Reversible Data Hiding in Encrypted Images by Reversible Image Transformation

A. Sathi Babu[1] P.N.B.Swamy[2] P.Chenna rao[3]
[1]Assistant Professor in Dept. of ECE, NRIIT, Vijayawada, Andhra Pradesh
[2]Assistant Professor in Dept. of ECE, NRIIT, Vijayawada, Andhra Pradesh
[3]Professor, Dept. of of ECE, SSIET, Nuzvid, Andhra Pradesh

Abstract

Now a day's data hiding and data encryption is more popular. People seek more security for day to day life styles. Here we presents an approach for data hiding in encrypted images at cloud server based on reversible data hiding in encrypted images by reversible image transformation. This approach is different from previous data hiding and encryption based frame works, in which we presents the cipher texts that are used embedding image and stores at cloud it may attract to allows the user to transform the content of user original image data into another target image with same size and format. We realize an RIT based method by improving the image transformation technique to be reversible data hiding model approaches.

Keywords: *Reversible data hiding, PSNR, Target image, embedding image*

I. INTRODUCTION

Internet is a common way for data transmission. More and more data is available on the internet due to growth in information technology. With growth in digital data there have some security problems. To release the burden of data management user preferred outsourcing of data to the cloud. For data privacy and security many user utilized cryptography techniques for data encryption before uploading it on cloud. In order to confidentially convey secret messages stenography is the efficient way used by user for multimedia data hiding. In stenography, carrier can be images, audio or video. Original image is treated as cover image and the other image in which data is embedded known as stego image. Trust and security have prevented businesses from fully accepting cloud platforms. To protect clouds, providers must first secure virtualized datacenter resources, uphold user privacy, and preserve data integrity. The authors suggest using a trust-overlay network over multiple data centers to implement a reputation system for establishing trust between service providers and data owners.

Data coloring and software watermarking techniques protect shared data objects and massively distributed software modules. These techniques safeguard multi-way authentications, enable single sign-on in the cloud, and tighten access control for sensitive data in both public and private clouds.[1]The Cloud Security Alliance has identified a few critical issues for trusted cloud computing, and several recent works discuss general issues on cloud security and privacy. Public and private clouds demand different levels of security enforcement. We can distinguish among different service-level agreements (SLAs) by their variable degree of shared responsibility between cloud providers and users. Critical security issues include data integrity, user confidentiality, and trust among providers, individual users, and user groups. The three most popular cloud service models have varying security demands [2].Reversible data hiding (RDH) has been intensively studied in the community of signal processing. Also referred as invertible or lossless data hiding, RDH is to embed a piece of information into a host signal to generate the marked one, from which the original signal can be exactly recovered after extracting the embedded data. The technique of RDH is useful in some sensitive applications where no permanent change is allowed on the host signal. In the literature, most of the proposed algorithms are for digital images to embed invisible data or a visible watermark. So far, many RDH methods on images have been proposed. In essence, all these methods can be viewed as a process of semantic lossless compression [3], [24], in which some space is saved for embedding extra data by losslessly compressing the image. Here in, "semantic compression" means that the compressed image should be close to the original image, and thus one can get a marked image with good visual quality. Because the residual part of images, e.g., the prediction errors (PE), has small entropy and can be easily compressed, almost all recent RDH methods first generate PEs as the host sequence [5]–[7], and then reversibly embed the message into the host sequence by modifying its histogram with methods like histogram shifting or difference expansion. To evaluate the performance of a RDH algorithm, the hiding rate and the marked image quality are important metrics. There exists a trade-off between them because increasing the hiding rate often causes more distortion in image content. To measure the distortion, the peak signal-to-noise ratio (PSNR) value of the marked image is often calculated. Generally speaking, direct modification of image histogram provides less embedding capacity. In contrast, the more recent algorithms manipulate the more centrally distributed prediction errors by exploiting the correlations between neighboring pixels so that less distortion is caused by data hiding.

II. PROPOSED SYSTEM

Digital data hiding is a scheme used for embedding information into host media such as audio, video, and images. It can be utilized in many ways, including protection of intellectual property, integrity authentication. The major problem in data hiding is the distortion of the original media from the embedding process - making it impossible to recover. Sincere covering the original media is essential in applications such as medical imagery, law enforcement, and remote sensing, distortion cannot be tolerated. Finding a technique to eradicate distortion has proven to be a challenge for researchers. To address the problem, the reversible data hiding (RDH) technique has been introduced. This technique can completely recover both the host and the embedded data. Even so, differences remain between the original and the embedded host media. When comparing the original host media with the embedded, the best technique provides the lowest difference between media with the highest embedding capacity [9].The proposed system is illustrated in Fig.2.1. We propose a novel framework for RDH-EI by using reversible image transformation (RIT) [10]. RIT transfers the semantic (content) of the original image I into the semantic of another image J, and "reversibility" means that I can be losslessly restored from the transformed image. Therefore RIT can be viewed as a special encryption scheme, called "Semantic Transfer Encryption." In other words, the resultant transformed image which is also the encrypted image E(I) will look similar with J The image J is selected to be irrelevant with I but as the same size of I, and thus the content of the image I is protected. Because the "encrypted image" is in a form of plaintext, it will avoid the notation of the cloud server, and the cloud server can easily embed data into the "encrypted image" with traditional RDH methods for plaintext images.

