEE Region 10 Conference, vol., no., pp.1-4.

[7] AroojNissar, A. H. Mir,(Decemeber,2010), "Classification of Steganalysis Techniques: A Study", Digital Signal Processing, Elsevier,Vol.20 No.6, pp.1758-1770.

[8] Bin Li, JunhuiHe,JiwuHuang,Yun Qing Shi (April,2011),"A Survey on Image Steganography and Steganalysis", Journal of Information Hiding and Multimedia Signal Processing Volume 2, Number 2.

[9] C. C. Thien, J. C. Lin, (2003), "A Simple and High-Hiding Capacity Method for Hiding Digit-By-Digit Data in Images Based On Modulus Function". Pattern Recognition, vol. 36, p. 2875-2881.

[10] C. H. Yang, C. Y. Weng, (December, 2006), "A Steganographic Method for Digital Images by Multi-Pixel Differencing." In Proc. International Computer Symposium. Taipei (Taiwan), pp. 831 to 836.

[11] Cheng-Hsing Yang, Chi-Yao Weng, Shiuh-Jeng Wang, Hung-Min Sun (2008), "Adaptive Data Hiding in Edge Areas of Images with Spatial LSB Domain Systems". IEEE Transactions on Information Forensics and Security, Vol. 3, No. 3,pp. 488-497.

[12] Chi-Kwong Chan, L. M. Cheng, (2001), " Improved Hiding Data in Images by Optimal Moderately-Significant-Bit Replacement", IEE Eelectronics letters, vol. 37, no. 16, p. 1017-1018.

[13] Chi-Kwong Chan, L. M. Cheng, (2004), "Hiding data in images by simple LSB substitution. Pattern Recognition", Vol. 37, p. 469-474.

[14] Chin-Chen Chang, Chi-Shiang Chan, Yi-Hsuan Fan (2006), "Image Hiding Scheme with Modulus Function and Dynamic Programming Strategy on Partitioned Pixels." Pattern Recognition, vol. 39, no. 6, p. 1155-1167.

[15] Chin-Chen Chang, Ju-Yuan Hsiao, Chi-Shiang Chan (2003), "Finding Optimal Least-Significant-Bit Substitution in Image Hiding By Dynamic Programming Strategy". Pattern Recognition, Vol. 36, p.1538-1595.

[16] Chin-Chen Chang, Min-Hui Lin, Yu-Chen Hu (2002), "A Fast And Secure Image Hiding Scheme Based on LSB Substitution", International Journal of Pattern Recognition and Artificial Intelligence,vol. 16, no. 4, p. 399-416.

[17] Chin-Chen Chang, Tseng, H.W. (2004), "A Steganographic Method for Digital Images Using Side Match.", Pattern Recognition Letters, Vol. 25, pp.1431-1437.

[18] Chung-Ming Wang, Nan-I Wu, Chwei-Shyong Tsai, Min-Shiang Hwang (2008), "A high quality steganographic method with pixel-value differencing and modulus function." Journal of Systems and Software, Vol. 81, No. 1, pp. 150-158.

[19] D.C. Wu, W. H. Tsai (2003), "A Steganographic Method for Images by Pixel-Value Differencing", Pattern Recognition Letter, Vol. 24, No. 9-10, p. 1613–1626.

[20] GandharbaSvvalin,Saroj Kumar Lenka (June,2012),"A Novel Approach to RGB Channel Based Image Steganography Technique ",International Arab Journal of e-Technology, Vol. 2, N0. 4.

[21] H. B. Kekre, ArchanaAthawale, Pallavi N. Halarnkar (2008), " Increased Capacity of Information Hiding In Lsb's Method For Text And Image" International Journal of Electrical, Computer, and Systems Engineering, Vol. 2, No. 4, p. 246-249.

[22] H. B. Kekre, ArchanaAthawale,Pallavi N. Halarnkar, (2009)"Performance Evaluation Of Pixel Value Differencing And Kekre's Modified Algorithm For Information Hiding In Images", ACM International Conference on Advances in Computing, Communication and Control (ICAC3).

