© Universiti Tun Hussein Onn Malaysia Publisher's Office

IJIE

Journal homepage: http://penerbit.uthm.edu.my/ojs/index.php/ijie
ISSN : 2229-838X e-ISSN : 2600-7916

The International Journal of Integrated Engineering

Reversible Data Hiding Technique using Novel Interpolation Technique and Discrete Cosine Transform

Basant Sah[1*], **Vijay Kumar Jha**[2]

[1]Research Scholar, Department of Information Technology
Birla Institute of Technology, Mesra, Ranchi, 835215, INDIA

[2]Associate Professor, Department of Computer Science and Engineering
Birla Institute of Technology, Mesra, Ranchi, 835215, INDIA

*Corresponding Author

DOI: https://doi.org/10.30880/ijie.00.00.0000.00.0000
Received 16 June 2018; Accepted 12 September 2018; Available online 30 April 2019

Abstract: The application of internet has expanded colossally in recent years. Subsequently, security of data has turned out to be a quite important factor to focus. It is likewise discernible that over the internet, data can be controlled, cautioned or compromised. To overcome these issues numerous methodologies are proposed. These methods depend on some type of mechanism, where data is encoded and with the help of a key message and using the key it can be recovered the receiver. In pictures, reversible data hiding (RDH) is a strategy which is exceptionally useful in regaining original cover after the embedded message is separated. This key strategy is comprehensively utilized as a piece of medical imagery, law crime scene investigation and military symbolism, where no distortion of the cover is allowed. Because of being reversible, RDH has pulled in great research interest. This paper discusses an improvement in RDH technique while also considering Discrete Cosine Transform (DCT) based compression technique for further improvements. Simulation has been carried out to evaluate the performance in terms of Peak-to-Signal Noise Ratio (PSNR) and Structural Similarity Index (SSIM). It has been found that our proposed method performs better than earlier methods.

Keywords: RDH, DCT, SSIM, PSNR

1. Introduction

In the practical point of view, various RDH systems have been created in recent years. [1] built up a general framework for RDH. With the underlying extraction of the compressible qualities of unique cover and after this packing them lossless, we can create space for installing auxiliary information [2]. In the initial studies of data hiding difference expansion (DE) method as proposed by [3], [4], and [5] is extensively used. Later on, histogram shift (HS) is another strategy in which space is conserved for inserting of information by modifying the histogram bins as proposed by [6]. L. Zhang and X. Wu [7] make the division of encrypted image into numerous blocks. With flipping 3 LSBs of the half of pixels in each block, room can be cleared for the inserted bit. The data extraction and picture recuperation proceed by finding which segment has been flipped in one block. This methodology can be recognized by methods for spatial relationship in decrypted picture. Hong et al. [8] enhanced Zhang's procedure at the decoder end the further misuse of the spatial correlation using a different estimation equation and side match strategy to achieve very lower fault rate. These two procedures indicated above rely upon spatial correlation of initial image to get information. In the

*Corresponding author:basantbitmtech2008@gmail.com other words, the encoded picture should be decoded first prior to the data extraction. Hong W [8] used interpolation method for embedding of data.

2. Discrete Cosine Transform (DCT) technology

One drawback of the DFT for a few applications is that the change is not simple esteemed, notwithstanding for genuine information. On the other hand, we do not have such issues with the discrete cosine transform (DCT). It is quite dissimilar from the DFT. It is broadly utilized as a part of applications dealing with image and video compression, e.g., Joint Photographic Experts Group (JPEG) and Moving Picture Experts Group (MPEG). It is additionally conceivable to utilize DCT for filtering by making use of a marginally extraordinary type of convolution known as symmetric convolution as discussed by [9]. One application of DCT is in images compression, due to the fact that it just holds most noteworthy frequency parts and dispose-off others. This could be defined as a lossy compression, where a portion of the excess information is left, although we have almost the same characteristics of the real image. However, the extent of compression depends on the number of non-zero components in the DCT matrix (detailed below). In this work, we have used SSSIM as a compression parameter and we compress the image till the SSSIM remains one, in such a case all the redundant information will be lost while still maintaining all the important information.
Let us consider that the data array has definite rectangular support on $[0, N-1]\times[0, N-1]$, at this point, we have the 2-D DCT as

$$X_C(k_1,k_2)=\begin{cases}\dfrac{C(k_1)C(k_2)}{\sqrt{2N}}\displaystyle\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1}x(n_1,n_2)\cos\frac{\pi k_1}{2N}(2n_1+1)\cos\frac{\pi k_2}{2N}(2n_2+1), \\ \text{for}(k_1,k_2)\in[0,N-1]\times[0,N-1] \\ 0 \qquad \text{otherwise}\end{cases} \tag{1}$$

$$C(v)=\begin{cases}\dfrac{1}{\sqrt{2}} & \text{if } v=0 \\ 1 & \text{if } v>0\end{cases} \tag{2}$$

Here, $x(n_1,n_2)$ is the pixel values in space domain. C(.) are coefficients and cosine terms represents transform. Note that unlike the DFT, where the highest frequencies occur near (N/2, N/2), the highest frequencies of the DCT occur at the highest indices $(\mathrm{k}_1,\mathrm{k}_2)=(0,0)$. DCT is less computationally complex as compared to DFT. It turns out that eigenvectors of the unitary DCT are the same as those of the symmetric tri-diagonal matrix,

$$\mathrm{Q}=\begin{bmatrix}1-\alpha & -\alpha & 0 & \cdots & \cdots & 0 \\ -\alpha & 1 & -\alpha & 0 & \cdots & 0 \\ 0 & -\alpha & 1 & -\alpha & \ddots & \vdots \\ \vdots & 0 & -\alpha & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 & -\alpha \\ 0 & \cdots & \cdots & 0 & -\alpha & 1-\alpha\end{bmatrix} \tag{3}$$

and this holds true for arbitrary values of the parameter α.