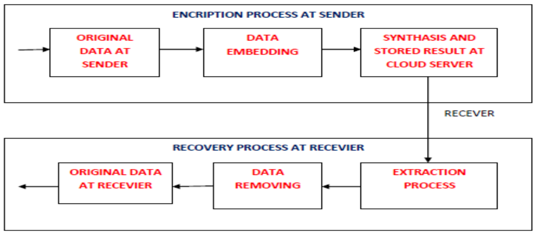

ENCRIPTION PROCESS AT SENDER
ORIGINAL DATA AT SENDER
DATA EMBEDDING
SYNTHASIS AND STORED RESULT AT CLOUD SERVER
RECEVER
RECOVERY PROCESS AT RECEVIER
ORIGINAL DATA AT RECEVIER
DATA REMOVING
EXTRACTION PROCESS

Fig.2.1: Reversible image transformation

A. Image Encryption:

Assume the original image is in uncompressed format and each pixel with gray value falling into [0, 255] is represented by 8 bits.

Where $b_{i,j,0}, b_{i,j,1}, \dots \dots \dots b_{i,j,\tau}$ denotes the bits of a pixel and (i, j) indicates the pixel position, and the gray value as $P_{i,j,k.}$ Thus

$$\boldsymbol{b_{i,j,k}} = \left[\frac{P_{i,j}}{2^k}\right] \boldsymbol{mod\ 2, k = 0, 1, 2, .., 7} \quad \textbf{(1)}$$

And

$$\boldsymbol{P_{i,j} = \sum_{u=0}^{7} b_{i,j,k}.\, 2^{\,k}} \qquad \textbf{(2)}$$

In encryption phase, the exclusive-or results of the original bits and pseudo-random bits are calculated as

$$\boldsymbol{B_{i,j,k} = b_{i,j,k} \oplus \mathrm{r}_{\mathrm{i,j,k}}} \qquad \textbf{(3)}$$

Where $r_{i,j,k}$ are determined by an encryption key using a standard stream cipher. Then, $B_{i,j,k}$ are concatenated orderly as the encrypted data.A number of secure stream cipher methods can be used here to ensure that anyone without the encryption key, such as a potential attacker or the data hider, cannot obtain any information about original content from the encrypted data [11].

B.Data Embedding and Image decryption

The seller of digital multimedia content encrypts the pristine data and embeds an encrypted dactyl gram supplied by the client.

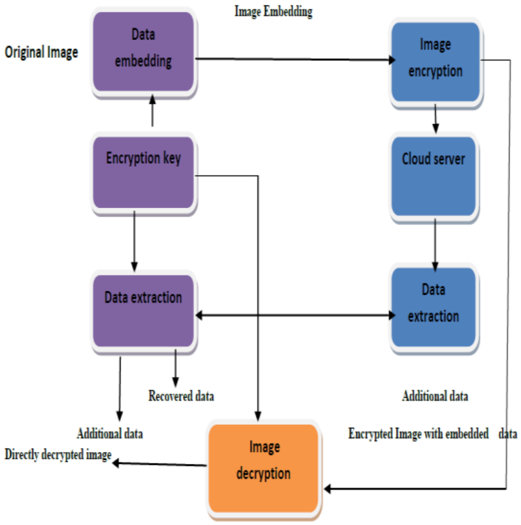


Fig.2.2: General block diagram for Data Embedding

In this case, the seller cannot obtain the client's dactyl gram, and the buyer cannot access the pristine version unless he/she makes the payment to consummate the transaction. Some methods of RDH in encrypted pictures have been intended. Zhang divides the encrypted image into blocks, and embeds one bit into each slab by flipping three LSBs of half the pixels in the block.

On the receiver side, the secret bits are extracted and the pristine pictures recuperated by analyzing the fluctuation of the pixel values in all decrypted block RIT scheme for encrypted images by compressing the encrypted data utilizing a source coding scheme with side information, making data extraction independent of encryption. To do so, LSBs of some pixels are first embedded into other pixels utilizing a traditional RDH method, and the image is then encrypted. As a result, positions of these LSBs in the encrypted picture can be used for embedding information with the data-hider. As JPEG is widely utilized, the scheme is defined to fluster the principal content of the pristine image while preserving the bit stream structure.