[23] H.C. Wu, N.I. Wu, C.-S Tsai, M.S Hwang, (2005), "Image Steganographic Scheme Based on Pixel-

Value Differencing and LSB Replacement Methods", IEE Proceedings-Vision, Image and Signal Processing, Vol. 152, No. 5, pp. 611-615.
[24] J. C. Joo, T. W. Oh, H. Y. Lee, H. K. Lee (Jan, 2011), "Adaptive Steganographic Method Using the Floor Function with Practical Message Formats," International. Journal of Innovative Computing, Information and Control, Vol. 7, No. 1, pp. 161-175. ISSN 1349-4198.
[25] J. F. Canny,(1986), "A Computational Approach to Edge Detection". IEEE Trans. Pattern Anal. Machine Intell., Vol. PAMI-8, No. 6, pp. 679-697.
[26] J. K. Mandal and Debashis Das (July,2012),"Colour Image Steganography Based on Pixel Value Differencing in Spatial Domain International Journal of Information Sciences and Techniques (IJIST) Vol.2, No.4.
[27] J.K.Mandal, A. Khamrui, (2011),"A Data-Hiding Scheme for Digital Image Using Pixel Value Differencing (DHPVD)", Electronic System Design (ISED), International Symposium, pp: 347 - 351.
[28] Jarno Mielikainen, (2006), "LSB Matching Revisited", IEEE Signal Processing Letters, Vol. 13, no. 5, p. 285-287.
[29] Jen-Chang Liu, Ming-Hong Shih (2008), "Generalizations of Pixel Value Differencing Steganography For Data Hiding In Images", FundamentaInformaticae, Vol. 83, No. 3, pp. 319-335.
[30] Jessica Fridrich, MiroslavGoljan, and Rui Du. (2001), "Reliable Detection of LSB Steganography in Color and Grayscale Images". In Proceedings of 2001 ACM workshop on Multimedia and Security: New Challenges, pp 27-30. ACM Press, 2001.
[31] Juan José Roque and JesúsMaríaMinguet (2009), "SLSB: Improving the Steganographic Algorithm LSB", 7th International Workshop on Security in Information Systems, 57-66, (2009).
[32] Kathryn Hempstalk, (11- 19 February 2006),"Hiding Behind Corners: Using Edges in Images for Better Steganography", Proceedings of the Computing Women's Congress, Hamilton, New Zealand.
[33] Ki-Hyun Jung, Kyeoung-Ju Ha, Kee-Young Yoo (Aug28-30, 2008), "Image Data Hiding Method Based on Multi-Pixel Differencing and LSB Substitution Methods.", In Proc. 2008 International Conference on Convergence and Hybrid Information Technology (ICHIT '08). Daejeon (Korea), pp. 355-358.
[34] M. Hussain, M. Hussain,(5-6 September, 2011) , "Embedding data in edge boundaries with high PSNR", Proceedings of 7th International Conference on Emerging Technologies (ICET 2011), vol., no., pp.1-6.
[35] M. Hussain,(2010), "Pixel Intensity Based High Capacity Data Embedding Method" International Conference on Information and Emerging Technologies (ICIET), pp.1 -5.
[36] Manglem Singh,,Birendra Singh, ShyamSundar Singh (April, 2007), "Hiding Encrypted Message in the Features of Images", IJCSNS, VOL. 7, No.4..
[37] Markus Kahn, (1995), Steganography Mailing List, 5 July.
[38] Mohammad TanvirParvez and Adnan Gutub (9-12 December 2008), "RGB Intensity Based Variable-Bits Image Steganography", Proceedings of 3rd IEEE Asia-Pacific Services Computing Conference (APSCC 2008), Yilan, Taiwan.
[39] Najme Maleki, MehrdadJalali, M. VafaeiJahan ,(July,2011),"An Adaptive Data Hiding Method Using Neighborhood Pixels Differencing Based On Modulus Function," International Conference on. Information Processing, Computer Vision, and Pattern Recognition (IPCV'11), Las Vegas, Nevada, USA.
[40] Nanhay Singh, Bhoopesh Singh Bhati, R. S. Raw, (2012) ,"A Novel Digital Image Steganalysis Approach for Investigation. International Journal of Computer Applications", 6/1/2012, Vol. 47, pp18 .
[41] Niel F. Johnson, ZoranDuric, SushilJajodia (2000), "Information Hiding, and Watermarking - Attacks & Countermeasures," Kluwer Academic Publishers.
[42] Niels Ferguson and Bruce Schneier, (2003), Practical Cryptography, John Wiley.
[43] Niels Provos and Peter Honeyman (2002), "Detecting Steganographic Content on The Internet". In Proceedings of NDSS'02: Network and Distributed System Security Symposium, pp1-13, Internet Society, 2002.
[44] P. Mohan Kumar, K. L. Shunmuganathan (2012), "Developing a Secure Image Steganographic System Using TPVD Adaptive LSB Matching Revisited Algorithm for Maximizing the Embedding Rate", Information Security Journal: A Global Perspective, Vol. 21, Issue 2.
[45] P.V.C. Hough1962), Method and Means for Recognizing Complex Patterns, U.S. Patent 3069654 .
[46] R. Amirtharajan, J. Qin and J.B.B. Rayappan(2012). "Random Image Steganography and Steganalysis: Present Status and Future Directions", Information Technology Journal., Vol 11, pp 566-576.
[47] R. H. Alwan., F. J. Kadhim, and A. T. Al-Taani, (2005), Data Embedding Based on Better Use of Bits in Image Pixels. International Journal of Signal Processing, 2 (1), 104-107.
[48] Ran-Zan Wang, Chi-Fang Lin, Ja-Chen Lin (2000), "Hiding Data in Images by Optimal Moderately Significant Bit Replacement" IET Electronics Letters, vol. 36, no. 25,pp. 2069-2070.