3. Additive Interpolation-Error Expansion

Additive interpolation error expansion is proposed by L. Luo [10]. In this method which interpolation technique is used, to estimate the interpolation values of pixels. This strategy works by speculating a surrounding pixel value. So, now we get the interpolation-errors (ε) through the equation

$$\varepsilon = y - y' \tag{4}$$

This is a powerful technique for data embedding, moreover, lesser the error more data embedding is possible. Lesser error mans better PSNR and SSIM and thus more space for data embedding.
The equation stated here, we have the parameter y' representing the interpolation values of pixels y. First of all, interpolation-errors histogram is plotted. Histogram plot is a method in which distribution of intensity is shown with respect to indexing. Let us consider that left peak (LP) and right peak (RP) define the two peak points of interpolation-errors histogram and we can define it as

$$\begin{cases} LP = \arg \max\limits_{\varepsilon \in E} hist(\varepsilon) \\ RP = \arg \max\limits_{\varepsilon \in E - \{LM\}} hist(\varepsilon) \end{cases} \qquad (5)$$

In the equation 5, the parameter *hist* (*ε*) is used to represents the event with interpolation-error is equal to *ε* and *E* is the interpolation-error set. Considering *LP*<*RP*. We can group the interpolation errors into the below given two classes. These are

1) Right interpolation-errors (*RE*): where interpolation error ε satisfies $\varepsilon \geq RP$.

2) Left interpolation-errors (*LE*): where interpolation-error ε satisfies $\varepsilon \leq LP$.

The equation given below defines additive interpolation-error expansion

$$\varepsilon' = \begin{cases} \varepsilon + sgn(\varepsilon) \times b, & \varepsilon = LP \ or \ RP \\ \varepsilon + sgn(\varepsilon) \times 1, & \varepsilon \in (LP, LM) \cup (RP, RM) \\ \varepsilon, & \text{else} \end{cases} \qquad (6)$$

In the above equation, ε' represent the expanded interpolation-error, *b* is the bit to be embedded, and *sgn*(.) is a signum function and defined as

$$sgn(\varepsilon) = \begin{cases} 1, & \varepsilon \in RE \\ -1, & \varepsilon \in LE \end{cases} \qquad (7)$$

The parameters left minimum (*LM*) and right minimum (*R*M) in the equation 6 could be defined as follows

$$\begin{cases} LM = \arg \min\limits_{e \in LE} hist(\varepsilon) \\ RM = \arg \min\limits_{e \in RE} hist(\varepsilon) \end{cases} \qquad (8)$$

By and large it has been discovered that LP is an integer of very little measure and more often than not it is 0, comparably LM is an integer with smaller values, and it does not have any interpolation-error, it fulfils ε= LM. Likewise, for the most of the times, RP is equivalent to 1 and RM is a bigger integer without any interpolation-error fulfilling ε= RM.

In the system of extraction, the similar interpolation calculation could be applied; it is conceivable to have the past interpolation values and y' the interpolation-errors with the help of the equation given below

$$y'' = y' + \varepsilon' \qquad (9)$$

While performing the extracting process, with the same methodology of interpolation, we can have similar interpolation values y' , and the corresponding interpolation-errors by means of

$$\varepsilon' = y'' - y' \qquad (10)$$

At the stage, we get the parameters *LP*, *LM*, *RP*, and *RM*, the embedded matter could be extracted by the equation given below

$$b = \begin{cases} 0, & \varepsilon' = LP \ or \ RP \\ 1, & \varepsilon' = LP - 1 \ or \ RP + 1 \end{cases} \qquad (11)$$

With the end goal of the recovery of real interpolation-errors, we can make use of the inverse function of additive interpolation-error expansion and hence we get,

$$\varepsilon=\begin{cases}\varepsilon'-sign\left(\varepsilon'\right)\times b, & \varepsilon'\in\left[LP-1,LP\right]\cup\left[RP,RP+1\right]\\ \varepsilon'-sign\left(\varepsilon'\right)\times 1, & \varepsilon'\in\left[LM,LP-1\right]\cup\left[RP+1,RM\right]\\ \varepsilon, & else\end{cases} \quad (12)$$

At last, the original pixels can be restored by

$$y=y'+\varepsilon \quad (13)$$

4. Reversible Data Hiding in Encrypted Images

Few years before, an RDH procedure was introduced in which before making the encryption, the room is kept reserved and which is observed to be superior to earlier works by [11]. In this segment, similar method is explained point by point and modifications are recommended to have some additional enhancement in the outcomes.

Generation of Encrypted Image

A RDH process comprises of three parts: image partition, second one is self-reversible embedding and the third one is picture encryption. At first, step of image segment makes the division of information picture into two segments A and B and; after this, the LSBs of A are reversibly inserted into B using RDH algorithm. At last, encrypt the modified picture to create its final form.