The secret bits are encoded with error rectification codes and then embedded into the JPEG bit stream. The differences between the novel framework and previous frameworks, which shows that, by frameworks VRAE and RRBE, the user's images are stored in the form of cipher text in the cloud account, while by the RIT-based frame work, the image is stored in a form of plaintext. In the framework VRAE such as schemes in [11] and [12], the image owner (the sender) encrypts the image I into E(I)with a key K. The cloud server embeds data by compressing the encrypted image E(I) and generates Ew(I) that is stored in the cloud.

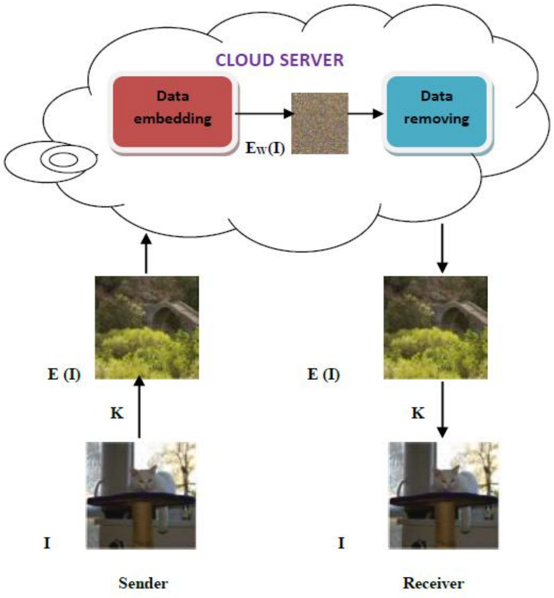


Fig.2.3: RIT based Data Embedding

When getting a retrieval request, the cloud server returns Ew(I)to the receiver, may be an authorized third party, who generates I through a process of joint decompression and decryption with the key K. Here in, E(I) may be just Ew(I) or a modified version obtained by removing the embedded data. Note that the cloud server cannot restore E(I) from Ew(I), since decompression should be joined with decryption with the help of K. In this framework, the complexity is taken on by the receiver who must join the process of decompression and decryption to get the original image. In other words, the compression-based RDH method used by the cloud server should be specified together with the receiver, i.e., the RDH method is receiver-related. In the framework RRBE, such as schemes in [13], [14], the image owner (the sender) reserves room from the image I and encrypts it into E(I) with a key K, and then sends it to the cloud server who embeds data into the reserved room and generates Ew(I). Ew(I) is stored in the cloud, from which the cloud server can extract the data that is used for management. When an authorized user (the receiver) wants to retrieve the image, the cloud server can restore E(I) from Ew(I) and send E(I) to the user who can decrypt E(I)and get I with the key K. In the framework RRBE, the complexity is borne by the sender who should reserve room for RDH by exploiting the redundancy within the image and the RDH method used by the cloud should be mentioned with the sender, that is, the RDH method used by cloud is sender-related. In the RIT based framework depicted in Fig.2.3, the image I is "encrypted" into another plaintext image E(I) with a key K, so all images of the users, encrypted or not, will be stored in the cloud in the form of plaintexts.

The cloud server can embed/extract data into/from E(I) with any classical RDH method for plaintext images. And E(I) can be recovered from the watermarked image Ew(I) by the cloud and sent back to the authorized user who anti-transforms it to get the original image I with the key K. The main contributions of this novel framework include: 1) the idea of RIT is exploited for RDH-EI, by which the user can outsource the encrypted image to the cloud in a form of plaintext and thus it will avoid the attention of the curious cloud; and 2) in the RIT based framework, the cloud server can easily embed data into the "encrypted image" by arbitrarily selecting RDH methods for plaintext images such as those in [4], [6], [7], [10]. In other words, the RDH used by the cloud is irrelevant with both the sender and receiver, which is called a client-free RDH-EI scheme by us. "Client free" is important for the scenarios of public clouds, in which it is hard for the cloud server to ask the clients how to encrypt or decrypt their data, because the cloud is thought to be only semi-honest [13].

