Secure and Robust Image Steganography Using a Reference Image as Key

Giridhar Maji, Sharmistha Mandal

Abstract: ***A new steganography scheme using a reference image is proposed with two variants. In the first kind, secret text bits are directly embedded into the cover image after encryption using a reference image. In the second variant secret text is written/printed on a white canvas and then that image containing the secret text is converted into binary (BW) image. Any image could be used as a cover image. A special reference image will act as an encryption key and it is assumed that the used reference image is known to sender and receiver apriori. For a grayscale reference image, any k LSB bit planes could be used to encode the secret binary image bits before embedding them into k LSBs of the cover image. When k=1, the dimension of the secret image becomes the same as that of the reference image and cover image. With k=2, secret binary image size could be doubled or two secret images could be hidden using two LSB bit planes of the reference image and embedding them into two LSB bits of each cover image pixel. Any reversible mechanism could be used for encoding. Here XOR operation will be used for experiments. Two distinct advantages of this generic scheme are (i) It is far more secure than other image steganography techniques as adversary may know the encoding formulae , may collect the stego image but could not get information about the reference image used and which and how many bit planes are used for encoding as there could be millions of images and many combinations of bit planes that could have been employed; (ii) As the secret text is printed and then converted into an image, it overcomes one of the main limitations of LSB image steganography, i.e. robustness to random noise. Even with corrupted Stego image, the extracted binary image becomes corrupted too but most of the secret are still readable.***

This scheme has lower capacity than the capacity that would have been achieved by directly embedding the secret text binary as in the first variant, but this could be overcome by using k>=2 as k=2 doubles the capacity. This whole scheme is generic and any digital media could be used as reference or cover. Experimental results demonstrate the robustness of the second variant against random noise, and standard image quality metrics such as MSE, PSNR, SSIM are evaluated and compared between the two variants.

Index Terms**: *Text in image, LSB steganography, image inside image, reference image as key, secure and robust data hiding.***

I. INTRODUCTION

Digital communication prevails everywhere due to cheap internet and low-cost electronics. Globe becomes a small and connected place due to the emergence of social networking along with many different modes and mediums of communication. Digital data objects (scanned images, snapshots, audio, video, etc.) have become the way of information sharing. Digital communication has provided many benefits like ease of communication, quick delivery, global reachability, etc. But it comes with reduced privacy and security due to the inherent nature of the Internet. The Internet is an unsecured public channel and any data in transmit could be intercepted by anyone. Cryptography techniques are commonly used to encrypt any secret information using an encryption key before sending through the Internet. Upon receiving, the intended recipient can only decrypt the secret information using the valid key. In such a scheme any third party probing the unsecured internet channel gets some scrambled text which he cannot understand, and also cannot decipher as he does not have the encryption key. There exist two important problems with that approach. First, the adversary knows about the secret communication though he may not understand and secondly, an attacker with sufficient computational capability could brute-force the key and decrypt the information. So, it is apparent that only cryptography is not enough to maintain security as well as privacy in today's digital realm. Also, cryptography involves a great amount of computational power and complexity. Steganography [1], commonly known as the closest sibling of cryptography joins hand with cryptography to solve the issue. Steganography literally means the art of hidden writing. In any steganography technique, the goal is to hide the existence of the secret information so that it does not even raise an eyebrow of an attacker to look into intercepted data for further brute-force. Steganography uses some digital cover (can be Text, Image, audio, video) to hide the actual message before transmitting through the Internet so that the attacker does not see it as a secret communication. The best way to security and privacy would be to combine cryptography with steganography [36]. Interested readers may refer to a good study on steganography versus cryptography in [36]. Generally, secret information is encoded and embedded into a cover media using some combination of different parameters (called stego-key) and that key is shared separately with the recipient. Sometimes the secret message is first encrypted using standard cryptography techniques and then hid inside cover media. In such a case the stego-key consists of two parts, the encryption key, and the embed key.

Steganography has many diverse applications and depending on the application the goal changes [2,3]. When it is used for secret communication through innocent cover media, the goal becomes to hide the existence of the communication and making the secret information retrievable with stego-key at the receiver side unaltered. In digital copyright protection, the goal of the steganography scheme becomes to hide the owner's copyright symbol/mark in the creative digital object (photograph, creative art, digital painting, etc.) in such a way that no one can remove it without damaging the cover. Again, steganography is also used in watermarking [7] where the objective is to maintain the

Revised Manuscript Received on May 10, 2019.

Giridhar Maji,, Department of Electrical Engineering, Asansol Polytechnic, Asansol, West Bengal, India.

Sharmistha Mandal, Department of Computer Science and Technology, Kanyapur Polytechnic, Asansol, West Bengal, India.

integrity/authenticity of the digital object. In such a scenario the watermark is hidden in such a way that any image processing, editing will damage the watermark and the originality of the object could be verified by extracting the watermark. Two types of watermarking are normally employed. First one is called robust watermarking where the copyright symbol is visible and used to establish the ownership as can be seen on many images where the logo of the company/owner is visibly watermarked on the image. Another type is called fragile watermarking [5] where any slight modification damages the watermark. An example would be like someone collects a famous photograph and edit it to put her signature on it and then claims it to be the original one. If the original photograph has hidden watermarking done by the owner/authority then forensics could easily identify the forged copy by extracting the hidden watermark from the forged copy and comparing the same with the original watermark.

Research community categorizes steganography based on the cover media used to hide the secret. So, there are image steganography, audio steganography, video steganography, and text steganography. In this paper, we shall focus on image steganography. Depending on the mechanism of hiding the secret into the cover media, image steganography could be of two types viz, spatial domain and transform domain. In the transform domain, the cover image is transformed into the frequency domain and secret information is hidden into the coefficients. There exist many such techniques like DCT [6], DFT [31], DWT [32], etc. In the spatial domain, bitmap image pixels are used in hiding the secret message. Most commonly one or more Least Significant Bits (LSBs) are modified to embed the secret message [4]. Many other spatial domain techniques are proposed such as Pixel Value Differencing [8], Gray level modification [9], Quantization index modulation, Pixel pair matching (PPM) [33], Pixel Mapping Method [34], edge detection based, predictive coding technique, etc. A good survey of spatial domain steganography techniques is available in [26].

Secret information could be a stream of ASCII characters or another image, audio or even video comprising of a set of image frames. In all the cases secret information is converted into an equivalent binary bit stream first and then embedded into the cover image. The cover image could be a 24-bit RGB color image with three 8-bit channels of Red, Green, and Blue or it could be an 8-bit grayscale image. The most common and widely employed image steganography is the LSB based steganography. In such a scheme the LSB of each cover image pixel is used to embed secret bits. Embedding capacity could be greatly increased with the use of more than a single LSB. An image pixel LSB bits are considered as random noise and do not contain any significant information and this fact is exploited in LSB based image steganography techniques. Though LSB techniques are simple and easy to implement and less computation-heavy, it comes with a lack of security and robustness than other transform domain techniques. Introduction of a small amount of noise is detrimental to LSB steganography as it corrupts the secret bits at the LSBs and reduces the robustness.