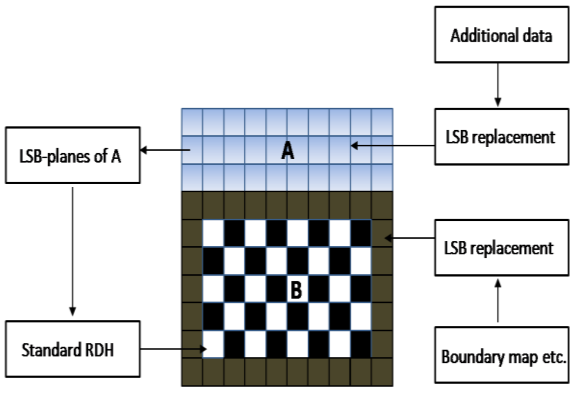


Fig. 1- Pictorial representation of image partition and data embedding process

The first step is of portioning the image. Assume a unique picture *C* and it is an 8 bits-grey scale image. Its measurement is M×N and with pixel esteems C_i, *j* ε [0,255]. As the initial step, the content owner divides the input image into overlapping blocks along the rows. The number of blocks depends on size of embedded messages, defined by '*em*'. Considering in a definite way, each block contains '*R*' rows, where $R=\lceil em/N\rceil$, and the total number of required overlapping blocks are $B=M-R+1$. A crucial perspective now is that all blocks are covered by before as well as sub-successive blocks with the rows. A function for each piece is required to be defined with the end goal of estimation of its first-order smoothness which is nothing but the difference between current pixels and average of surrounding pixels.

$$f=\sum_{u=2}^{R}\sum_{v=2}^{N-1}\left|P_{u,v}-\frac{P_{u-1,v}+P_{u+1,v}+P_{u,v-1}+P_{u,v+1}}{4}\right| \quad (14)$$

Here, the function *f* check the difference in the pixel values from its surrounding pixels in 0^0, 90^0, 180^0 and 270^0. This operation is performed for all the pixels expect boundary pixels. For complete smooth surface *f* is zero. Higher value of estimate *f* implies that the concerned block has complex textures. The content owner, consequently, picks the particular block with the highest *f* to be *A*, and embed it to the image front of the connected by the rest of the part *B* with smaller finished zones, as outlined in Figure 1.

Self-Reversible Embedding

In this method first image is divided into two portion A and B, and on these portion following operations are performed. First of LSB of A is inserted into plane B using RDH process as in below step 1:

STEP 1: LSB of A $\rightarrow$ Into B Using RDH
In partition B black and white pixels are defined as

$(i+j) \bmod 2 = 0$ White pixel

$(i+j) \bmod 2 = 1$ Black pixel

Here, all white pixels could be estimated, $B_{i,j}$ with the help of interpolation value gained through the four black pixels that surrounds it as illustrated in the below equation

$$B'_{i,j} = W_1 B_{i-1,j} + W_2 B_{i+1,j} + W_3 B_{i,j-1} + W_4 B_{i,j+1} \qquad (15)$$

In this equation, the parameter weights W_i , $1 \le i \le 4$ is estimated via the method defined in [10]. In the previous work, interpolation pixels at 0^0 and 90^0 are considered.

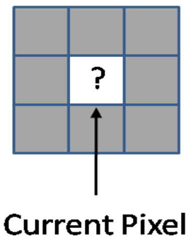


Fig. 2- Estimation of center pixel through surrounding pixels

Let us have a 3×3 part of an image as shown in Figure 2. The pixel value of this image changes from *P* (1, 1) to *P* (3, 3), at that point, we get the mean value with the help of the pixel values of the pixels at 0^0 and 90^0 respectively. Suppose we have to evaluate $y = P(2,2)$, with assessed value $\hat{y}$.
The mean value of surrounding pixel of *x*, at 0 and 90 degrees is

$$m_p = \frac{1}{4}\left[P(1,2) + P(2,1) + P(2,3) + P(3,2)\right] \qquad (16)$$

Defining average value in 0 and 90 degrees respectively as

$$x_0 = \frac{1}{2}\left[P(2,1) + P(2,3)\right] \text{ and } x_{90} = \frac{1}{2}\left[P(1,2) + P(3,2)\right].$$

Considering,

$$S_0 = [P(2,1), P(2,3), y_0] \text{ and } S_{90} = [P(1,2), P(3,2), y_{90}]$$

The variances in 0 and 90-degree directions are

$$\begin{cases} \sigma(e_0) = \frac{1}{3}\sum_{k=1}^{3}\left(S_0(k) - u\right)^2 \\ \sigma(e_{90}) = \frac{1}{3}\sum_{k=1}^{3}\left(S_{90}(k) - u\right)^2 \end{cases} \qquad (17)$$

The weight in 0 and 90-degree direction is given by

$$w_0 = \frac{\sigma_{90}}{\sigma_0 + \sigma_{90}} \text{ and } w_{90} = \frac{\sigma_0}{\sigma_0 + \sigma_{90}} \qquad (18)$$

The estimated value of the pixel is obtained by

$$\hat{y} = w_0 y_0 + w_{90} y_{90} \qquad (19)$$

The estimated error is

$$\varepsilon = y - \hat{y} \qquad (20)$$

For the most part, for any pixel value $B_{i,j}$ evaluating error is assessed with the help of $e_{i,j} = B_{i,j} - B'_{i,j}$ and while later a certain measure of data could be embedded into the estimating error structure with histogram shift. We will detail all this in the following sections. At that point, moving ahead, we can make the estimation of the black pixels could be done with the help of surrounding white pixels. These pixels might be already modified. Completing this, some other estimating error sequence which can facilitate accommodation to the messages as well is generated. Additionally, we can in the same manner execute multilayer embedding design with the thought of the altered B as "original" one at the point of process it is needed. At last, the result is with the end purpose to exploit each pixel of B, two having the measurement of error sequences are created for planting the messages in each of the single-layer embedding techniques.
On each error sequence we can add data with the help of Histogram shift. Make the division of the Histogram in two sections L and R and look for highest peaks in each part of *LP* and *RP*. Normally, *LP* = -1 and *RP* = 0. Next, we search for zero in each part and define as *LM*, *RM* respectively.
Keeping in mind the end goal to embed messages into places alongside an evaluating error which is equal to RM, move every error value amongst RP+1 and RM-1 with one stage to right, at that point, the bit 0 is represented by RP and the bit 1 with RP+1. The manner of embedding in the left part is same except for that the direction of moving is left, and the move is recognized by subtracting 1 from the relating pixel values.
If after sufficient level of embedding, some part of the message is still left, then a part of error sequence with enough peak points can be selected we call them proper points and denoted by LPP and RPP.