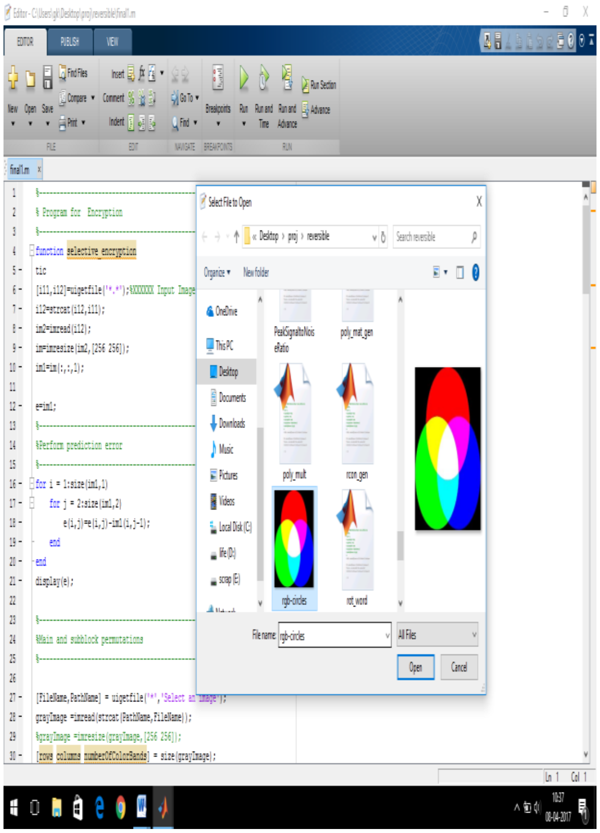
We propose a method of RIT to encrypt spatial images, which is inspired by the technique of image transformation proposed by Lee and Tsai. Lee et al.'s method can transform the original image to a freely-selected target image with the same size, yielding a secret-fragment-visible mosaic image. But the original image cannot be restored in a lossless way. It is not reversible, so it is not suitable for the scenario of RDH-EI. We will modify Lee et al.'s method to be reversible and obtain an encrypted image which looks like the target image. For color images, we transform the color channel R, G, and B respectively in the same manner. So we just take gray images (one channel) as an example to describe the method. For an original image I, we randomly select a target image J having the same size with I from an image database. Firstly, we divide the original image I and the target image J into N non-overlapping blocks respectively, and then pair the blocks of I and J as a sequence such that $(B_1,T_1),\ldots,(B_N,T_N)$, where Bi is an original block of I and Ti is the corresponding target block of J, 1≤ i ≤ N. We will transform Bi toward Ti and generate a Ti similar to Ti. After that, we replace each Ti with Ti in the target image J to get the transformed image J. Finally we embed some accessorial information (AI) into J with an RDH method and generate the ultimate "encrypted image" E(I).These AI is necessary for recovering I from J. Before being embedded, these AI will be compressed and encrypted with a key K shared with the receiver, so only a receiver having K can decrypt E(I). The proposed transformation process consists of three steps: block pairing, block transformation and AI embedding .We will mainly elaborate the first two steps in the sections and the third step can be implemented by any traditional RDH method.

IV. EXPERIMENTAL RESULTS

The following results show the strength of proposed algorithm and results meets best from the previous approaches. Let us take an image as original image in other words wanted image that can be embedded with a target image by using RIT algorithm. We only decrypted original image when only by using right key otherwise we detect falls image, the process is shown below.

Step: 1

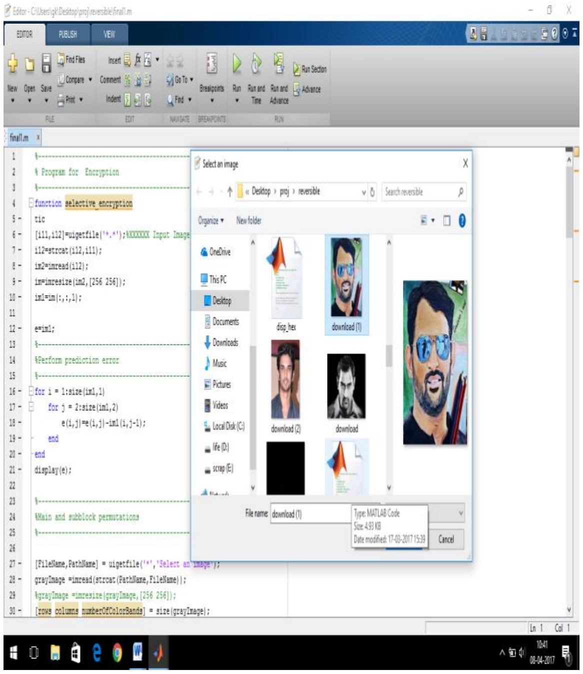
Select the image which we have to hide from being visible. So that the code will convert the Original image to 256x256 and operate on it.



```
%-----------------------------------------
% Program for  Encryption
%-----------------------------------------
function selective_encryption
tic
[i11,i12]=uigetfile('*.*');%XXXXXX Input Image
i12=strcat(i12,i11);
im2=imread(i12);
im=imresize(im2,[256 256]);
im1=im(:,:,1);

e=im1;
%-----------------------------------------
%Perform prediction error
%-----------------------------------------
for i = 1:size(im1,1)
    for j = 2:size(im1,2)
        e(i,j)=e(i,j)-im1(i,j-1);
    end
end
display(e);

%-----------------------------------------
%Main and subblock permutations
%-----------------------------------------

[FileName,PathName] = uigetfile('*','Select an image');
grayImage =imread(strcat(PathName,FileName));
%grayImage =imresize(grayImage,[256 256]);
[rows columns numberOfColorBands] = size(grayImage);
```