In this paper, we shall propose a novel secret hiding technique using the secret text as an image and a reference image as stego-key that will overcome both of the above problems without compromising much on the capacity. We shall also use the same scheme with directly embedding secret text bits for comparison.

A. LSB Substitution and Bit-planes

Let us briefly understand the process of hiding secret message bits in the cover image LSBs and bit-plane representation. Let's consider the secret message (text or image) is converted to binary. In case of a text message, each characters' ASCII could be converted to binary. If a secret message is an 8-bit grayscale image then each pixel intensity value ranges between 0-255. So, in binary, each pixel intensity is an 8-bit number. So, pixel intensities for all the pixels are appended together to generate the binary for the secret image. While recreating the image from the binary stream two information is required. The image dimension and the number of bits used for each pixel must be known. In our example suppose the image is 64*64 pixel 8-bit grayscale image, then the total number of secret bits will be 64*64*8. Now let us assume that the cover image is also an 8-bit grayscale image and dimension of the cover image is n*n pixel. Each cover image pixel intensity is an 8-bit binary. The least significant bit (LSB) of pixel intensity has a minimum contribution

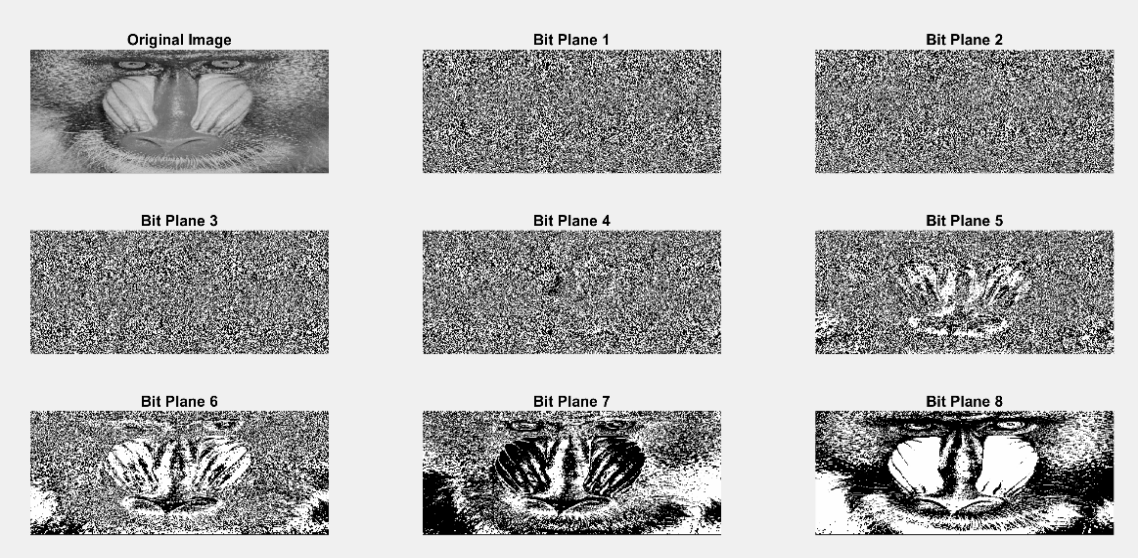


Fig. 1: Original image and different bit planes of an 8-bit grayscale image (baboon) taken from SIPI database

in the image representation. If the LSB of pixel intensity is changed from 0 to 1 or 1 to 0 then the pixel intensity value change is 1/256 which imperceptible. Using the above fact cover image pixel intensity LSB bits are substituted by secret message bits to hide the secret inside the cover image. If only the Least significant bits of each pixel are used then maximum secret message size could be $1/8^{th}$ of cover image size as each pixel hides only 1 bit. Many authors have proposed multi-LSB (mLSB) substitution where more than one LSB bits are used for secret bit substitution and capacity increases accordingly.

If all the least significant bits of all pixels of an 8-bit grayscale image are taken together as a matrix of 0s and 1s then that is known as LSB-bit-plane of the image. Similarly, the bit-planes for each higher order bits could be generated like 2-LSB, 3-LSB, etc. Fig 1. Shows an 8-bit grayscale image and all 8 bit-planes (1-LSB is the Least Significant Bit plane and 8-LSB is the Most Significant Bit plane) of it. It can be observed that 1-LSB bit-plane contains almost no information and consists of random noise and that's why LSBs are targeted for secret embedding.

II. RELATED STUDY

LSB embedding is the simplest secret hiding technique where the secret binary bits are embedded into the cover image pixel LSBs to generate the stego-image. The stego image is then transmitted to the receiver through a public channel. Extraction is also very simple. The recipient has to extract the stego-image pixel LSBs and then group them based on the type of data. If it is secret text then each 8-bit is replaced by equivalent ASCII characters and if the secret message is a grayscale image then each 8 bit represents the image pixel intensity. An attacker could easily extract the secret message if simple LSB embedding is used. Hence classical LSB technique is rarely used. Many researchers have used some kind of shuffling/encryption of secret bits before embedding them into the cover image.

LSB substitution is combined with pixel value differencing (PVD) in many studies [17]. In basic PVD cover image pixels are grouped into non-overlapping blocks of 2 bits and then the pixel difference is calculated. A range table is also used depending on the image pixel density. Based on the pixel difference and range table variable number of secret bits are embedded into the corresponding pixel block [8]. Authors in [18] have used RGB color image instead of an 8-bit grayscale image and solved the boundary problem. Swain in [19] used 3X3 pixel blocks and 9 pixel differences to hide secret bits. He has also solved the *Fall in Error Problem* (FIEP). The same author in [20] used 2X2 pixel blocks and 4 pixel differences and then embed variable number of secret bits using LSB substitution. Optimal Pixel Adjustment Process (OPAP) and PVD have been improved upon using optimal LSB substitution and authors proposed OOPAP and OPVD in [30].

Swain has used the quotient value differencing technique in combination with LSB to enhance security with a high capacity [22]. Many evolutionary computing algorithms have also been used to minimize image distortion while finding the near optimal secret bit placement. Cuckoo search and chaotic map have been used in [27] for generating optimal stego-key and encrypting secret bits correspondingly. Genetic Algorithm has been employed for the same purpose in [28]. PSO based optimization is used along with PVD and LSB in [29].