Image Encryption

After making the adjustment of the self-embedded image, denoted by the parameter Y, is created, we can influence the encryption of Y to develop to the encrypted image, demonstrated by E. With the assistance of a stream cipher, the encryption form of Y is viably obtained. To influence it all the more clear, let us have a gray value that ranges from 0 to 255 could be spoken to by methods for 8 bits, $Y_{i,j}(0)$, $Y_{i,j}(1)$, ..., $Y_{i,j}(7)$, ..., in a way that

$$Y_{i,j}(k) = \left\lfloor \frac{Y_{i,j}}{2^k} \right\rfloor \bmod 2, \qquad k = 0,1,\ldots,7. \qquad (21)$$

The encrypted bits $E_{i,j}(k)$ can be estimated by means of exclusive-or operation.

$$E_{i,j}(k) = Y_{i,j}(k) \oplus C_{i,j}(k) \qquad (22)$$

The parameter used in the above equation $C_{i,j}(k)$ is developed with the help of a standard stream cipher discovered by the encryption key.

Decrypted Image Generation

Keeping in mind the end goal to build up the marked decoded picture Y'' which is develop of A'' and B'', the owner of the content ought to take after the below given steps.

1. By making use of the encryption key, the owner completes the process of decrypting the image excluding the LSB-planes of A_E. Now, we can make the calculation of the decrypted form of E' containing the embedded data through

$$Y''_{i,j}(k) = E'_{i,j}(k) \oplus C_{i,j}(k) \qquad (23)$$

And

$$Y''_{i,j} = \sum_{k=0}^{7} Y''_{i,j}(k) \times 2^k, \qquad (24)$$

In the given equations, the parameters $E'_{i,j}(k)$ and $Y''_{i,j}(k)$ denotes the binary bits of $E'_{i,j}(k)$ and $Y''_{i,j}(k)$, gained by means of (eqn. 22) respectively.

2. Now, we should make the procedure of extraction of SR and ER in negligible zone of B''. With the rescheduling of A'' and B'' to its genuine frame, we obtain the plain picture that includes the embedded information. Due to

the fact that the marked decrypted image Y'' is undefined to changed Y except for LSB-planes of A. In the interim, it keeps up perceptual straightforwardness in contrast with the real image C.
Likewise, especially the distortion is suggested by methods for two distinct ways: the implanting strategy by altering the LSB-planes of A and self-reversible embedding methodology by inserting LSB planes of A into B. The beginning step distortion is very overseen by means of misusing the LSB-planes of only A and the following section can provide benefit from sensational execution of current RDH systems.

Image Restoration and Data Extraction

Presently, after the development of the marked decoded image, the owner of the content can also make the data extraction and influences the recovery of real image. The system is more or less similar to that of common RDH strategies [12] and [13].

The steps shown below explain the specific steps

1. Make the Recording and decryption of the LSB-planes of A'' as per the data hiding key; extract the data till reach the end label.

2. Using the marginal area LSB of B'' find *LM*, *RM*, *LP*, *RP*, *LPP*, *RPP*, R_b(data rate), *y* and boundary map. After this, scan B'' and take following steps.

3. If value of R_b is 0, this implies that none of the black pixels involved in embedding procedure move to Step 5.

4. Evaluate the estimating errors $\varepsilon'_{i,j}$ of the black pixels $B''_{i,j}$. If $1 \le B''_{i,j} \le 254$, makes the recovery of the original pixel value estimating error and in a reverse order and extract embedded bits when $\varepsilon'_{i,j}$ is equal to *LM*, *LP* (or *LPP*), *RP* (*RPP*) and *RM*. Else, if $B''_{i,j} \in \{0,255\}$, check for bit *b* in boundary map. If *b*=0, no information, else follow same procedure as for $B''_{i,j} \in [1,254]$. Make the repetition of this step up to the point till we extract the part of payload R_b. In the case of extracted bits are form marginal area and they are LSBs of pixels, restore them without making any delay.

5. Make the calculation of estimating errors $\varepsilon'_{i,j}$ of the white pixels $B''_{i,j}$, and extract embedded bits and carry out the recovery of white pixels in the similar manner as in Step 4. In the event of the extracted bits are LSBs of pixels in marginal zone, restore them at once.

6. Repeat Step 2 to Step, to 5, Y-1 times on B'' and merge each extracted bit to develop LSB-planes of A. This should be continued till we recover B perfectly.

7. Marked LSB-planes of A'' should be replaced with its original bits extracted from B'' in order to recover cover image C.

Proposed Interpolation Method
In the proposed interpolation method, we consider the all 8 pixels which surround central pixel in evaluating interpolated value. In evaluating mean all 8 pixels values are considered, in the proposed method, we have considered zonal values not any particular direction pixel values. For this let us have an image 3×3 part. The pixel value of this image changes from P (1, 1) to P (3, 3); at that point, we get the mean value with the help of the pixel values of the pixels in zone I and zone II respectively. The division of zones is shown in Figure 4. Suppose we have to evaluate $y = P(2,2)$, with assessed value $\hat{y}$.