Step: 2

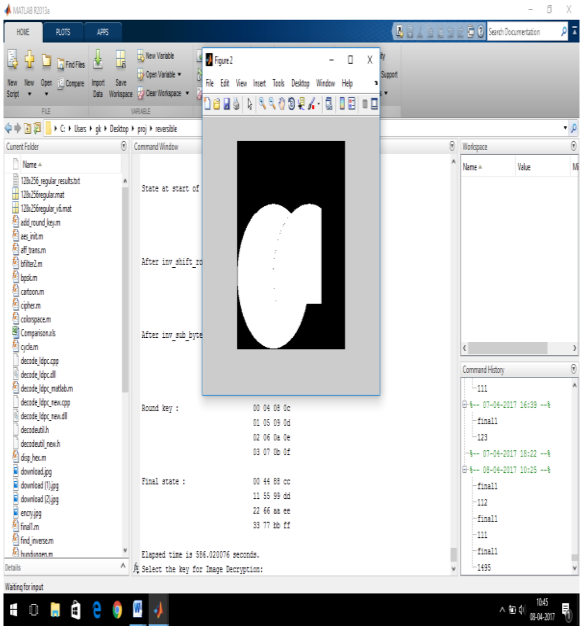
Select the cover image which we have no privacy with it. To be an encryption covers for the image.



```
%-----------------------------------------
% Program for  Encryption
%-----------------------------------------
function selective_encryption
tic
[i11,i12]=uigetfile('*.*');%XXXXXX Input Image
i12=strcat(i12,i11);
im2=imread(i12);
im=imresize(im2,[256 256]);
im1=im(:,:,1);

e=im1;
%-----------------------------------------
%Perform prediction error
%-----------------------------------------
for i = 1:size(im1,1)
    for j = 2:size(im1,2)
        e(i,j)=e(i,j)-im1(i,j-1);
    end
end
display(e);

%-----------------------------------------
%Main and subblock permutations
%-----------------------------------------

[FileName,PathName] = uigetfile('*','Select an image');
grayImage =imread(strcat(PathName,FileName));
%grayImage =imresize(grayImage,[256 256]);
[rows columns numberOfColorBands] = size(grayImage);
```

Step: 3

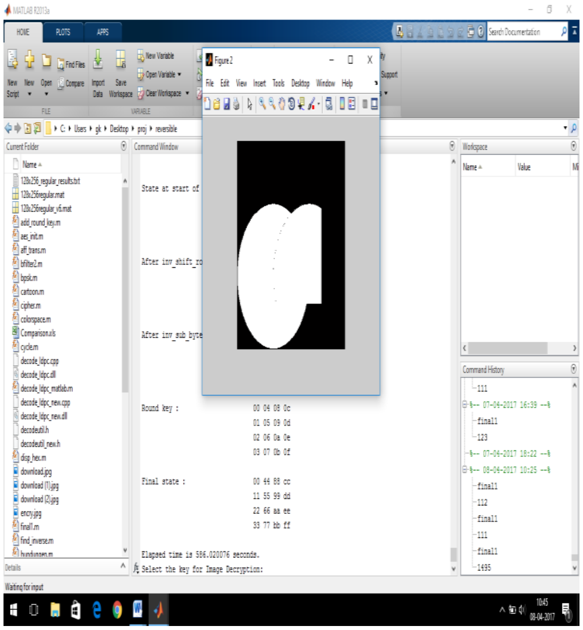
Give the key for encrypting the image.



```
State at start of

After inv_shift_ro

After inv_sub_byte

Round key :              00 04 08 0c
                         01 05 09 0d
                         02 06 0a 0e
                         03 07 0b 0f

Final state :            00 44 88 cc
                         11 55 99 dd
                         22 66 aa ee
                         33 77 bb ff

Elapsed time is 586.020076 seconds.
Select the key for Image Decryption:
```

Step: 4

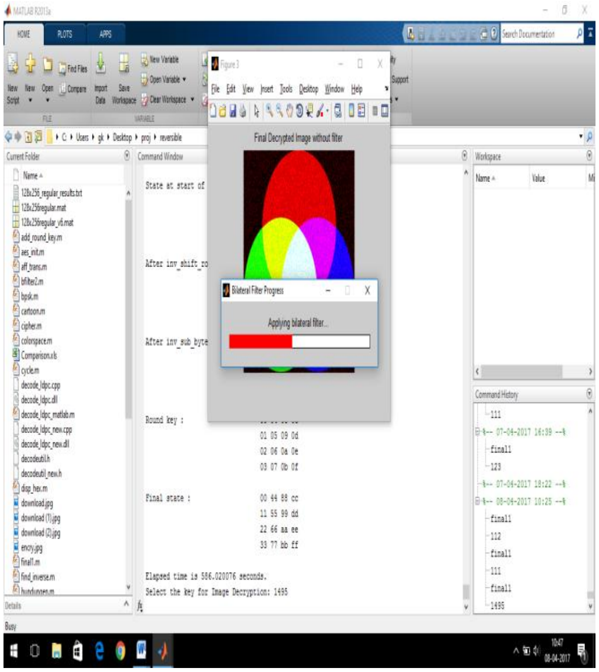
This is the encrypted image and is nowhere look like the original image which we have to hide.



```
State at start of

After inv_shift_ro

After inv_sub_byte

Round key :              00 04 08 0c
                         01 05 09 0d
                         02 06 0a 0e
                         03 07 0b 0f

Final state :            00 44 88 cc
                         11 55 99 dd
                         22 66 aa ee
                         33 77 bb ff

Elapsed time is 586.020076 seconds.
Select the key for Image Decryption:
```