Edge areas in the cover image have abrupt changes in intensity values among adjacent pixels. This fact has been used in edge-based LSB steganography techniques to hide more secret bits without causing any visual distortion [23,24,25]. Generally, edge detection filters are used to identify the edge areas and then those pixels are targeted for data hiding. Laplacian of Gaussian (LoG) edge detector has

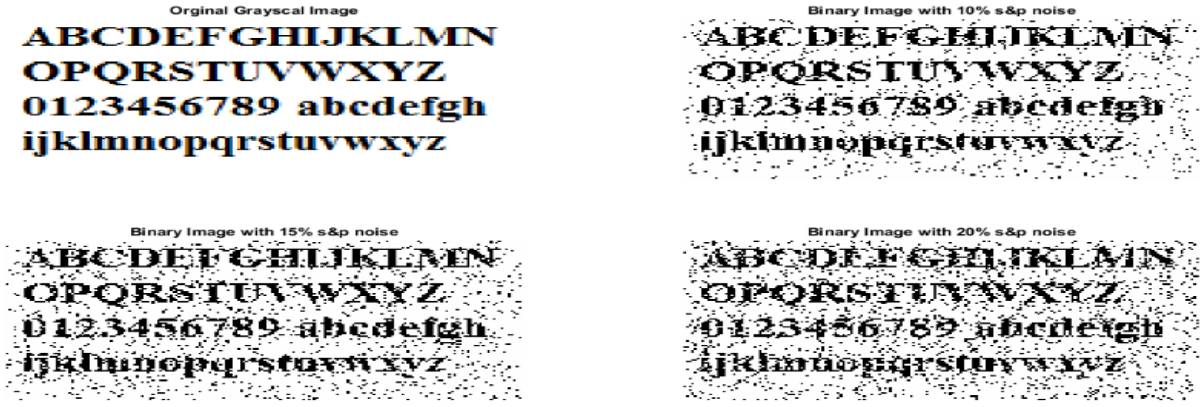


Fig. 2: (From top left clockwise) Secret images with (a) Printed Alphabets on a canvas in gray scale (b) after adding 10% salt and pepper noise and binarization (c) after adding 15% salt and pepper noise and binarization (d) after adding 20% salt and pepper noise and binarization

been used in [23].

Das et al. [10] have used a 24-bit RGB image as cover and used 3 grayscale images as the secret to hide. They have used Arnold transform to encrypt the images using 3 different keys before embedding the encrypted secret image bits into the Red, Green and Blue channels 3 LSB bits randomly. They have achieved high capacity as each cover image pixel is used to hide 3 secret image bits. The limitations of the technique lie in the fact that the stego-key size becomes large as 3 different large keys are required for Arnold transform and the same has to be communicated to the receiver to enable him to extract and decrypt the secret images. Also, this scheme requires the use of the original unaltered cover image during the extraction phase. If unaltered cover images are also sent along with the stego image then it will certainly make any adversary suspicious. If instead any standard image has been used as the cover image then the attacker might compare the stego with the cover and security could be jeopardized. Akhtar et al. [11] employed the *RC4* algorithm (*Rivest Cipher*) with shared stego-key to randomize the cover image pixels that are used to embed the secret image bits. Authors in [12] proposed a bit inversion before embedding the secret message to minimize the cover image distortion. Researchers in [13] used modulo based operation and range table to encode secret data bits before embedding. Dutta et al. [14] have employed modulo arithmetic for a multi-bit robust image steganography. Binary addition has been used in [15] to randomize the secret bits to improve security. Banik et al. [16] used *Bit Plane Complexity Segmentation (BPCS)* analysis and QR Decomposition of linear algebra to identify the area to embed the secret bits. Maji et al. in [35] used two images during encoding and decoding phases. A reference image has been used in combination with the secret text bits to encode them and then hidden in the cover image LSBs. The limitation of the scheme is that it requires to send both the images to the receiver with a pre-shared stego-key.

From the literature survey, we observe that one common limitation with LSB based data hiding schemes is the absolute sensitivity to the noise and image processing. As random noise disturbs the cover image LSBs and hence the secret becomes corrupted and unrecoverable. Another common issue is relying on the obscurity of the scheme believing that an attacker does not have the encoding algorithm. But for real security one should assume that the attacker has the encoding and embedding algorithms but even then, the security of the system should hold. The different parameter settings that constitute the stego-key should belong from a very large search space to provide safety against brute-forcing. In this paper we propose to use images from standard databases as a reference image and only the reference image id will be communicated to the recipient as a part of the stego-key with a few other parameters. One or more selected LSB bit-plane(s) will be used as an encryption-key for enhanced security and secret text will be printed on an image to increase robustness against random noise.

III. PROPOSED TECHNIQUE

The proposed technique uses the ability of human visual understanding to derive/identify known objects from partial and incomplete objects. Humans could easily identify a printed alphabet which is partially damaged and even some part of the item missing. As an illustration, consider the fig. 2(a) where the English alphabets are printed on a white canvas and then converted into a grayscale image. Now we have added random noises using salt and pepper noise to the image.

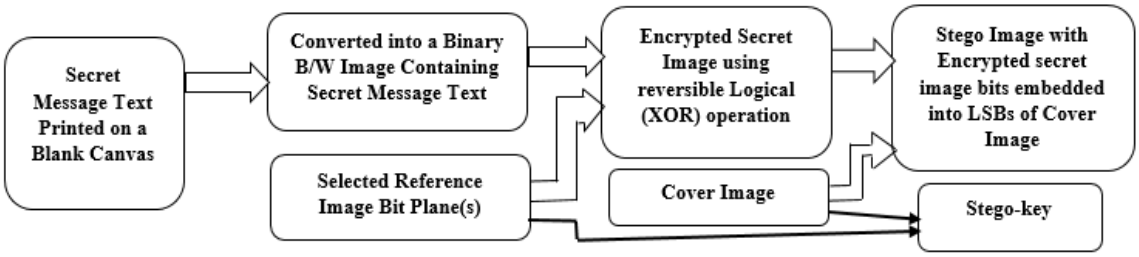


Fig. 4: Secret Message Encoding and Embedding Process at the sending side

In fig. 2(b), 2(c) and 2(d) a noise of 10%, 15%, and 20% added respectively to the original image and then converted to a black & white binary image. As we can observe that even with the addition of a considerable amount of noise most part of the characters is easily readable by humans. The proposed scheme utilizes this human ability along with a reference image bit plane-based encryption key. The basic block diagram of the stego-image generation from the cover image is shown in Fig. 3 and the secret message extraction at the receiver side are depicted in Fig. 4.

A. Stego-Image Generation: Encrypt and Embed

Step 1: Print the secret text message on a white canvas and convert into a Binary BW image to be treated as the secret image for hiding and transmission.