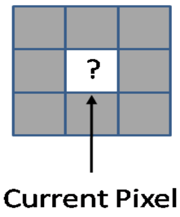


Fig. 3- Estimation of center pixel through surrounding pixels using proposed method

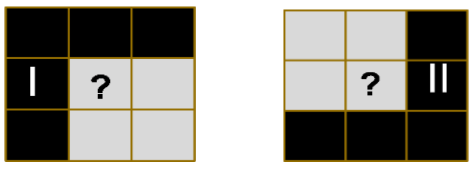


Fig. 4- Estimation of center pixel through surrounding pixels in the zone I and zone II

We estimate the mean value of surrounding pixel of y, at I and II degree as

$$m=\frac{1}{8}\begin{bmatrix} P(1,1)+P(1,2)+P(1,3) \\ +P(2,1)+P(2,3)+P(3,1)+P(3,2)+P(3,3) \end{bmatrix} \quad (22)$$

In the mean central pixel is also included once in I and other in II directions.
Defining average value in zone I and zone II respectively as

$$y_I=\frac{1}{5}\left[P(1,1)+P(1,2)+P(1,3)+P(2,1)+P(3,1)\right] \text{ and}$$

$$Y_{II}=\frac{1}{5}\left[P(1,3)+P(2,3)+P(3,1)+P(3,2)+P(3,3)\right]. \text{ Considering,}$$

$$S_I=[C(2,1),(2,3),y_I] \text{ and } S_{II}=[C(1,2),(3,2),y_{II}]$$

The variance in I and II direction is

$$\begin{cases} \sigma(e_I)=\frac{1}{3}\sum_{k=1}^{3}\left(S_I(k)-u\right)^2 \\ \sigma(e_{II})=\frac{1}{3}\sum_{k=1}^{3}\left(S_{II}(k)-u\right)^2 \end{cases} \quad (23)$$

The weight in zone I and zone II is given by

$$w_I=\frac{\sigma_{II}}{\sigma_I+\sigma_{II}} \text{ and } w_{II}=\frac{\sigma_I}{\sigma_I+\sigma_{II}} \quad (24)$$

The estimated value of the pixel is obtained by

$$\hat{y}=w_I y_I+w_{II} y_{II} \quad (25)$$

The further process is same as above.

Evaluation Parameters

PSNR could be defined as a measure of image quality and could be rewritten as

$$PSNR=10\log_{10}\frac{(255)^2}{MSE} \quad (26)$$

In the above equation, the parameter MSE is an estimate of pixel difference in real and watermarked image. SSIM is a method which measures the adjustment in luminance, contrast, and image structure. Average pixel intensity used to model luminance, variance between the reference and distorted image is used to model contrast; the cross-correlation between the two images is used for structural similarity.

$$SSIM(x,y)=\frac{\left(2\mu_x\mu_y+c_1\right)(2\sigma_{xy}+c_2)}{\left(\mu_x^2+\mu_y^2+c_1\right)\left(\sigma_x^2+\sigma_y^2+c_2\right)} \quad (27)$$

The parameters μ_x and σ_x^2 used in the above equation is the mean and variance of real image, while talking about μ_y and σ_y^2, these are the mean and variance of the distorted image and the parameter σ_{xy} is used to define cross co-variance between the two image. c_1 and c_2 are constants. The mean SSIM is given by

$$MSSIM=\frac{1}{T}\sum_{j=1}^{T}SSIM_j \quad (28)$$

5. Results

This chapter discusses the results obtained through simulation process. In the simulation six images ‘Airplane’, ‘Baboon’, ‘Barbara’, ‘Boat’, ‘Lenna’ and ‘Peppers’ in pgm format is considered. The file format known as “plain pgm”, which stands for “Portable Greymap”, is a text-based image format for greyscale images. Each image is of size 512×512 and of 257KB.

Airplane

(a) Original Image

(b) Watermarked Image

Baboon

(a) Original Image

(b) Watermarked Image

Barbara

(a) Original Image

(b) Watermarked Image

Boat

(a) Original Image

(b) Watermarked Image

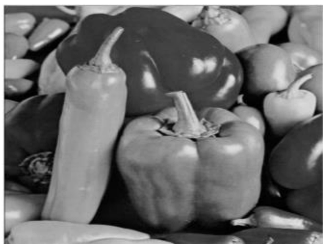
Lenna

(a) Original Image

(b) Watermarked Image

Peppers

(a) Original Image



(b) Watermarked Image

Fig. 5- Original and watermarked for six images

In above figure results for six images are shown. In (a) original image in (b) watermark image is shown. It is clear from the figures that original and watermarked is very much similar. To further analyze the images, baboon results are re-drawn for earlier method [11] at (Bit Per Pixel) BPP of 0.05 and 0.5. Thus, embedding capacity has been increased ten times, but PSNR reduces from 50.49 dB to 29.89 dB while SSIM varies from 0.9988 to 0.9005. Thus, it can be inferred that PSNR and SSIM shows same trends.

In Table 1 and 2, results are shown for PSNR and SSIM under different BPP under proposed interpolation technique. It is clear from the table that as BPP increases, the PSNR decreases. In among all the image PSNR value of airplane is comparatively better. However, the result for Baboon is poorest among these six images.