Step: 6

For decryption, after giving the key for decryption we will get the output image i.e., Original image that we have hide with application of bilateral filter.



```
State at start of

After inv_shift_ro

After inv_sub_byte

Round key :
                         01 05 09 0d
                         02 06 0a 0e
                         03 07 0b 0f

Final state :            00 44 88 cc
                         11 55 99 dd
                         22 66 aa ee
                         33 77 bb ff

Elapsed time is 586.020076 seconds.
Select the key for Image Decryption: 1495
```

Step: 7

This is the image before and after application of bilateral filter.

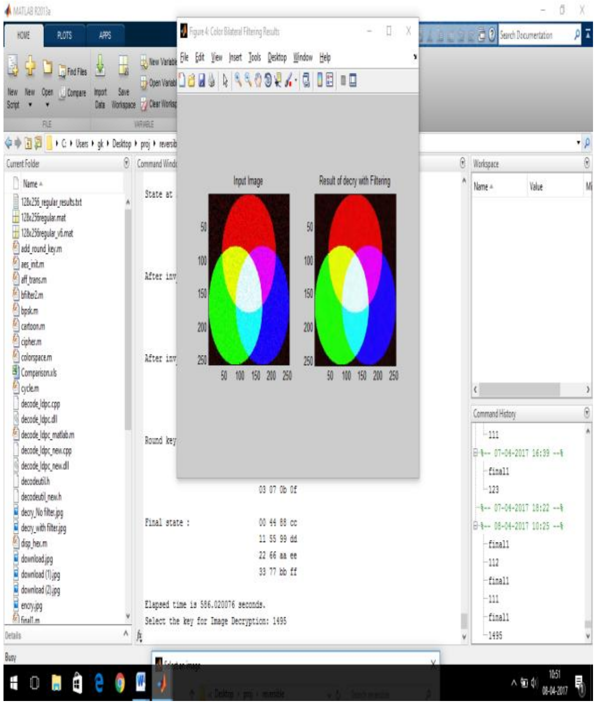
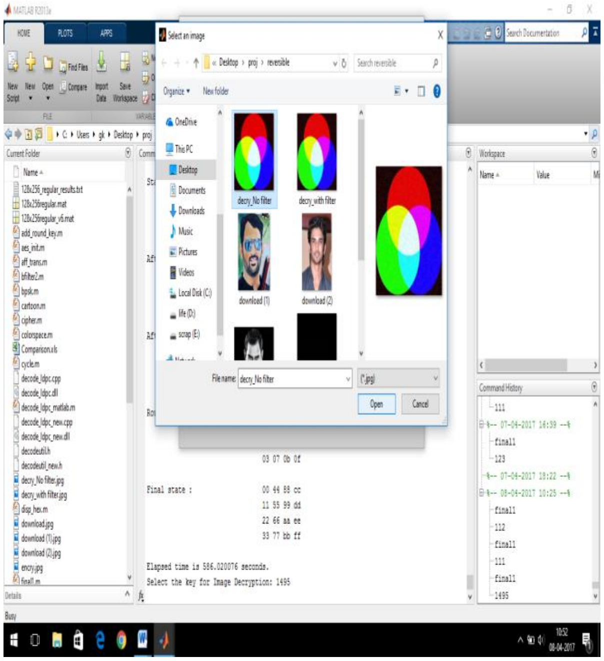


Figure 4: Color Bilateral Filtering Results
File Edit View Insert Tools Desktop Window Help
Input Image
Result of decry with Filtering
50
100
150
200
250
50 100 150 200 250
Command Window
Workspace
Command History

```
State at

After inv

After inv

Round key

                         03 07 0b 0f

Final state :            00 44 88 cc
                         11 55 99 dd
                         22 66 aa ee
                         33 77 bb ff

Elapsed time is 586.020076 seconds.
Select the key for Image Decryption: 1495
```