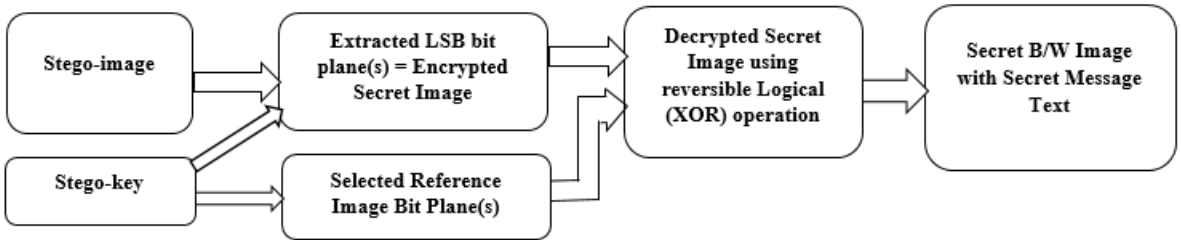
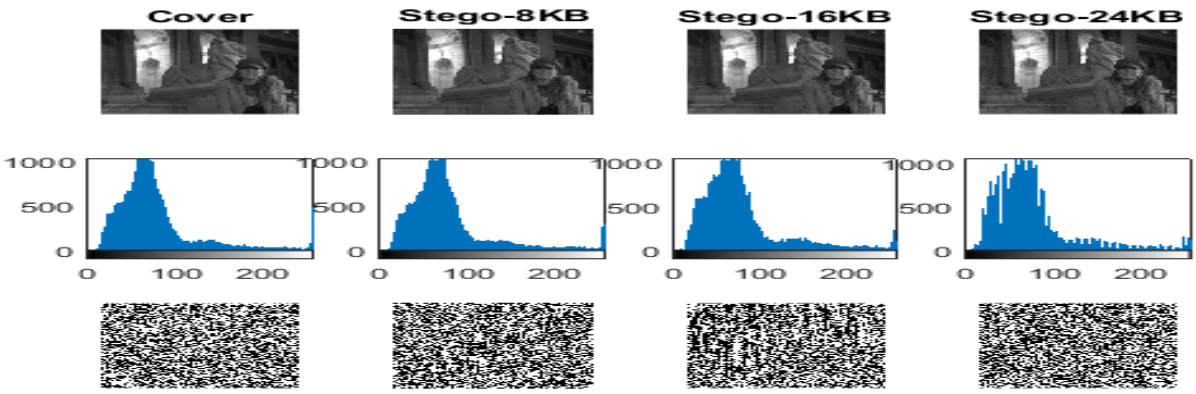


Fig. 3: Secret Message Extraction and Decoding Process at the recipient side



Step 2: Select a reference image from some well-known image database. Randomly select one or more LSB bit-planes as the encryption key

Step 3: Create stego-key containing the following: Reference image id (we assume both parties knows about the used image database), used LSB bit-plane(s), used image dimension (we assume same image dimension for the reference image, secret image, and cover image)

Step 4: Binary Secret image (BS) is a binary matrix of size m×m (assuming all image dimension as m×m), also any bit-plane of the reference image (BR) is also a binary matrix of the same size. Encrypt secret image bits as encrypted secret bits, EB = BS XOR BR.

Step 7: Stego-key is transmitted separately using private-key public-key cryptography (briefly discussed in sec 4.3)

Step 8: Transmit the stego image

A. Secret message Extraction and Decoding

Step 1: Decrypt and read parameter settings from stego-key that is received separately and decrypted using public key cryptography.

Step 2: From the stego key get the reference image using id from a pre-decided public image database. Image dimension is also retrieved from the stego-key file.

Step 3: Get the reference image bit-plane(s) used as BR, which will a binary matrix of size m×m.

Fig. 5: Stego-key encryption, transmission, and decryption using public key cryptography

Fig. 6:(Left) Cameraman, (right) Areal, size 256×256, 8-bit grayscale used as reference images in experiments

Step 5: Select any arbitrary personal image as cover image (so that no one could check differences between original and stego, though it's not essential)

Step 6: LSB bit plane(s) of the cover image (8-bit grayscale) is substituted by EB. If more than one secret image is used then higher LSB bit-planes of the cover image are also substituted.

Step 4: Extract the stego image LSB bit-plane(s) as SB which are actually the encrypted secret bits (EB)

Step 5: Decrypt the secret bits using reference image it-plane(s) to get the actual secret binary image as BS = SB XOR BR

Step 6: Display BS as a binary image and read the secret text

B. Stego-key Transmission

Different parameter setting used during data hiding at the sender side is written to a text file and then encrypted using public key cryptography. The basic process is quite simple. In this scheme both sender and receiver generate a public key-private key pair. The public key is known to all and the corresponding private key is kept secret and known to only the user who generated it. Any plain text encrypted using the public key of a user could be decrypted only by the corresponding private key. The sender first encrypts the stego-key using the public key of the receiver and then sends across. Now upon receiving the ciphertext, the receiver uses his own private key for decryption to obtain the stego-key. The process is depicted pictorially in fig 5.

IV. Experiment and Simulation

A. Experimental Setup

We have used Matlab R2015a for implementing the proposed scheme. We have used "cameraman.tif" and "areal.tif" with dimension 256×256 as the reference image in all experiments as shown in figure 6. Many different cover images have been used from SIPI image database for replicability of results otherwise it is better for the scheme to use personal fresh
images as cover. Binary images of same dimension containing secret text have been embedded into the cover images. We have converted any RGB image to 8-bit grayscale and resized to 256×256 before using. "Cameraman" has been used as a reference image for embedding a binary image with secret text printed on it and "Areal" has been used as reference image while embedding secret text message directly into cover images. We have used 256×256 sized blank white canvas and printed different amount of text to create the secret binary images. As shown in Fig. 7 (left) the secret binary image with 1KB secret text and in Fig 8(left) 0.5KB text has been printed on the same sized canvas.

B. Evaluation measures

1) Hiding Capacity

In image steganography, literature data hiding capacity is commonly defined as the maximum number of secret bits that can be hidden inside the cover media. In such a case it is Table 2: Secret text is directly embedded into the LSB bit-planes of the cover image.

Sl. No	Cover Image (256 256)	With payload as 8 KB text hidden into LSB bit-plane				With payload as 16 KB text hidden into 2 LSB bit-planes				With payload as 24 KB text hidden into 3 LSB bit-planes			
		PSNR	MSE	SSIM	BPP	PSNR	MSE	SSIM	BPP	PSNR	MSE	SSIM	BPP
1	Mandi	54.18	0.250	0.9965	1	47.44	1.17	0.9822	2	40.98	5.19	0.9333	3
2	peeper	54.12	0.250	0.9968	1	48.13	0.99	0.9850	2	41.07	5.08	0.9402	3
3	baboon	54.17	0.250	0.9987	1	47.08	1.27	0.9936	2	40.75	5.48	0.9750	3
4	boat	54.05	0.260	0.9974	1	47.18	1.25	0.9874	2	41.09	5.06	0.9524	3
5	couple	54.20	0.247	0.9950	1	47.45	1.17	0.9838	2	40.79	5.42	0.9212	3
6	house	54.36	0.238	0.9973	1	47.04	1.29	0.9909	2	40.67	5.58	0.9551	3
7	bridge	54.11	0.253	0.9989	1	47.50	1.16	0.9947	2	40.87	5.32	0.9815	3
8	Female	54.40	0.236	0.9961	1	47.57	1.14	0.9798	2	41.40	4.72	0.9304	3
9	Sailboat	54.14	0.251	0.9975	1	48.14	1.00	0.9878	2	40.89	5.30	0.9539	3
10	tree	54.34	0.240	0.9975	1	47.17	1.25	0.9885	2	41.03	5.14	0.9575	3

