Image based Secret Communication using Double Compression

Chaitanya Kommini
Dept of IT,
SVEC, Tirupati
J.N.T.Univeristy, Anantapur
A.P – 517 102, India

Kamalesh Ellanti
Dept of IT,
SVEC, Tirupati
J.N.T.Univeristy, Anantapur
A.P – 517 102, India

Srinivasulu Asadi
Dept of IT,
SVEC, Tirupati
J.N.T.Univeristy, Anantapur
A.P – 517 102, India

ABSTRACT
The Information exchange via any media needs privacy and secrecy. Cryptography is widely used for providing privacy and secrecy between the sender and receiver. But, now, along with Cryptography, we are using Steganography to have more protection to our hidden data. In this paper, we show how a JPEG can be used as an embedding space for a message by adjusting the values in the JPEG Quantization tables (QTs). This scheme also uses some permutation algorithms and it can be widely used for secret communication. This JPEG double compression will give satisfactory decoded results.

GENERAL TERMS
Image Processing, Signal Processing, Steganography.

INDEX TERMS
Quantization Tables, JPEG Compression, Steganography, Double Compression.

1. INTRODUCTION
There is a great demand for speed and integrity of the information transfer on the Internet. In addition to the speed and integrity, there is a great need for secrecy and privacy in the information exchange. Many research works have used cryptography to provide secure communication. But besides cryptography, Steganography is also a widely used technique for providing more protection to the hidden data. Using this technique the data hiding is done in such a way that it avoids the people to even think that there exists information in the carrier media.

Data hiding inside an image can be done using many domains where Steganography algorithms exploit like spatial domain or DCT. Images are also of various types but we take the JPEG format because it is the commonly used standard with 30:1 compression ratio and very less loss in image quality. Other reasons for JPEG standard are that we can adjust the degree of compression.

In this paper, data is protected in two stages. The first stage is encryption stage where the message to be embedded is encrypted using certain encryption algorithms. In this stage, permutation algorithms are used and they require a pair of numbers as a key. After this stage, the encrypted message is embedded in JPEG image by managing the values in the JPEG Quantization tables (QTs). Thus from this hiding stage we get the JPEG image. This image has some regions with different image quality. This whole process is done at the sender side. At the receiver end, all the above steps are performed in the reverse order to extract the original data i.e., first we extract the scrambled message in the image and later using the key we perform the decryption.

In the later sections we describe the JPEG quantization process and the process of embedding encrypted data inside an image. Then we give the experimental results.

2. INFORMATION EMBEDDING AND EXTRACTING SCHEME

2.1 JPEG image compression

The first step in image compression is to convert the image from RGB format into YCbCr format. Later, the chrominance channels (Cb and Cr) are subsampled to the rate half of the Y.

Now, the partition of each channel into 8x8 non-overlap blocks takes place and the shifting of pixel values in the channel from [0-255] to [-128,127] is done to move to next step i.e., Discrete Cosine Transformation (DCT) [5]. In the DCT step, the low-frequency and the high frequency coefficients are separated and it allows the truncating of high-frequency coefficients by applying specific quantization tables. A larger coefficients set will lead to high compression ratio and reduced image quality. There is a chance of different camera models having same image quality but different QTs. Examples of such QTs can be seen in [6].

The QT of Photoshop and Matlab in fig 1 and fig 2 are different but they have same quality factor 80.

6	**4**	**4**	**6**	**10**	**16**	**20**	**24**
5	**5**	**6**	**8**	**10**	**23**	**24**	**22**
6	**5**	**6**	**10**	**16**	**23**	**28**	**22**
6	**7**	**9**	**12**	**20**	**35**	**32**	**25**
7	**9**	**15**	**22**	**27**	**44**	**41**	**31**
10	**14**	**22**	**26**	**32**	**42**	**45**	**37**
20	**26**	**31**	**35**	**41**	**48**	**48**	**40**
29	**37**	**38**	**39**	**45**	**40**	**41**	**40**

Fig 1: QT of Photoshop

2.2 The main principle in the proposed Message hiding scheme

Suppose consider the case where a block of 8x8 DCT coefficients (let cij be c11 to c88) be mapped to a QT of same size. Here, each DCT coefficient is quantized to corresponding value in QT. The quantized value is rounded to an integer which is nearest to the value.

$$Cij = round\ (Cij/qij) \quad (1)$$

6	**4**	**3**	**6**	**9**	**15**	**19**	**22**
4	**4**	**5**	**7**	**9**	**21**	**22**	**20**
5	**4**	**6**	**9**	**15**	**21**	**25**	**21**
5	**6**	**8**	**10**	**19**	**32**	**30**	**23**
6	**8**	**13**	**21**	**25**	**40**	**38**	**28**
9	**13**	**20**	**24**	**30**	**29**	**42**	**34**
18	**24**	**29**	**32**	**38**	**45**	**45**	**37**
27	**34**	**35**	**36**	**42**	**37**	**38**	**37**