[49] Ran-Zan Wang, Chi-Fang Lin, Ja-Chen Lin, (2001), "Image Hiding by Optimal LSB Substitution And Genetic Algorithm." Pattern Recognition, vol. 34, p. 671-683.
[50] Shao-Hui Liu, Tian-Hang Chen, Hong-Xun Yao, Wen Gao (Aug. 26-29, 2004), "A Variable Depth LSB Data Hiding Technique in Images". In Proc. 2004 International Conference on Machine Learning and Cybernetics. Shanghai (China), Vol. 7, p. 3990-3994.
[51] Specification for the Advanced Encryption Standard (AES), (Nov. 2001), Federal Information Processing Standards Publication 197.
[52] Stanley, C.A. (2005), "Pairs of Values and the Chi-squared Attack", in CiteSteer. 2005, pp. 1-45.
[53] W. Chen, C. Chang, T. Le, (2010), "High Payload Steganography Mechanism Using Hybrid Edge Detector", Expert Systems with applications, vol. 37, pp 3292-3301.
[54] W. Luo, F. Huang and J. Huang (2010), "Edge Adaptive Image Steganography Based on LSB Matching Revisited", IEEE Trans. Inf. Forensics Security, vol. 5, no. 2, pp.201 -214.
[55] Wen-Nung Lie, Li-Chun Chang (October 24-28, 1999), "Data hiding in images with adaptive numbers of least significant bits based on the human visual system." In Proc. IEEE Int. Conf., Image Processing. Kobe (Japan), pp 286-290.
[56] Westfeld and A. Pitzmann (1999), "Attacks on Steganographic Systems - Breaking the Steganographic Utilities Ezstego, Jsteg, Steganos, and S-tools-and Some Lessons Learned", In Proceedings of the 3rd Information Hiding Workshop, volume 1768 of LNCS, pages 61-76. Springer, 1999.
[57] X. Liao, Q.-Y. Wen, and J. Zhang (2011), "A Steganographic Method for Digital Images with Four-Pixel Differencing and Modified LSB substitution," Journal of Visual Communication and Image Representation, vol. 22, no. 1, pp. 1−8.
[58] X. Zhang,.; S. Wang,(2004): Vulnerability of Pixel Value Differencing Steganography to Histogram Analysis and Modification for Enhanced Security, Pattern Recognition Letters, vol.25, pp. 331-339.
[59] Xiaolong Li, Bin Yang, Daofang Cheng, TieyongZeng (2009), "A Generalization of LSB Matching". IEEE Signal Processing Letters, vol. 16, no. 2, pp. 69-72.
[60] Y. K. Lee, L. H. Chen, (2000), "High Capacity Image Steganographic Model", IEEE Proc., Vis. Image Signal Process, Vol. 147, no. 3, p. 288-294.
[61] Y. R. Park, H. H. Kang, , S. U. Shin, K. R. Kwon, (2005), "A Steganographic Scheme in Digital Images Using Information of Neighboring Pixels.", In Proc. International Conference on Natural Computation. Berlin (Germany), Springer-Verlag LNCS, Vol. 3612, pp. 962–968.
[62] Youssef Bassil (December, 2012),"Image Steganography Based on a Parameterized Canny Edge Detection Algorithm", International Journal of Computer Applications (0975 – 8887) Volume 60– No.4.
[63] Yung-Chen Chou, Chin-Chen Chang, Kuan-Ming Li (2008) , " A Large Payload Data Embedding Technique for Color Images", FundamentaInformaticae, Volume 88, Number 1-2, pp47-61.

Mamta Juneja
An Assistant Professor in University Institute of Engineering and Technology, Panjab University, Chandigarh, India. She did masters in Computer Science from Punjab Technical University, India and currently pursuing Doctorate in the same. Her interest areas include Image Processing, Steganography, Information Hiding and Information Security.

Parvinder Singh Sandhu
Doctorate in Computer Science and Engineering and working as Professor in Computer Science & Engineering department at Rayat & Bahra Institute of Engineering and Bio-Technology, Mohali, Punjab, INDIA. He is editorial committee member of various International Journals and conferences. He has published more than 150 research papers in various referred International journals and conferences. He chaired more than 100 renowned International Conferences and also acted as keynote speaker in different countries. His current research interests are Software Reusability, Software Maintenance, Machine Learning and Image Processing.