Table 1: Each secret binary image dimension is the same as that of the cover image i.e. 256×256. 1KB text is printed on each binary image. When a single binary image is hidden into the cover image then only LSB bit plane is substituted by the encrypted

Sl. No	Cover Image (256×256)	Payload as single Binary Image hidden in LSB bit-plane			Payload as 2 Binary Images hidden in 2 LSB bit-planes			Payload as 3 Binary Images hidden in 3 LSB bit-planes		
		PSNR	MSE	SSIM	PSNR	MSE	SSIM	PSNR	MSE	SSIM
1	Mandi	47.31	1.21	0.9969	46.92	1.32	0.9851	40.90	5.29	0.9353
2	peeper	46.87	1.34	0.9968	47.18	1.25	0.9847	41.02	5.14	0.9497
3	baboon	47.65	1.12	0.9989	47.10	1.27	0.9938	41.31	4.82	0.9746
4	boat	46.75	1.38	0.9977	47.28	1.22	0.9875	41.39	4.72	0.9531
5	couple	48.23	0.98	0.9946	47.45	1.17	0.9849	40.43	5.89	0.9239
6	house	47.44	1.17	0.9977	46.55	1.44	0.9916	39.89	6.67	0.9591
7	bridge	47.11	1.27	0.9989	45.76	1.73	0.9950	41.31	4.82	0.9844
8	Female	46.28	1.53	0.9962	45.48	1.84	0.9820	40.63	5.63	0.9296
9	Sailboat	45.61	1.79	0.9976	47.14	1.25	0.9881	40.85	5.34	0.9537
10	tree	46.94	1.32	0.9975	47.74	1.09	0.9884	39.71	6.96	0.9608

expressed in absolute terms i.e in number of Bytes. Many researchers use the embedding rate to measure the same where the percentage of maximum capacity is used. It is calculated as in equation (1).

$$P = \frac{\text{size of secret message embedded}}{\text{maximum size of secret that can be embedded}} \quad \text{where } 0 \leq P \leq 1 \tag{1}$$

Sometimes it is also expressed as bits per pixel (BPP) which is nothing but the ratio of the total number of secret bits to the total number of pixels in the cover image.

1) MSE and PSNR

Mean square error (MSE) of the stego image is the average squared difference between stego pixels and original cover image pixels using equation (2)

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[CI(i,j) - SI(i,j)]^2 \tag{2}$$

Where:

CI and *SI* are the original cover image and stego image respectively with image dimension as m pixel × n pixel i.e. both the images have m rows and n columns.

Peak Signal to Noise Ratio (PSNR) is the most commonly used metric for evaluation of any steganography. It is the ratio between maximum signal possible and the influence of modifying noise to the fidelity of its representation. It is calculated using MSE using equation (3).

$$\text{PSNR} = 10 * \log_{10}\frac{MAXi^2}{MSE} \tag{3}$$

Where:

MAXi is maximum fluctuation in the pixel value of an image I. For an 8-bit grayscale image MAXi =255.

A higher value of PSNR indicates that the reconstruction of the stego image is of better quality. PSNR is expressed in decibel (dB).

2) Structural Similarity Index (SSIM)

This method is normally employed to quantify the similarity between two given images of the same dimension. This index measures the quality of the distorted stego image if the original cover image is considered as the reference image and of perfect quality. It is an improved version of the universal image quality index (QI) proposed by the same group of researchers [37]. The Mathematical expression to calculate this index is given below [37]:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3}$$

Where X and Y are the cover image and stego image. μ_x, μ_y are arithmetic mean and σ_x ,σ_y represents the standard deviation between X and Y. C_1,C_2 are constants defined as : $C_1 = (K_1L)^2$; $C_2 = (K_2L)^2$, where $K_1 = 0{:}01$; $K_2 = 0{:}03$ and $L = 2^{\#\text{bitsperpixel}} - 1$.

3) Histogram

Pixel intensity versus frequency plot of an image is known as a histogram plot. It shows how many pixels are there and of which intensity. It helps to identify if the image is on the darker side or brighter side. Once LSB bits are modified to embed the secret bits, the number of pixels with a particular intensity changes and hence histogram pattern also changes. This is often used to detect the existence of steganography visually. If suspected then other statistical analysis follows. So, it is better to keep histogram differences to a minimum to make the scheme robust towards histogram- based attacks.

4) LSB bit-plane

If all least significant bits (LSBs) of the grayscale cover image are taken together and plotted as a binary image, it is known as LSB bit-plane. Similarly, for all 8 bits of each pixel could be plotted to get bit-plane plots for each higher LSBs. In case of

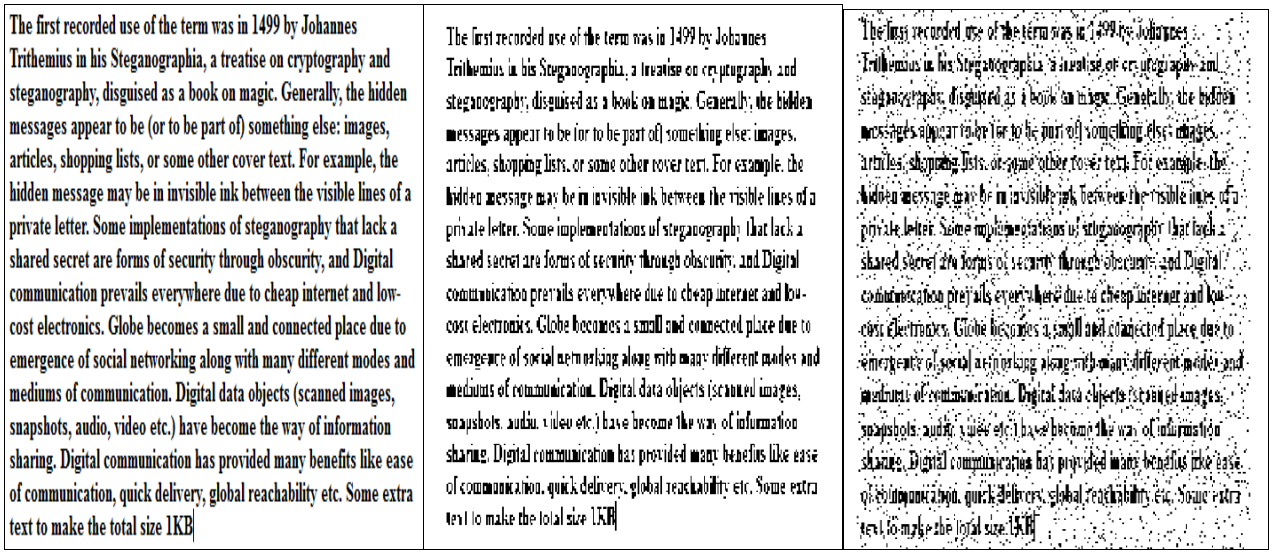


Fig. 7: (Left) The secret image (256×256) with 1KB text printed on it; (middle) Extracted Binary image (256×256) with 1KB secret text, and (right) Extracted binary image with 10% salt & peeper noise added to the stego-image

an RGB image, there exist 3 LSB bit-planes for each color channel. In a natural image, it is observed that the LSB bit-plane consists of random noise and hence looks complete random when visualized as an image. When some information is hidden in the LSBs then their randomness goes away and some patterns could be seen which indicates the non-natural image and this phenomenon is used to detect LSB based steganography. Many techniques nowadays, randomize the secret bits to evade such detection.