Fig 2: QT of Matlab

If we recalculate the DCT set (C1=Cij x qij) and if we quantize the c1 set again, it gives the DCT coefficients set C2. Whenever q2=q1, the difference between C2 and C1 is minimum and this feature is made clear in [4]. In other words, if we process the C1 with lower quality, then the difference in (2) will be a minimum as q2=q1 (only when we take difference in (2) in terms of function of q2).

$$\boldsymbol{difference} = \sum_{i,j} \left| c_1^{ij} - c_2^{ij} \right|^2 (i, j = 1, \ldots, 8). \quad (2)$$

To embed the information in JPEG image, some regions which are so specific are compressed with lower quality q0. These lower quality regions will carry the secret information.

(1) Encryption stage

Fig 3 shows the steps in the embedding and extracting of the message in the proposed scheme. In the first stage, we apply an automorphism algorithm [7] to the image where the permutation of pixels in the hidden message takes place. Here, we can also apply an encryption algorithm with a strong key. Now we apply modulo operation to the image which gives a random pattern. For example, in the fig 3, we have applied modulo operator with parameter e=2 on the original image for 66 times (k=66) and this gives the image next to the original image in fig 3. 'W' is the key for decryption. So for recovering the image in the receiver side, we have to know the pair (e,k).

(2) Embedding stage

We manage the quality factor of each pixel in the hidden image and we embed it in the specific region of the JPEG image. Let us consider our secret message C be a binary image of size LxM with C(l,m)=0 for black pixels and C(l,m)=1 for white pixels. The Host image is O of size PxQ.

The aspect ratio of secret image should be same as that of host image so that the shape of the secret image remains unchanged. This embedding has two steps:

Step 1:

The original image is divided into blocks of size P/L x Q/M.

Step 2:

If C(l,m)=0, O(x,y) is compressed with Q2

If C(l,m)=1, O(x,y) is compressed withQ1,

Where x,y values satisfy the condition round(x/(P/L))=l and round(y/(Q/M))=m.

After embedding, the image is saved in JPEG format. The quality factor of the saved image must be high i.e., at least 95.

In decoding process, first the receiver has to detect the regions with different quality factors so that the scrambled pattern will easily be extracted from the Host image. The key which was shared with the sender will be used to rescramble the extracted pattern. Later the reconstruction of hidden message takes place.

For example, consider that we have embedded black pixels in lower quality blocks. The extracting of image from Host image is as follows:

Let the recipient's received image be I1. The recipient will resave the image I1 to I2 with lower quality factor Q2. Subtracting I2 from I1 yields difference image. After scaling the difference image, the blocks that were compressed with lower quality factor Q2 will appear as black blocks whereas the other blocks will be decoded as white blocks.

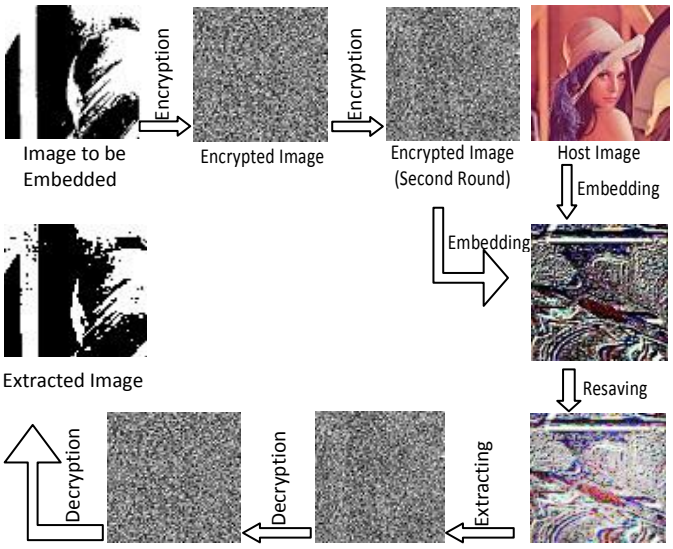

Fig 3: Embedding and extracting message scheme.

The difference between the resaved image and original image is small and hence the difference image will be very dark and blurred. Based on thresholding, we can extract the hidden message from the Host image. This process is as follows:

Step 1:

The embedded image is resaved with lower quality factor Q2.

Step 2:

In the second step we have to subtract the above resaved image from the original image and we get the difference image.

Step 3:

The difference image is now divided into the blocks i.e., 8x8 blocks.