Table 1- PSNR for proposed RDH process for different images

	Bit Per Pixels					
	0.125	0.250	0.375	0.5	0.625	0.75
Image	**PSNR (dB)**					
Airplane	52.39	49.53	44.89	42.00	38.76	35.85
Baboon	43.48	37.28	33.11	29.46	25.49	20.27
Barbara	49.74	44.21	39.98	36.36	32.61	29.21
Boat	51.16	45.64	42.57	38.92	35.79	32.12
Lenna	50.89	45.96	42.87	38.97	35.64	32.34
Peppers	50.25	45.13	41.52	37.74	33.39	------

Table 2- SSIM for proposed RDH process for different images

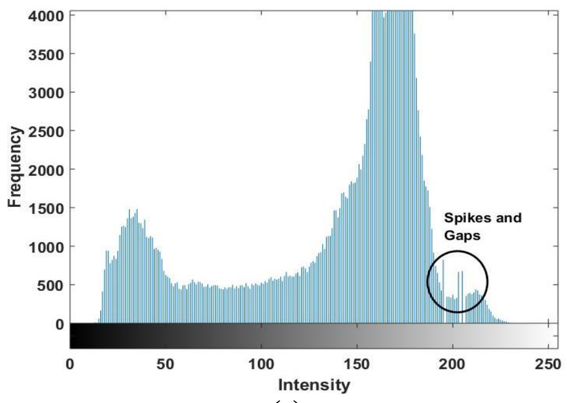
	Bit Per Pixels					
	0.125	0.25	0.375	0.5	0.625	0.75
Image	**SSIM**					
Airplane	0.9969	0.9937	0.9833	0.9601	0.9354	0.8877
Baboon	0.9934	0.9749	0.9485	0.8991	0.8241	0.7078
Barbara	0.9969	0.9884	0.9750	0.9464	0.8976	0.8373
Boat	0.9963	0.9868	0.9776	0.9499	0.9118	0.8379
Lenna	0.9958	0.9884	0.9766	0.9487	0.9048	0.8416
Peppers	0.9955	0.9870	0.9713	0.9404	0.8782	------

Table 3- Length of boundary map under different embedding rates

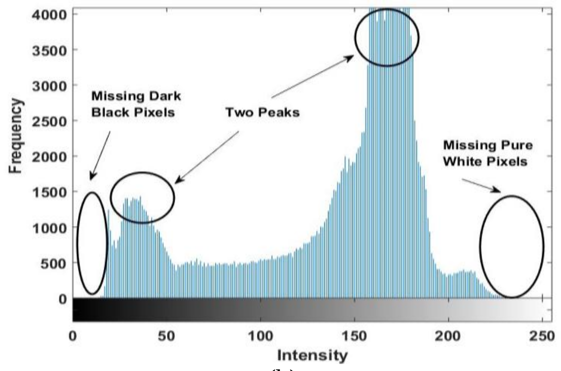
	Bit Per Pixels					
	0.125	0.25	0.375	0.5	0.625	0.75
Image	**PSNR (dB)**					
Airplane	0	0	0	0	0	0
Baboon	0	2	16	144	1359	19711
Barbara	0	0	0	0	0	54
Boat	0	0	0	0	0	8
Lenna	0	0	0	0	0	0
Peppers	88	495	1284	3694	12065	-----

In Table 2, results for proposed RDH process is shown in which SSIM values are detailed. Again, here as the BPP increases, the SSIM value decreases. Comparing the results at 0.125 BPP, best SSIM is 0.9969 which is for Airplane and Barbara images, PSNR for Barbara image 49.74 dB and PSNR for airplane is 52.39 dB. Similar trends in other images at different BPP is also shown, where SSIM is better in even in lesser PSNR. The boundary map for each of six images at different BPP is detailed in Table 3.

In general, DCT is used for the compression of images. DCT is a kind of lossy compression technique where minute finer details are sacrificed. The main advantages of DCT are concentration of energy in low frequency components and also reduction in blocking artifacts. It is also noticeable that, the quality of image also depends on camera and how picture is taken. To judge the quality of image SSIM is an excellent parameter with maximum value of 1. Therefore, using DCT image can be compressed and redundant information can be removed. In this work we have compressed image until unless SSIM remain 1, this condition makes sure that all the important information is intact. In DCT only 20 low frequency components are used and size of reduced images is 29 kB (minimum) for Airplane and 52 kB (maximum) for Baboon image.



(a)



(b)

Fig..6-(a) Original histogram for boat image (b) Histogram after DCT process

For example, boat image is considered, and its histogram is shown in Figure 6(a) it can be visualized that spikes and gaps are seen with two peaks; therefore, image consists of two distinctive parts that are clearly visible. In Figure 6(b) histogram of DCT compressed image is shown, where important peaks and other parts are intact but gaps and peaks are removed. Therefore, better embedding, PSNR and SSIM are possible in DCT based compressed image.

Table 4-PSNR for proposed RDH+ DCT process for different images

	Bit Per Pixels					
	0.125	0.250	0.375	0.5	0.625	0.75
Image	**PSNR (dB)**					
Airplane	54.10	51.18	47.88	44.88	42.74	39.72
Baboon	48.97	42.18	38.14	34.53	31.11	27.39
Barbara	51.86	49.00	44.57	41.06	37.83	34.36
Boat	54.42	51.27	47.57	44.73	42.11	38.55
Lenna	53.52	50.69	47.31	44.07	41.57	38.24
Peppers	53.25	50.33	46.49	43.58	40.81	36.67