C. Results

In the first experiment, we have used a secret image with 1KB text printed on it as shown in Fig. 7(left) and embedded inside a cover image of the same size after encrypting it using the chosen reference image. The extracted secret image has been shown in Fig. 7(middle). Then we add 10% 'salt and peeper' noise to the stego-image and then extracted the secret image as shown in Fig. 7(right). We observe that with the added noise it becomes tough to recover the secret, though without noise the secret image is fully readable without much difficulty.

Next, we have reduced the amount of secret text and printed 0.5KB text (with larger font and boldface) in a 256×256 sized canvas and then stego-image is generated with the proposed scheme. The binary image with 0.5KB secret text and extracted binary image from the stego-image are shown in Fig. 8(left) and Fig. 8(middle). Finally, we have added 10% 'salt and peeper' noise to the stego image and then again extract the secret image as shown in Fig. 8(right). We observe that with a reduced amount of text, even after adding noise to the stego-image, the secret text is fully readable. We have also embedded 1KB text using the proposed scheme and then added noise to the stego. When tried to extract the secret text we observe that secret text is completely damaged and unrecognizable. We conclude that

the proposed scheme with secret text printed on an image is more robust against random salt and pepper noise.

In the second experiment, we have hidden secret text directly into the cover images. We have considered 8KB,16KB and 24KB text for hiding inside a 256×256 cover image. When 8KB text is embedded only the LSB i.e. 8^{th} bit of pixel intensities is used. Similarly, 2 LSB bits are employed with 16KB text and 3 LSBs used with 24KB.

Table 1 depicts the MSE and PSNR values with 10 different cover images and a reference image (cameraman) with different amount of secret text embedded inside the secret image. Here also 1KB text printed on a 256×256 secret image is hidden in the 8^{th} bit-plane of cover image, when capacity is increased to 2KB, then two secret images of size 256×256, each containing 1KB text are embedded into 2 LSB bit-planes of cover image and similarly, when capacity becomes 3KB, then 3 secret images are used to hide in 3 LSB bit-planes i.e. 6^{th},7^{th} and 8^{th} bit-planes of the cover image. Table 2 shows the MSE and PSNR values with 10 cover images and a reference image (areal) with different amount o

f text directly embedded using proposed scheme i.e. pixel used to embed.

In both the cases i.e. secret as an image and secret as a text directly embedded, we have shown histograms and LSB bit-plane plots for 3 selected cover images. In Fig 9(a), Fig. 9(b) and Fig. 9(c) shows the original cover image and stego-images with the embedded secret text of different size, histograms of them and LSB bit-plane plots. It is apparent that histograms are almost similar and LSB bit-planes are almost random with 1KB and 2KB text and in some cases (Fig. 9(b)baboon with 24KB text) LSB bit-plane plot shows some pattern indicative of possible data hiding. In Fig 10(a), Fig. 10(b) and Fig. 10(c) show same cover Mandi, Baboon and Couple and stego images with embedded images with printed secret text of size 1KB,2KB, and 3KB. Histograms and LSB bit-planes are plotted. Here we observe almost random LSB bit-plane plots and very minimal changes in histograms

It is seen that if the proposed scheme is used with secret text then hiding capacity becomes 8 times more than when secret text is printed on an image to be used for hiding. But directly embedding the secret text lowers the robustness against random noise that may be introduced during transmission. We have tested by adding 10% salt and pepper noise to the stego image after embedding text directly. After extraction following the scheme, we observe that the whole secret text is damaged and unrecognizable. Again, the same scheme when used with secret binary images containing the secret text printed on them, even though capacity reduces but protects information against random noise. Fig 8 establishes the robustness of the scheme when used with secret images against random noises.

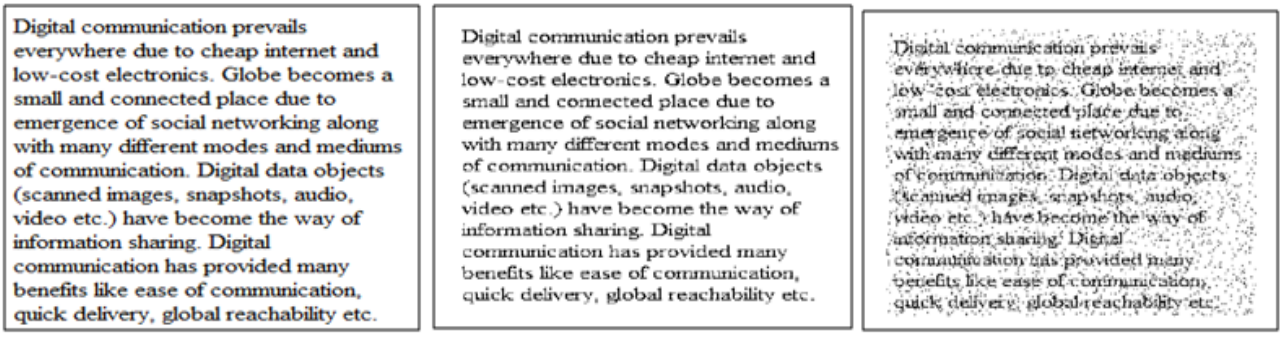


Fig. 8: (Left) Secret image (256×256) with 0.5KB secret text, (middle) Extracted image without adding noise to stego-image and, (right) Extracted image with 10% salt & peeper noise added to the stego-image