Step 4:

We take the greatest sum values as s1 and least sum values as s2 in the difference image. Now we calculate s1-s2 value which is the greatest difference in the difference image. And that value is the best candidate and the average of these s1 and s2 is taken as threshold.

Step 5:

We decode the pixel as black if the sum of the pixel value in that block is smaller than the threshold or else we decode it as white.

3. EXPERIMENTAL RESULTS

Fig 4 shows the example. Let the original image be of size 1024x1024 and the secret message is of size 128x128. Let us take the permutation parameters as e=2 and k=83 and for decoding we need to take the parameters as e=2 and k=109.

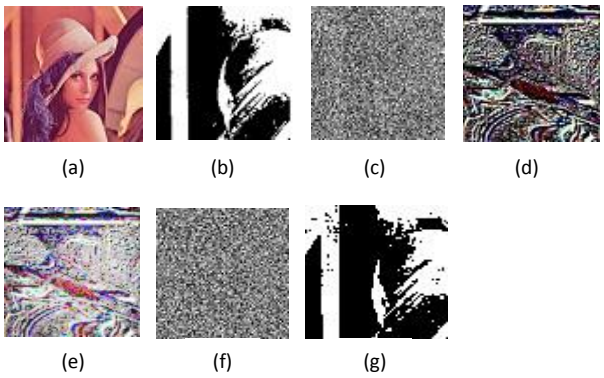

Fig 4: Example of embedding/extracting processes: (a) Original image (b) Secret message (c) Secret message after encryption (d) Embedded image (e) Difference image after scaling 25 times (g) Extracted embedded pattern, and (h) Secret message after decoding.

By comparing the host image and embedded image and the Correct Decoding Rate (CDR) value, we get PSNR that is used to assess the performance of the proposed scheme. The ratio of the number of pixels decoded correctly and the number of total pixels in the secret image gives the CDR value. In fig 4, the quality factors taken are Q1=95 andQ2=80 and PSNR value is 42.25 dB. The resaved image is scaled 25 times.

The above table shows the PSNR value with different Q1 and Q2. The PSNR value will be around 40dB if the difference between Q1 and Q2 is around 10 to 15.

Fig 5 shows another example with host image of size 512x512 and secret image with size 64x64. The quality factors taken are Q1=80 and Q2=55. In that we have used two pairs of parameters (e1,k1)=(1,40) and (e2,k2)=(2,66). In the receiver side, the permutation algorithm in reverse order is applied with parameters (e2,k2')=(2,30) and (e1,k1')=(1,8), so that we get the expected result.

TABLE I
PSNR (DB) WITH DIFFERENT SETS OF Q1 AND Q2

Q1 \ Q2	95	90	85	80	75	70	65	60	55
90	44.1								
85	43.2	43.4							
80	42.2	41.8	43.2						
75	41.2	41.3	41.6	43.2					
70	40.7	41.0	40.7	41.5	43.3				
65	40.2	40.3	40.3	40.6	41.4	43.0			
60	39.7	39.8	40.1	39.9	40.5	41.4	43.1		
55	39.4	39.5	40.0	39.6	39.8	40.4	41.3	43.0	
50	39.0	39.2	39.4	39.4	39.2	39.7	40.2	41.0	42.7
45	38.7	38.8	38.9	39.2	38.9	39.1	39.5	39.9	40.8
40	38.3	38.4	38.4	39.0	38.6	38.5	38.7	39.0	39.5
35	37.8	37.9	38.1	38.2	38.5	38.3	38.0	38.3	38.5

In the fig 5, the PSNR calculated for embedded image is 39.8dB and CDR is 94.4%. QTs are responsible for getting the correct extraction rate. From the Table II, we can observe the CDR with different pairs of Q1 and Q2. The selectable difference between Q1 and Q2 must be very small and range is about 20 to 40. If the difference is very small then the CDR cannot be 100% in most case because of the poor contrast of the difference image.

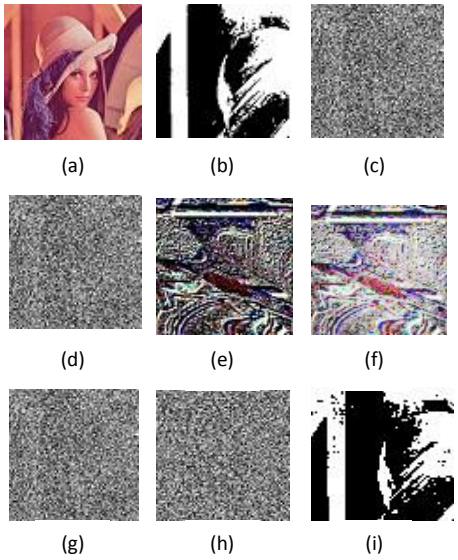

Fig 5: Two-round encryption result. (a) Original image, (b) Secret message, (c)–(d) Secret message after the first and second encryption round, (e) Embedded image, (f) Difference image after scaling 25 times, (g) Extracted patterns, (h)–(i) Decoded secret message.