Table 5-SSIM for proposed RDH+ DCT process for different images

	Bit Per Pixels					
	0.125	0.25	0.375	0.5	0.625	0.75
Image	**SSIM**					
Airplane	0.9978	0.9959	0.9926	0.9836	0.9723	0.9445
Baboon	0.9975	0.9887	0.9745	0.9459	0.9037	0.8355
Barbara	0.9978	0.9958	0.9889	0.9768	0.9542	0.9080
Boat	0.9982	0.9965	0.9933	0.9848	0.9723	0.9400
Lenna	0.9977	0.9956	0.9907	0.9804	0.9670	0.9342
Peppers	0.9975	0.9952	0.9888	0.9777	0.9615	0.9110

Table 6- Length of boundary map under different embedding rates

	Bit Per Pixels					
	0.125	0.25	0.375	0.5	0.625	0.75
Image	**PSNR (dB)**					
Airplane	0	0	0	0	1	4

Baboon	25	57	100	196	512	1480
Barbara	0	0	0	0	0	0
Boat	0	0	0	0	0	8
Lenna	0	0	0	0	0	0
Peppers	341	661	1153	2158	4915	11036

In Tables 4 to 6 results are again shown for PSNR, SSIM and boundary map at different BPP for six images is shown under proposed interpolation technique + DCT. Comparing, Tables 1 to 3 with tables 4 to 6 respectively; it has been found that results obtained with proposed approach +DCT are much better in comparison to proposed interpolation technique results.

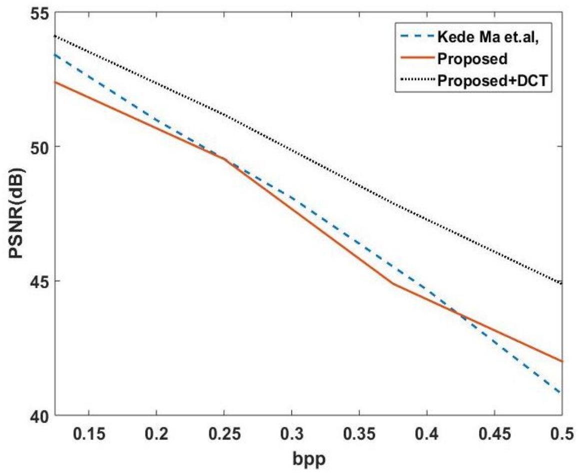


Fig.7. Comparison of PSNR vs. BPP for Airplane image.

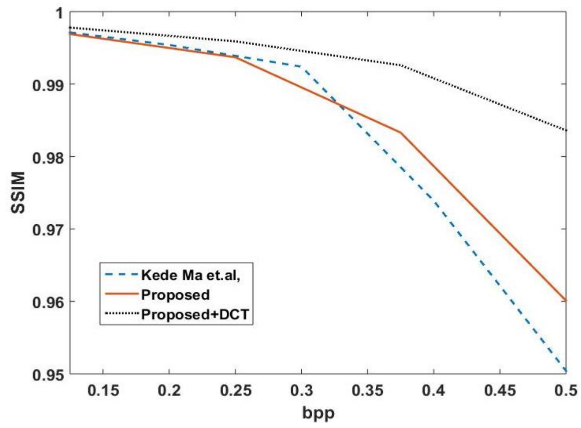


Fig. 8- Comparison of SSIM vs. BPP for Airplane image

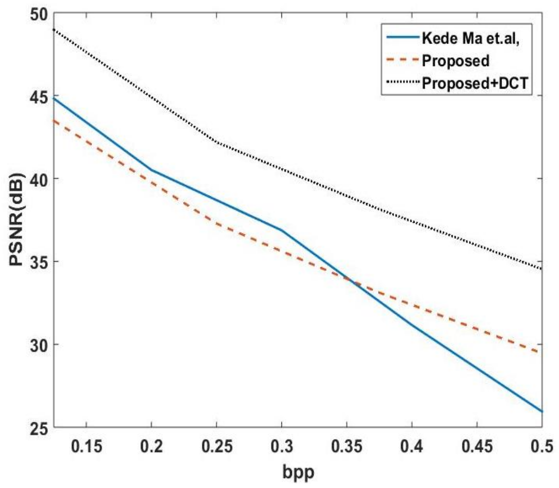


Fig. 9- Comparison of PSNR vs. BPP for Baboon image

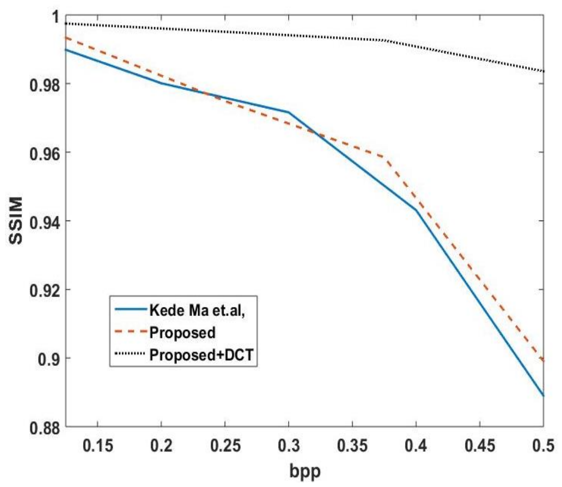


Fig.10- Comparison of SSIM vs. BPP for Baboon image

In Figure 7 the comparison of PSNR vs. BPP for Airplane image is shown. Results are compared with Kede Ma et.al, work [11]. It is clear from the figure a significant improvement in PSNR is observed. It is also noticeable that the improvement in PSNR is around 6 dB at the BPP of 0.5. The gap in PSNR increases, form lower BPP to higher BPP, which is desirable. In Figure 8, SSIM vs. BPP is detailed. Again, here the SSIM using proposed method is much superior in comparison to earlier results. As BPP increases the difference in SSIM also increases. The SSIM at the BPP of 0.5 is 0.9836 as compared to 0.95 in previous results. Same results for Baboon image is shown in Figures 9 and 10. Similar trends are again observed. In earlier method Kede Ma et.al.[11] maximum permissible bpp is 0.5, while in our case achievable bpp is 0.75. To make comparison easier, in Figure 11 bar graph is shown for PSNR vs. BPP; it is clear from the figure the performance of airplane image is best. In Figure 12, SSIM bar graph is shown and it has been found that SSIM is best for Boat image.