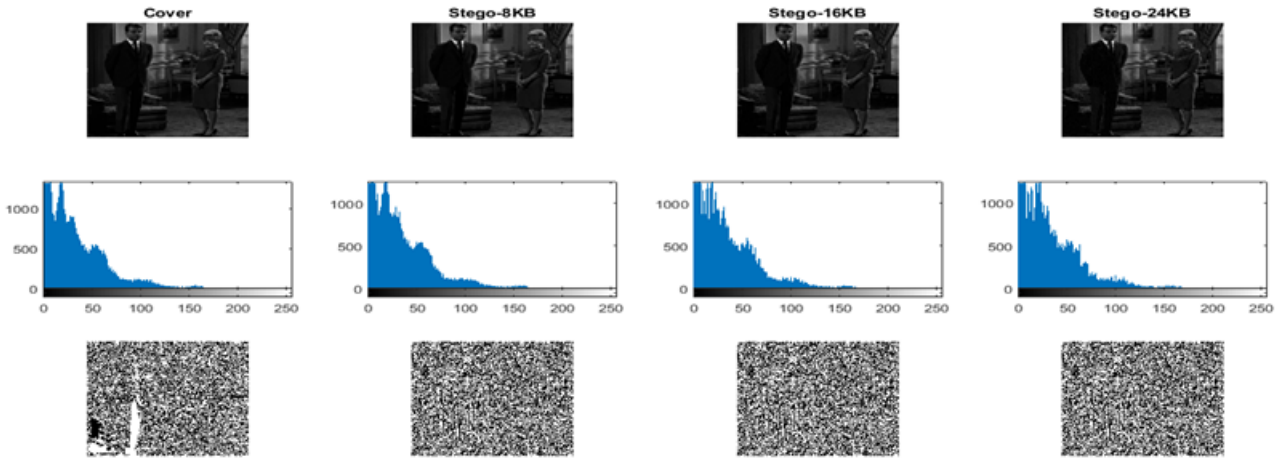


Fig. 9(c): Cover image: Couple and reference image: Cameraman, top row shows the visual imperceptibility, middle row depicts histograms and lower row plots the LSB bit-plane of (1st column) Original Cover, and Stego with embedded text of capacity (2nd column) 8KB (3rd column) 16KB and (4th column) 24KB respectively

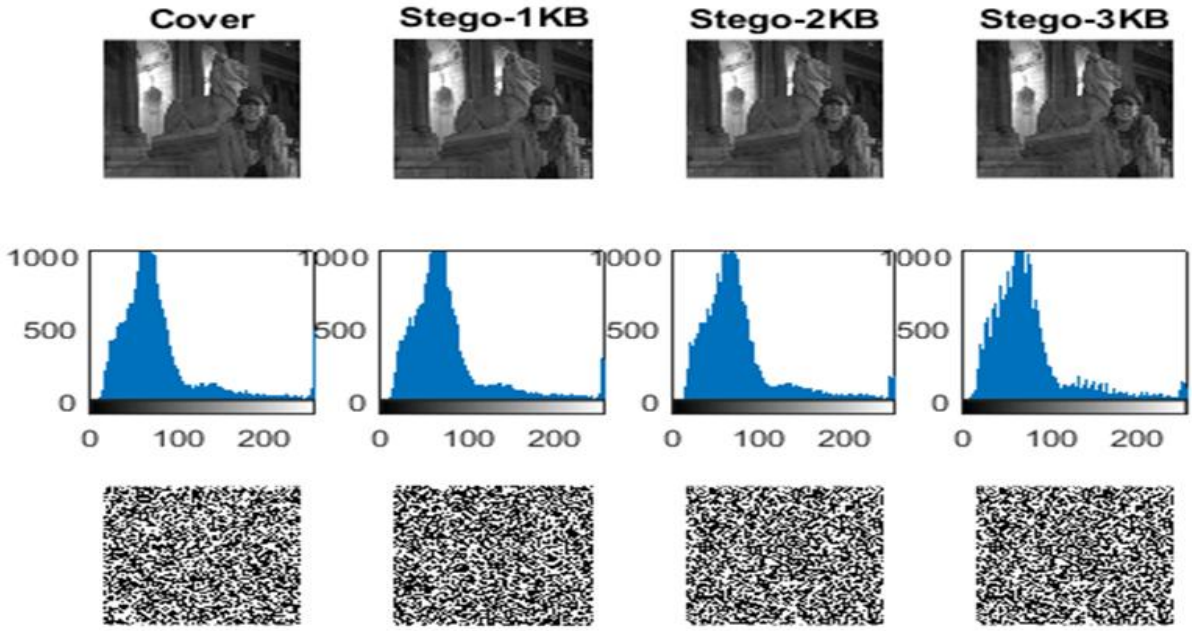


Fig. 10(a): Cover image: Mandi and reference image: Areal, top row shows the visual imperceptibility, middle row depicts histograms and lower row plots the LSB bit-plane of (1st column) Original Cover, and Stego with hidden binary image containing (2nd column) 1KB text (3rd column) 2KB Text and (4th column) 3KB Text respectively

V. CONCLUSION

In this paper, a new robust steganography scheme has been proposed that uses a reference image as an encryption key. This scheme encrypts the secret bits using randomly selected reference image bit-plane bits and then
embeds into cover image LSBs. Reference images are taken from open image databases and only the image id is communicated along with other steganography parameters in a stego-key file using public key cryptography. This scheme becomes secure due to encryption using random reference image bits. It is further made secure using different parameter setting every time which is communicated using stego-key secured using public key cryptography. Image quality metrics such as PSNR and MSE values show very negligible changes due to hidden information. Two application of the scheme has been evaluated. First, using an image containing secret text printed on it and second, directly embedding the
secret text binary into the cover image. In the first scenario amount of text embedded is less but it is robust against random noise. As results suggest even with 10% salt and pepper noise added to the stego image and then extracted the hidden image, secret text remain recognizable. But when direct text is embedded, capacity is much more but adding random noise to the stego-image completely damages the secret information. We have also seen that as we keep on increasing the amount of text printed on the secret image, it becomes more and more difficult to recognize. Hence the proposed scheme offers robustness with lower capacity and more capacity without noise
consideration.