In table III, the difference (Q1-Q2) vs. average PSNR and CDR is shown. High CDR is very essential and for that Q1 and Q2 should be correctly selected while maintaining the quality of the embedded image. The output embedded JPEG images are tested in order to evaluate the capability of evade steganalyser schemes.

The testing is done using two schemes: Stegdetect [9] and Farid's scheme [10]. Stegdetect is the scheme which determines whether the steganographic content is present and it also checks whether the hidden information is embedded using the system. Outguess, jphide and jsteg are the systems that stegdetect can easily identify. In the Farid's scheme, wavelet coefficients over N subbands are used. But generally the result obtained from steghide is very impressive.

TABLE II
CDR (%) WITH DIFFERENT SETS OF Q1 AND Q2

Q1 \ Q2	100	95	90	85	80	75	70	65	60
95	79.8								
90	85.2	78.7							
85	90.1	84.5	80.6						
80	92.7	88.6	84.1	81.2					
75	95.3	91.4	89.5	84.7	83.1				
70	96.1	92.9	92.4	86.9	84.4	83.0			
65	97.2	94.8	93.9	90.6	87.3	84.3	85.4		
60	98.1	96.8	95.0	93.7	89.1	88.0	88.9	87.3	

It returns negative results for different pairs of Q1 and Q2 for all the embedded images. Stegdetect gives the output image into innocent group even with the quality difference value as 70. Farid's scheme's detection rate performance will be 4.7% but it increases to the percentage of 13.3% if (Q1-Q2) value is over 60. The experimental result shows that the proposed scheme has evaluated the capability of steganalyser schemes.

TABLE III
AVERAGE PSNR AND CDR (%) VERSUS DIFFERENCE BETWEEN Q1 AND Q2

Q1-Q2	5	10	15	20	25	30	35	40
PSNR	43.3	41.6	40.6	39.6	39.3	39	38.8	38.3
CDR	82.8	85.6	88.4	91.2	93.1	94.6	96.5	97.5

4. CONCLUSION

The proposed scheme is very practical and it does not use any traditional space like DCT domain. The switching between the two different quality factors combined with encryption is done in proposed scheme. The reason for using encryption procedure is to enhance the security. The strength of the encryption key is very important in this scheme. The purpose of embedding technique using different QTs is adding a more protective layer for our secret information which is hidden in the JPEG image. Now-a-days, this scheme is very practical for maintaining secrecy and privacy for sender's information. Further it is suggested to make use of complex encryption procedures to provide more security and reliability.

5. REFERENCES

[1]. K. Raja and C. Chowdary, "A secure image steganography using LSB, DCT and compression techniques on raw images," in *3rd Int. Conf. Intelligent Sensing and Information Processing*, 2005, pp. 170–176.

[2]. Z. Ni and Y. Shi, "Robust lossless image data hiding designed for semi-fragile image authentication," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 4, pp. 497–509, 2008.

[3]. G. Gul and F. Kurugollu, "A novel universal steganalyser design: "LogSv"," in *IEEE Int. Conf. Image Processing (ICIP 2009)*, Cairo, Egypt, 2009.

[4]. H. Farid, "Exposing digital forgeries from JPEG ghosts," *IEEE Trans. Inform. Forensics Security*, vol. 4, no. 1, pp. 154–160, Mar. 2009.

[5]. W. B. Pennebaker and J. L. Mitchell, *JPEG: Still Image Data Compression Standard*. New York: Springer, 1993.

[6]. [Online].Available: http://www.impulseadventure.com/ photo/jpeg-quantization.html

[7]. G. Voyatzis and I. Pitas, "Applications of toral auto-morphisms in image watermarking," in *Int. Conf. Image Processing, Proceedings*, 1996, vol. 1, pp. 237–240.

[8]. H. K. Tso *et al.*, "A lossless secret image sharing method," in *8th Int. Conf. Intelligent Systems Designs and Application*, 2008, pp. 616–619.

[9]. [Online]. Available: Detection framework can be found at http://www.citi.umich.edu/u/-provos/papers/detecting. pdf. Supporting technical document and the utility can be downloaded at http://www.outguess.org

[10]. S. Lyu and H. Farid, "Steganalysis using color wavelet statistics and one-class support vector machine," in *SPIE Symp. Electronic Imaging*, 2004 [Online]. Available: http://www.cs.dartmouth.edu/farid/research/