Step: 8

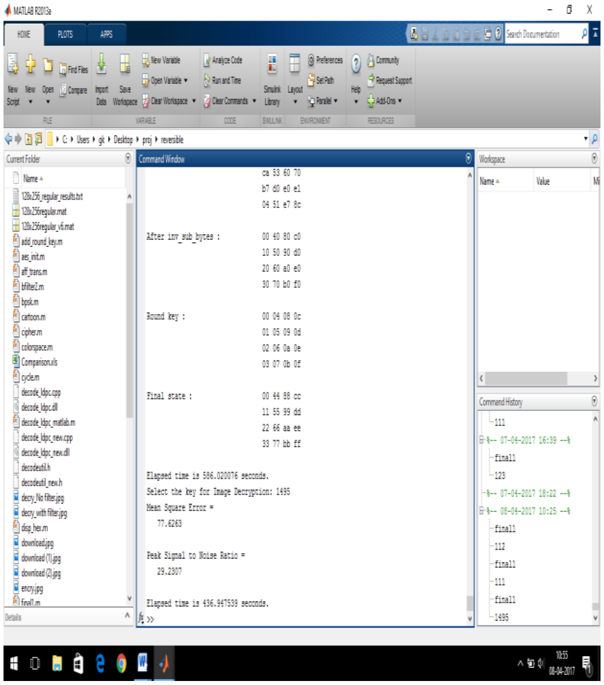
In order to know the quality of compression we have to calculate the PSNR Value of the images before and after the encryption process.



```
                              03 07 0b 0f

Final state :                 00 44 88 cc
                              11 55 99 dd
                              22 66 aa ee
                              33 77 bb ff

Elapsed time is 586.020076 seconds.
Select the key for Image Decryption: 1495
```

Step: 9

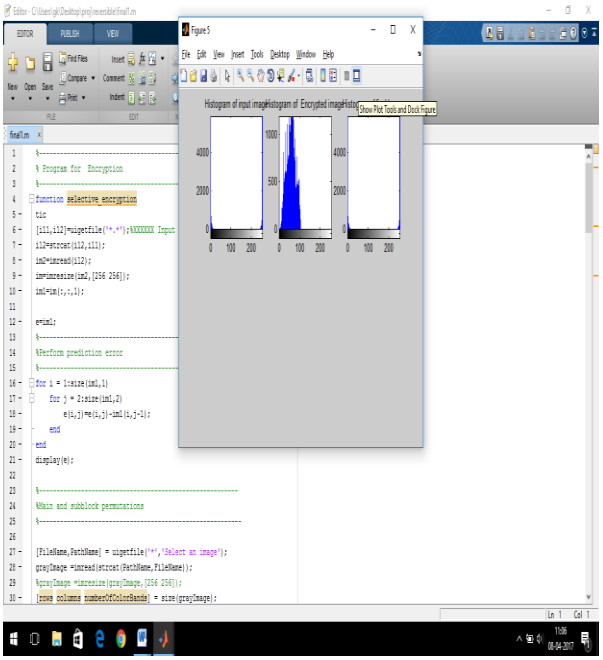
From the characteristics of PSNR it is considered that if the PSNR value is >20 Then the compression is Faithfull compression.



```
                              ca 53 60 70
                              b7 d0 e0 e1
                              04 51 e7 8c

After inv_sub_bytes :         00 40 80 c0
                              10 50 90 d0
                              20 60 a0 e0
                              30 70 b0 f0

Round key :                   00 04 08 0c
                              01 05 09 0d
                              02 06 0a 0e
                              03 07 0b 0f

Final state :                 00 44 88 cc
                              11 55 99 dd
                              22 66 aa ee
                              33 77 bb ff

Elapsed time is 586.020076 seconds.
Select the key for Image Decryption: 1495
Mean Square Error =
   77.6263

Peak Signal to Noise Ratio =
   29.2307

Elapsed time is 436.947539 seconds.
>>
```

Step: 10

The histogram values of the input and output images are identical which represents the restoration efficiency of the method.



```
%----------------------------------------
% Program for  Encryption
%----------------------------------------
function selective_encryption
tic
[i11,i12]=uigetfile('*.*');%XXXXXX Input
i12=strcat(i12,i11);
im2=imread(i12);
im=imresize(im2,[256 256]);
im1=im(:,:,1);

e=im1;
%----------------------------------------
%Perform prediction error
%----------------------------------------
for i = 1:size(im1,1)
    for j = 2:size(im1,2)
        e(i,j)=e(i,j)-im1(i,j-1);
    end
end
display(e);

%-----------------------------------------------------
%Main and subblock permutations
%-----------------------------------------------------

[FileName,PathName] = uigetfile('*','Select an image');
grayImage =imread(strcat(PathName,FileName));
%grayImage =imresize(grayImage,[256 256]);
[rows columns numberOfColorBands] = size(grayImage);
```