REFERENCES

1. Provos, N., & Honeyman, P. (2003). Hide and seek: An introduction to steganography. IEEE security & privacy, 99(3), 32-44.
2. Cheddad, A., Condell, J., Curran, K., & Mc Kevitt, P. (2010). Digital image steganography: Survey and analysis of current methods. Signal processing, 90(3), 727-752.
3. Subhedar, M. S., & Mankar, V. H. (2014). Current status and key issues in image steganography: A survey. Computer science review, 13, 95-113.
4. Chandramouli, R., & Memon, N. (2001). Analysis of LSB based image steganography techniques. In Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205) (Vol. 3, pp. 1019-1022). IEEE.
5. Lin, E. T., & Delp, E. J. (1999, October). A review of fragile image watermarks. In Proceedings of the Multimedia and Security Workshop (ACM Multimedia'99) Multimedia Contents (Vol. 1, pp. 25-29).
6. Chang, C. C., Lin, C. C., Tseng, C. S., & Tai, W. L. (2007). Reversible hiding in DCT-based compressed images. Information Sciences, 177(13), 2768-2786.
7. Tirkel, A. Z., Rankin, G. A., Van Schyndel, R. M., Ho, W. J., Mee, N. R. A., & Osborne, C. F. (1993). Electronic watermark. Digital Image Computing, Technology and Applications (DICTA'93), 666-673.
8. Wu, D. C., & Tsai, W. H. (2003). A steganographic method for images by pixel-value differencing. Pattern Recognition Letters, 24(9-10), 1613-1626.
9. Potdar, V. M., & Chang, E. (2004, June). Grey level modification steganography for secret communication. In 2nd IEEE International Conference on Industrial Informatics, 2004. INDIN'04. 2004 (pp. 223-228). IEEE.
10. Das, P., Kushwaha, S. C., & Chakraborty, M. (2015, February). Multiple embedding secret key image steganography using LSB substitution and Arnold Transform. In 2015 2nd International Conference on Electronics and Communication Systems (ICECS) (pp. 845-849). IEEE.
11. Akhtar, N., Johri, P., & Khan, S. (2013, September). Enhancing the security and quality of LSB based image steganography. In 2013 5th International Conference and Computational Intelligence and Communication Networks (pp. 385-390). IEEE.
12. Akhtar, N., Khan, S., & Johri, P. (2014, February). An improved inverted LSB image steganography. In 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT) (pp. 749-755). IEEE.
13. Akhtar, N., Bano, A., & Islam, F. (2014, April). An improved module based substitution steganography method. In 2014 Fourth International Conference on Communication Systems and Network Technologies (pp. 695-699). IEEE.
14. Datta, B., Roy, S., Roy, S., & Bandyopadhyay, S. K. (2019). Multi-bit robust image steganography based on modular arithmetic. Multimedia Tools and Applications, 78(2), 1511-1546.
15. Datta, B., Mukherjee, U., & Bandyopadhyay, S. K. (2016). LSB Layer Independent Robust Steganography using Binary Addition. Procedia Computer Science, 85, 425-432.
16. Banik, B. G., & Bandyopadhyay, S. K. (2017). Image Steganography using BitPlane complexity segmentation and hessenberg QR method. In Proceedings of the First International Conference on Intelligent Computing and Communication (pp. 623-633). Springer, Singapore.
17. Sahu, A. K., & Swain, G. (2016). A review on LSB substitution and PVD based image steganography techniques. Indonesian Journal of Electrical Engineering and Computer Science, 2(3), 712-719.
18. Mandal, J. K., & Das, D. (2012). Colour image steganography based on pixel value differencing in spatial domain. International journal of information sciences and techniques, 2(4).
19. Swain, G. (2014). Digital image steganography using nine-pixel differencing and modified LSB substitution. Indian Journal of Science and Technology, 7(9), 1444-1450.
20. Swain, G. (2016). A steganographic method combining LSB substitution and PVD in a block. Procedia Computer Science, 85, 39-44.
21. Jung, K. H. (2018). Data hiding scheme improving embedding capacity using mixed PVD and LSB on bit plane. Journal of Real-Time Image Processing, 14(1), 127-136.
22. Swain, G. (2019). Very high capacity image steganography technique using quotient value differencing and LSB substitution. Arabian Journal for Science and Engineering, 44(4), 2995-3004.
23. Ghosal, S. K., Mandal, J. K., & Sarkar, R. (2018). High payload image steganography based on Laplacian of Gaussian (LoG) edge detector. Multimedia Tools and Applications, 77(23), 30403-30418.
24. Luo, W., Huang, F., & Huang, J. (2010). Edge adaptive image steganography based on LSB matching revisited. IEEE Transactions on information forensics and security, 5(2), 201-214.
25. Yang, C. H., Weng, C. Y., Wang, S. J., & Sun, H. M. (2008). Adaptive data hiding in edge areas of images with spatial LSB domain systems. IEEE Transactions on Information Forensics and Security, 3(3), 488-497.
26. Hussain, M., Wahab, A. W. A., Idris, Y. I. B., Ho, A. T., & Jung, K. H. (2018). Image steganography in spatial domain: A survey. Signal Processing: Image Communication, 65, 46-66.
27. Walia, G. S., Makhija, S., Singh, K., & Sharma, K. (2018). Robust stego-key directed LSB substitution scheme based upon cuckoo search and chaotic map. Optik, 170, 106-124.
28. Mandal, J. K., & Khamrui, A. (2014). A genetic-algorithm-based steganography on colour images (GASCI). International Journal of Signal and Imaging Systems Engineering, 7(1), 59-63.
29. Li, Z., & He, Y. (2018). Steganography with pixel-value differencing and modulus function based on PSO. Journal of information security and applications, 43, 47-52.
30. Ansari, E., Keshtkaran, M., Wallace, R., Mirsadeghi, S. M. H., & Ansari, F. (2019). OOPAP and OPVD: Two Innovative Improvements for Hiding Secret Data Into Images. Iranian Journal of Science and Technology, Transactions of Electrical Engineering, 43(1), 55-65.
31. Alturki, F., & Mersereau, R. (2001, October). Secure blind image steganographic technique using discrete fourier transformation. In Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205) (Vol. 2, pp. 542-545). IEEE.
32. Liu, T., & Qiu, Z. D. (2002, August). A DWT-based color image steganography scheme. In 6th International Conference on Signal Processing, 2002. (Vol. 2, pp. 1568-1571). IEEE.
33. Hong, W., & Chen, T. S. (2012). A novel data embedding method using adaptive pixel pair matching. IEEE transactions on information forensics and security, 7(1), 176-184.
34. Bhattacharyya, S., & Sanyal, G. (2010, July). Hiding Data in Images Using Pixel Mapping Method (PMM). In security and Management (pp. 683-689).
35. Maji, G., Mandal, S., Sen, S., & Debnath, N. C. (2018, January). Dual image based LSB steganography. In 2018 2nd International Conference on Recent Advances in Signal Processing, Telecommunications & Computing (SigTelCom)(pp. 61-66). IEEE.
36. Wang, H., & Wang, S. (2004). Cyber warfare: steganography vs. steganalysis. Communications of the ACM, 47(10), 76-82.
37. Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, Apr. 2004.

AUTHORS PROFILE

Mr. Giridhar Maji, MIE is working as a Lecturer in Electrical Engineering Department, Asansol Polytechnic, Department of Technical Education and Training, West Bengal, India since 2014. He is an active Member of Institute of Engineers (India). He had worked in IT industry for more than 6 years in different capacities at Tata Consultancy Services and, Cognizant Technology Solutions. He has done his M. Tech from university of Calcutta and B. Tech from National Institute of Technology, Durgapur, India. He is currently working towards his PhD from University of Calcutta. He has published in more than 15 international journals and conferences. He has 4 book chapters to his credit. His area of research is data mining, data warehousing, social network analysis and information security/steganography.

Ms. Sharmistha Mandal is working as a Lecturer in Computer Science and Technology, Kanyapur Polytechnic, Department of Technical Education and Training, West Bengal, India since 2017. She completed her B.Sc., B.Tech. and M. Tech from university of Calcutta, India in 2009, 2012 and 2015 respectively. She is currently pursuing PhD from Department of Computer Science & Engineering, University of Calcutta. She has published many articles in international journals and conferences. Her field of work constitutes data warehousing, steganography and cloud computing.