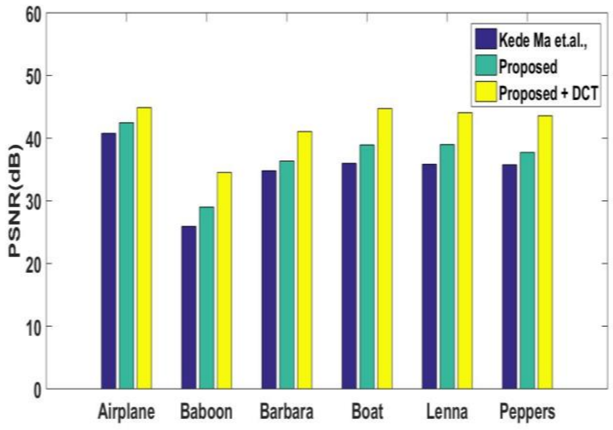


Fig.11- Comparison of PSNR for different images

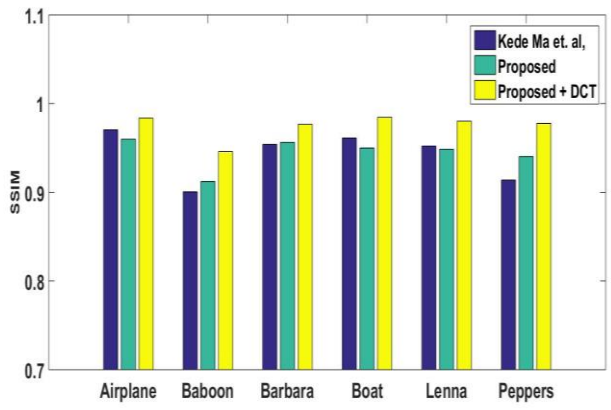


Fig. 12- Comparison of SSIM for different images

6. Conclusions

In this work, a novel reversible data hiding scheme for encrypted image with a low complexity is proposed, which consists of image encryption, data embedding and data extraction/image-recovery phases. The process is based on the estimation error of actual and estimated pixel values. Then histogram shifting is applied to embed data, and finally results are obtained in terms of PSNR, SSIM and Boundary map. This paper discusses the results for existing and proposed method. It has been found that with the proposed methods, the PSNR improvement is seen which is higher at higher BPP. It is also found that PSNR is not a very accurate measure for image quality assessment, and SSIM values should also be considered, sometime which is found to higher even in case of lower PSNR. From results it can be concluded that, proposed interpolation method found to be effective in comparison to older method. Moreover, DCT based efficient compression while maintain SSIM to 1, further improvements in results are observed.

References

[1] Fridrich, J., Goljan, M., & Du, R. (2001). Invertible authentication. In Security and Watermarking of Multimedia contents III International Society for Optics and Photonics, 4314, 197-209.

[2] Caldelli, R., Filippini, F., & Becarelli, R. (2010). Reversible watermarking techniques: An overview and a classification. EURASIP Journal on Information Security, 134546.

[3] Tian, J. (2003). Reversible data embedding using a difference expansion. IEEE transactions on circuits and systems for video technology, 13, 890-896.

[4] Alattar, A. M. (2004). Reversible watermark using the difference expansion of a generalized integer transform. IEEE transactions on image processing, 13, 1147-1156.

[5] Kim, H. J., Sachnev, V., Shi, Y. Q., Nam, J., & Choo, H. G. (2008). A novel difference expansion transform for reversible data embedding. IEEE Transactions on Information Forensics and Security, 3, 456-465.

[6] Hwang, J., Kim, J., & Choi, J. (2006, November). A reversible watermarking based on histogram shifting. In International Workshop on Digital Watermarking (pp. 348-361). Springer, Berlin, Heidelberg.

[7] Zhang, L., & Wu, X. (2006). An edge-guided image interpolation algorithm via directional filtering and data fusion. IEEE transactions on Image Processing, 15, 2226-2238.

[8] Hong, W., & Chen, T. S. (2011). Reversible data embedding for high quality images using interpolation and reference pixel distribution mechanism. *Journal of Visual Communication and Image Representation*, *22*, 131-140.

[9] Lin, C. C., & Shiu, P. F. (2010). High capacity data hiding scheme for DCT-based images. Journal of Information Hiding and Multimedia Signal Processing, 1, 220-240.

[10] Luo, L., Chen, Z., Chen, M., Zeng, X., & Xiong, Z. (2010). Reversible image watermarking using interpolation technique. IEEE Transactions on information forensics and security, 5, 187-193.

[11] Ma, K., Zhang, W., Zhao, X., Yu, N., & Li, F. (2013). Reversible data hiding in encrypted images by reserving room before encryption. IEEE Transactions on information forensics and security, 8, 553-562.

[12] Tsai, P., Hu, Y. C., & Yeh, H. L. (2009). Reversible image hiding scheme using predictive coding and histogram shifting. Signal processing, 89, 1129-1143.

[13] Zhang, X. (2011). Reversible data hiding in encrypted image. IEEE signal processing letters, 18, 255-258.