The tested results is shown as below

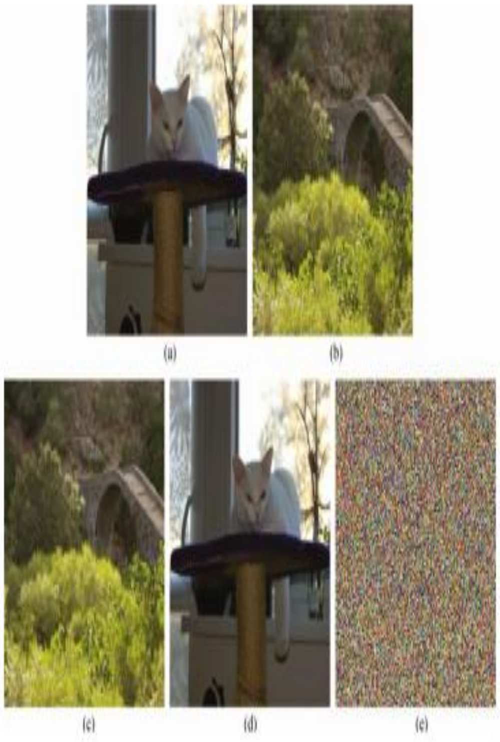


Fig.4.1Experimental results of test images (a) Original image. (b) Target image. (c) Encrypted image. (d) Decrypted image (right key). (e) Decrypted image (wrong key).

V. CONCLUSION

We realize an RIT based method by improving the image transformation technique to be reversible. For this work our proposed work RDH-EI based RIT. Here RIT transforms the schematic of original image to schematic of another image called target image. Hence the target image protects and provides privacy of original image at cloud server.

REFERENCES

[1] Z.Ni,Y.Shi,N.Ansari, and S.Wei, "Reversible data hiding, "IEEE Trans. Circuits Syst. Video Technol., vol. 16, no. 3, pp. 354–362, Mar. 2006.

[2] K.Hwang and D.Li, "Trusted cloud computing with securer sources and data coloring," IEEE Internet Computer., vol.14 ,no.5,pp.14–22,Sep./Oct. 2010.

[3] F.Willems, D. Maas, and T. Kalker, "Semantic lossless source coding," in Proc. 42nd Annu. Allerton Conf. Commun. Control Comput., 2004, pp. 1411–1418.

[4] W.Zhang, X. Hu, X. Li, and N. Yu, "Recursive histogram modification: Establishing equivalency between reversible data hiding and lossless data compression," IEEE Trans. Image Process., vol. 22, no. 7, pp. 2775–2785,Jul. 2013.

[5] V.Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y. Q. Shi, "Reversible watermarking algorithm using sorting and prediction," IEEE Trans. Circuits Syst. Video Technol., vol. 19, no. 7, pp. 989–999, Jul. 2009.

[6] B.ou, X. Li, Y. Zhao, R. Ni, and Y. Shi, "Pairwise prediction-error expansion for efficient reversible data hiding," IEEE Trans. Image Process., vol. 22, no. 12, pp. 5010–5021, Dec. 2013.

[7] I.-C.Dragoi and D. Coltuc, "Local-prediction-based difference expansion reversible watermarking," IEEE Trans. Image Process., vol. 23, no. 4, pp. 1779–1790, Apr. 2014.

[8] J.Tian, "Reversible data embedding using a difference expansion," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 8, pp. 890–896, Aug. 2003.

[9] Weiming Zhang Hui Wang, Dongdong Hou, and Nenghai Yu "Reversible Data Hiding in Encrypted Images by Reversible Image Transformation" IEEE transactions on multimedia, vol. 18, no. 8, august 2016

[10] X.Hu, W. Zhang, X. Li, and N. Yu, "Minimum rate prediction and optimized histograms modification for reversible data hiding," IEEE Trans. Inf. Forensics Security, vol. 10, no. 3, 653–664, Mar. 2015.

[11] Xinpeng Zhang "Reversible Data Hiding in Encrypted Image," IEEE SIGNAL PROCESSING LETTERS, VOL. 18, NO. 4, APRIL 2011.

[12] W.Hong, T. Chen, and H. Wu, "An improved reversible data hiding in encrypted images using side match," IEEE Signal Process. Lett., vol. 19, no. 4, pp. 199–202, Apr. 2012.

[13] S.Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in Proc. IEEE INFOCOM, Mar. 2010, pp. 1–9.

Author Profiles

P.Chenna Rao, received his PhD from Andhra University IN 2016 and received M.Tech from JNTU University, HYD. He having more than 15 years teaching experience and guided number of M.Tech and B.Tech projects. He is currently working as a Professor in SSIET Engineering College and he is an active member of MISOI, MISTE respectively. His area of interest is image processing, signal processing. He has published many papers in national and international journals.

A.Sathi Babu, received his M.Tech from CR Reddy College of Engineering, Andhra University, Present, He is worked as an assistant professor in NRI Institute of Technology & Engineering, in ECE Department. His area of interest is wireless communications, network information theory, source, channel, and network.